Lecture Notes 2: Polynomial Interpolation

CPSC 303: Numerical Approximation and Discretization

Ian M. Mitchell
mitchell@cs.ubc.ca
http://www.cs.ubc.ca/~mitchell

University of British Columbia
Department of Computer Science

Winter Term Two 2012–2013

Outline

- Background
  - Problem statement and motivation
  - Formulation: The linear system and its conditioning
- Polynomial bases
  - Monomial
  - Lagrange
  - Newton
  - Uniqueness of polynomial interpolant
- Divided differences
  - Divided difference tables and the Newton basis interpolant
  - Divided difference connection to derivatives
- Osculating interpolation: interpolating derivatives
- Error analysis for polynomial interpolation
  - Reducing the error using the Chebyshev points as abscissae

Interpolation Motivation

We are given a collection of **data samples** $\{(x_i, y_i)\}_{i=0}^n$

- The $\{x_i\}_{i=0}^n$ are called the **abscissae** (singular: abscissa), the $\{y_i\}_{i=0}^n$ are called the **data values**
- Want to find a function $p(x)$ which can be used to estimate $y(x)$ for $x \neq x_i$
- Why? We often get discrete data from sensors or computation, but we want information as if the function were not discretely sampled
- If possible, $p(x)$ should be inexpensive to evaluate for a given x

Interpolation Formulation

There are lots of ways to define a function $p(x)$ to approximate $\{(x_i, y_i)\}_{i=0}^n$

- **Interpolation** means $p(x_i) = y_i$ (and we will only evaluate $p(x)$ for $\min_i x_i \leq x \leq \max_i x_i$)
- Most interpolants (and even general data fitting) is done with a linear combination of (usually nonlinear) **basis functions** $\{\phi_j(x)\}$

$$p(x) = p_n(x) = \sum_{j=0}^{n} c_j \phi_j(x)$$

where c_j are the **interpolation coefficients** or **interpolation weights**

How Many Ways to Interpolate?

Given data samples $\{(y_i, t_i)\}_{i=0}^{6}$ (eg: $n+1=7$ samples), how many different mathematical functions can interpolate this data?

(A) One: A polynomial of suitable degree.

(B) Two: Either a polynomial of suitable degree, or the Fourier transform.

(C) Seven, because there are seven samples.

(D) An infinite number.

A B C D

Linear System for Interpolation

Our interpolant is $p(x) = \sum_{j=0}^{n} c_j \phi_j(x)$

- From interpolation condition

$$p(x_i) = \sum_{j=0}^{n} c_j \phi_j(x_i) = y_i \qquad i = 0, 1, \ldots, n$$

  which leads to the linear system $Ac = y$ where

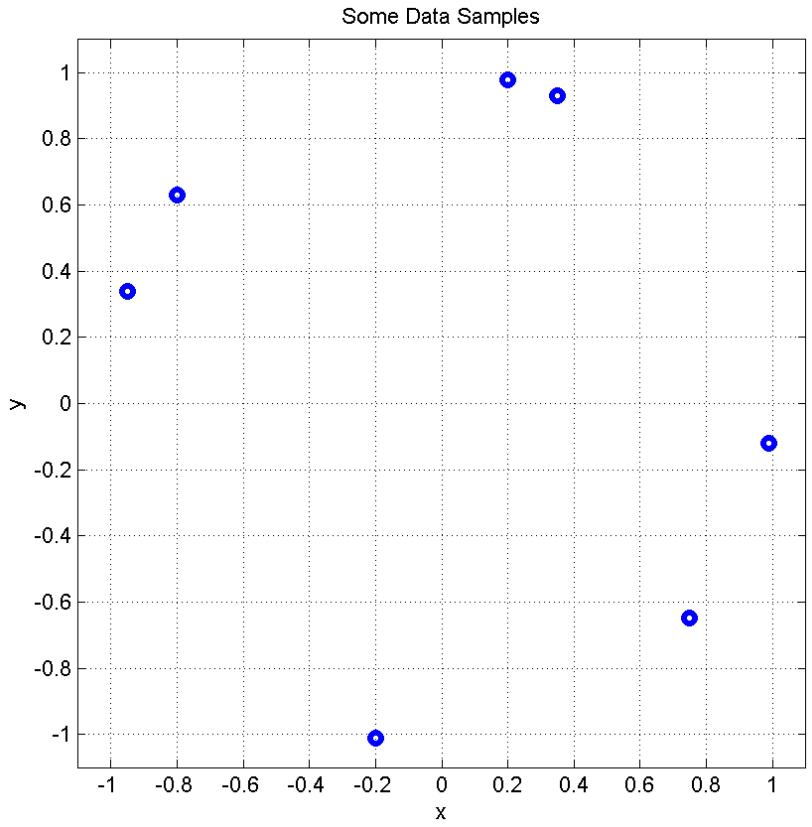
$$A = \begin{bmatrix} \phi_0(x_0) & \phi_1(x_0) & \cdots & \phi_n(x_0) \\ \phi_0(x_1) & \phi_1(x_1) & \cdots & \phi_n(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(x_n) & \phi_1(x_n) & \cdots & \phi_n(x_n) \end{bmatrix} \quad c = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{bmatrix} \quad y = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}$$

- When can we accurately solve this linear system?

How many basis functions?

How many basis functions do we need to interpolate this data set?

(A) Six.
(B) Seven.
(C) Eight.
(D) An infinite number.



A B C D

Determining the Interpolation Coefficients

Under what conditions can we find a unique set of interpolation coefficients $\{c_j\}_{j=0}^n$?

(A) If the basis functions $\{\phi_j\}_{j=0}^n$ are linearly independent.

(B) If A is nonsingular (eg: $\det(A) \neq 0$).

(C) If for any $x \neq 0$, $Ax \neq 0$.

(D) None of the above.

A B C D

Linear System Conditioning Digression

Consider solving linear system $Ax = b$ given A and b

- Due to approximation error, we actually use a slight perturbation $\hat{A} = A + \Delta A$
  - The solution to the perturbed system is $\hat{x} = x + \Delta x$
  - So $b = \hat{A}\hat{x} = A\hat{x} + \Delta A\hat{x}$ implies $b - A\hat{x} = \Delta A\hat{x}$
  - Then $\Delta x = \hat{x} - x = A^{-1}A\hat{x} - A^{-1}b = A^{-1}(A\hat{x} - b) = -A^{-1}\Delta A\hat{x}$
  - Now take norms

$$\|\Delta x\| \leq \|A^{-1}\|\|\Delta A\|\|\hat{x}\|$$

$$\underbrace{\frac{\|\Delta x\|}{\|\hat{x}\|}}_{\text{output error}} \leq \|A^{-1}\|\|A\|\frac{\|\Delta A\|}{\|A\|} = \text{cond}(A)\underbrace{\frac{\|\Delta A\|}{\|A\|}}_{\text{input error}}$$

    where $\text{cond}(A) = \|A^{-1}\|\|A\|$ is the definition of the **condition number** of matrix A

- Perturbations on b have a similar effect, so the overall result is roughly (assuming $\|\hat{x}\| \approx \|x\|$)

$$\frac{\|\Delta x\|}{\|x\|} \lesssim \text{cond}(A)\left(\frac{\|\Delta b\|}{\|b\|} + \frac{\|\Delta A\|}{\|A\|}\right)$$

Choosing Basis Functions

What should we use for the $\{\phi_j\}$?

- Many possibilities: polynomials, trigonometric, exponential, rational (fractions), wavelets / curvelets / ridgelets, radial basis functions, ...
- We will focus for now on polynomial basis functions
  - Most commonly used
  - Easy to evaluate, integrate, differentiate
  - Illustrates the basic interpolation ideas and techniques
- Virtually all other interpolation problems will follow the same procedure: form A and solve for c
- Now we are ready to examine particular polynomial bases which are often chosen due to their simplicity, efficiency, numerical robustness, extensibility, etc.

Monomial Basis

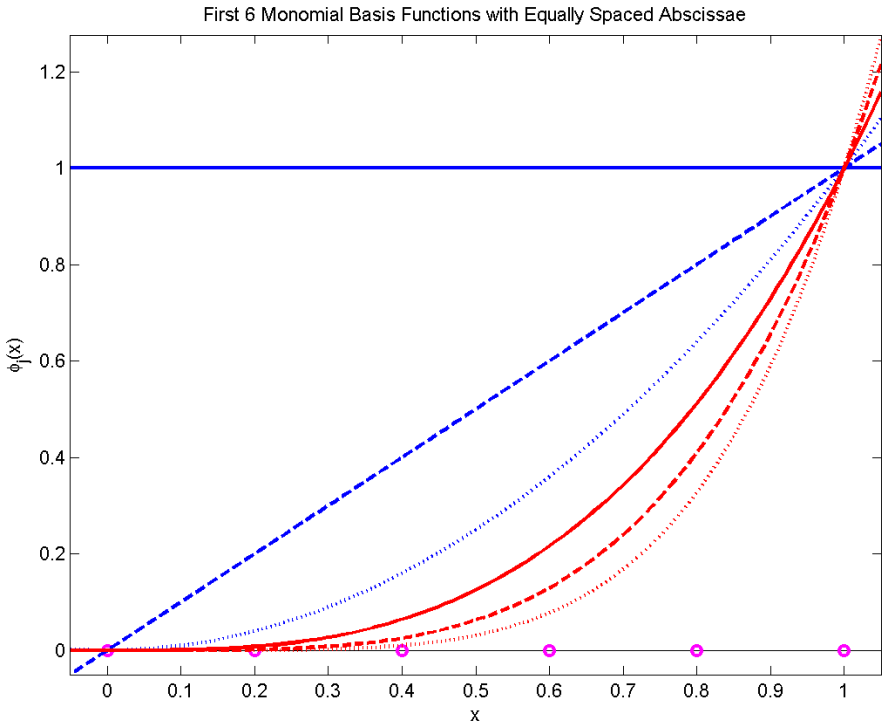
The monomial basis functions are typically defined as $\phi_j(x) = x^j$

- We could also scale and shift: $\tilde{\phi}_j(x) = s_j(x - \tilde{x})^j$ for fixed known scaling constants $\{s_j\}_{j=0}^n$ and shift constant $\tilde{x}$
- Entries of A are $a_{ij} = \phi_j(x_i) = (x_i)^j$
- Matrix is dense: $\mathcal{O}(n^2)$ to construct and $\mathcal{O}(n^3)$ to solve
- In fact, this particular matrix appears so often, it has a name: "Vandermonde" matrix (in MATLAB: `vander`)
- Interpolant $p(x)$ can be evaluated in $\mathcal{O}(n)$ with Horner's rule

Monomial Basis Conditioning

The monomial basis (and the corresponding Vandermonde matrix) are known to be very poorly conditioned as n gets large and/or $\{x_i\}$ cover a large range.

- Notice that $\phi_j(x)$ starts to look very similar to $\phi_{j-1}(x)$ and $\phi_{j+1}(x)$ as j gets larger.
- Conditioning can be improved somewhat by scaling and/or shifting basis functions



Monomial Example

Construct an interpolant for data points

$$\{(x_i, y_i)\} = \{(2, 14), (6, 24), (4, 25), (7, 15)\}$$

using the monomial basis.

- Requires four basis functions: $\{\phi_j(x)\} = \{1, x, x^2, x^3\}$
- The interpolant will be $p(x) = c_0 + c_1 x + c_2 x^2 + c_3 x^3$
- Construct linear system

$$A = \begin{bmatrix} 1 & 2 & 4 & 8 \\ 1 & 6 & 36 & 216 \\ 1 & 4 & 16 & 64 \\ 1 & 7 & 49 & 343 \end{bmatrix} \quad c = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \quad y = \begin{bmatrix} 14 \\ 24 \\ 25 \\ 15 \end{bmatrix}$$

  and solve $Ac = y$ to find $c \approx \begin{bmatrix} -0.267 & 1.700 & 2.767 & 3.800 \end{bmatrix}^T$
- Check cond$(A) \approx 6.1(10^3)$
  - However, if we scaled all the abscissae $\tilde{x}_i = 1000 x_i$, the resulting cond$(\tilde{A}) \approx 4.6(10^{12})$

Monomial Interpolation Practice

For the data set

$$\{(x_i, y_i)\} = \{(1, 11), (3, 13), (2, 12)\}$$

and the monomial basis functions $\phi_j(x) = x^j$, what is the form of the matrix A in $Ac = y$?

(A) $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 3 & 2 \\ 1 & 9 & 4 \end{bmatrix}$

(B) $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 3 & 2 \\ 11 & 13 & 12 \end{bmatrix}$

(C) $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 3 & 9 \\ 1 & 2 & 4 \end{bmatrix}$

(D) $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 3 & 2 \\ 1 & 9 & 4 \\ 11 & 13 & 12 \end{bmatrix}$

A B C D

Lagrange Basis

The best conditioned matrix is the identity.

- In order for $A = I$, we need

$$a_{ij} = \phi_j(x_i) = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

- We can achieve this by choosing the **Lagrange** basis functions

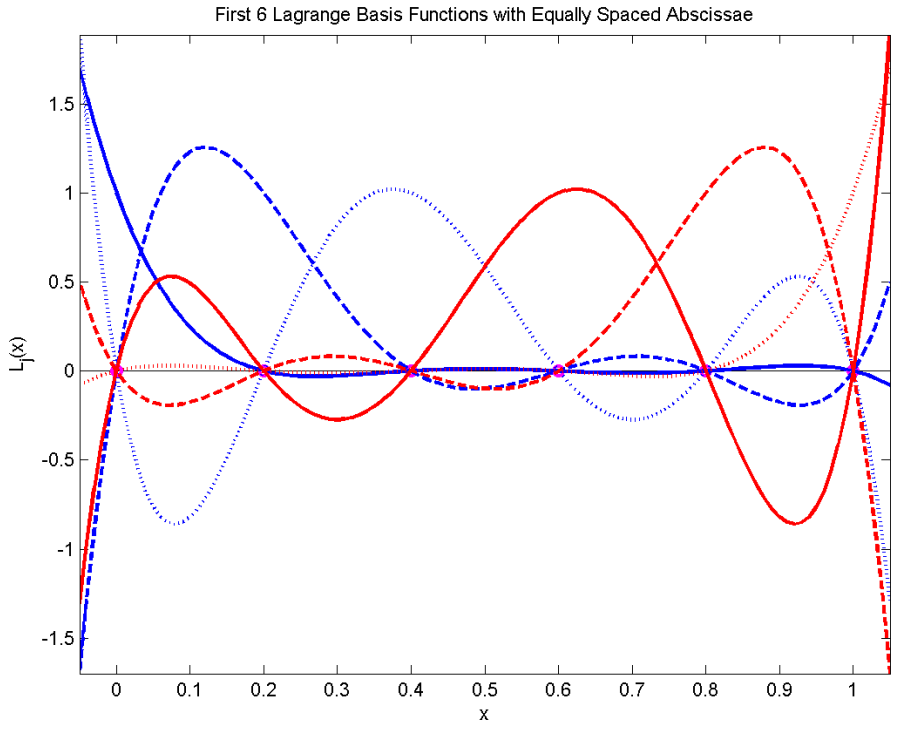
$$L_j(x) = \prod_{i=0, i \neq j}^{n} \frac{(x - x_i)}{(x_j - x_i)}$$

  - Numerator ensures that $a_{ij} = L_j(x_i) = 0$ for $i \neq j$
  - Denominator normalizes to get $a_{jj} = L_j(x_j) = 1$

- With $A = I$, no need to solve linear system: $c_j = y_j$ and $p(x) = \sum_{j=0}^{n} y_j L_j(x)$
- Pros & Cons:
  - Trivial to determine and modify interpolation coefficients
  - Nontrivial to compute normalization constants or add new data points

Lagrange Basis Conditioning

The Lagrange basis functions $L_j(x)$ are clearly distinct. With abscissae $x_i = i/5$ for $i = 0, 1, \ldots, 5$ the Lagrange basis function $L_j(x)$ are shown below.



Lagrange Example

Construct an interpolant for data points

$$\{(x_i, y_i)\} = \{(2, 14), (6, 24), (4, 25), (7, 15)\}$$

using the Lagrange basis.

- Four basis functions

$$L_0(x) = \frac{(x-6)(x-4)(x-7)}{(2-6)(2-4)(2-7)} \quad L_1(x) = \frac{(x-2)(x-4)(x-7)}{(6-2)(6-4)(6-7)}$$
$$L_2(x) = \frac{(x-2)(x-6)(x-7)}{(4-2)(4-6)(4-7)} \quad L_3(x) = \frac{(x-2)(x-6)(x-4)}{(7-2)(7-6)(7-4)}$$

- Interpolant will be

$$\begin{aligned} p(x) =& 14\frac{(x-6)(x-4)(x-7)}{(-4)(-2)(-5)} + 24\frac{(x-2)(x-4)(x-7)}{(+4)(+2)(-1)} \\ &+ 25\frac{(x-2)(x-6)(x-7)}{(+2)(-2)(-3)} + 15\frac{(x-2)(x-6)(x-4)}{(+5)(+1)(+3)} \end{aligned}$$

- Scaling abscissae $\tilde{x}_i = 1000 x_i$ will produce different bases $\tilde{L}_j(x)$, but the accuracy of the coefficients will remain the same

Lagrange Interpolation Practice

For the data set

$$\{(x_i, y_i)\} = \{(1, 11), (3, 13), (2, 12)\},$$

what is the Lagrange interpolant?

(A) $11\frac{(x-2)(x-3)}{(-1)(-2)} + 12\frac{(x-1)(x-3)}{(+1)(-1)} + 13\frac{(x-1)(x-2)}{(+2)(+1)}$

(B) $11\frac{(x-2)(x-3)}{(-1)(+2)} + 12\frac{(x-1)(x-3)}{(-1)(+1)} + 13\frac{(x-1)(x-2)}{(-2)(+1)}$

(C) $1\frac{(x-12)(x-13)}{(-1)(-2)} + 2\frac{(x-11)(x-13)}{(+1)(-1)} + 3\frac{(x-11)(x-12)}{(+2)(+1)}$

(D) $1\frac{(x-12)(x-13)}{(-1)(+2)} + 2\frac{(x-11)(x-13)}{(-1)(+1)} + 3\frac{(x-11)(x-12)}{(-2)(+1)}$

A B C D

Newton Basis

Can we add a new data point without changing the entire interpolant?

- Need $n \to n+1$, would prefer well-conditioned and easy to construct and evaluate
- In order to easily add points, we need $\phi_j(x)$ to have certain properties:
  - New basis function cannot disturb prior interpolation: $\phi_j(x_i) = 0$ for $i < j$
  - Old basis function does not need information about new data values: $\phi_j(x)$ is independent of (x_i, y_i) for $i > j$
- Newton basis function

$$\phi_j(x) = \prod_{i=0}^{j-1} (x - x_i)$$

- Leads to special form for matrix A
  - A is not necessarily well-conditioned, but will not be worse than the monomial basis and can be quite good if the order of data values is chosen wisely

Newton Interpolation Matrix

What is the form of the matrix A in $Ac = y$ for the Newton basis functions, and how expensive is it to solve the linear system?
("Tridiagonal" means that the matrix only has entries on the main diagonal and the two diagonals immediately above and below; consequently, there are at most three nonzeros on each row or column.)

(A) A is diagonal, $\mathcal{O}(n)$ to solve.

(B) A is tridiagonal, $\mathcal{O}(n^2)$ to solve.

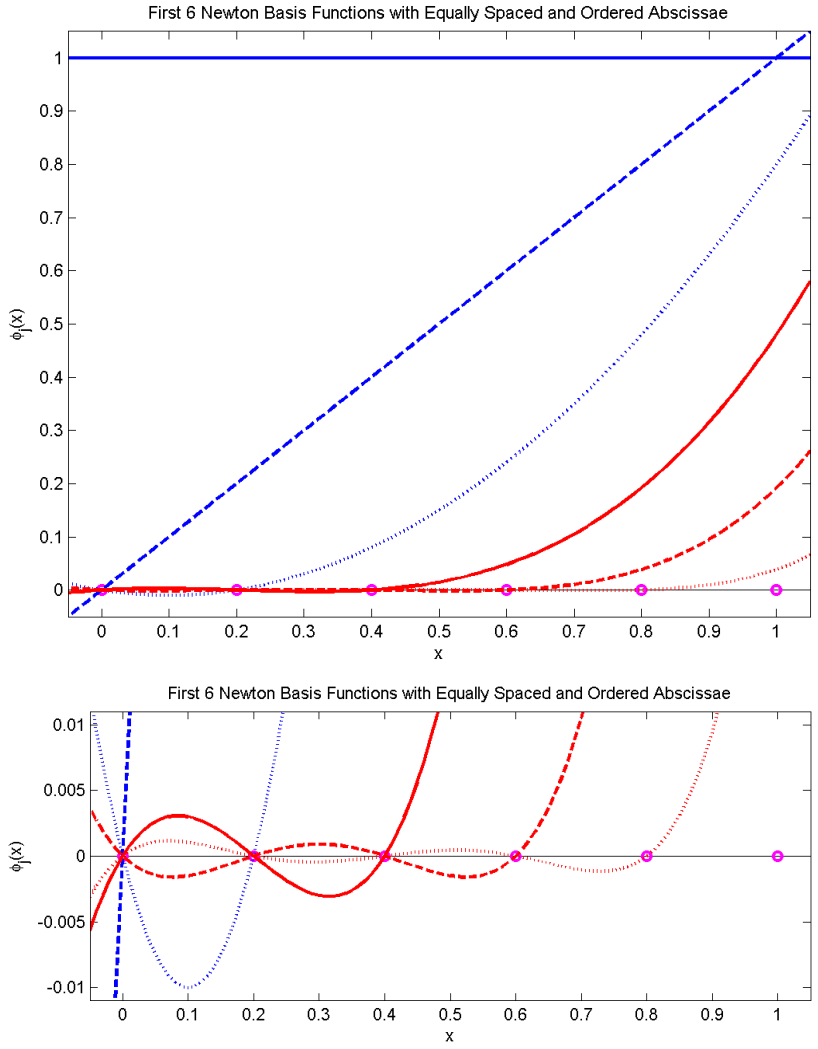
(C) A is lower triangular, $\mathcal{O}(n^2)$ to solve.

(D) None of the above.

A B C D

Newton Basis Conditioning

- We know that the Newton basis functions are linearly independent because $\phi_j(x)$ has exactly degree $j-1$.
- With abscissae $x_i = i/5$ for $i = 0, 1, \ldots, 5$ the Newton basis functions $\phi_j(x)$ are shown at right (vertically zoomed view on the bottom).
- Visually, they are not as distinct as the Lagrange basis functions but they are better than the monomials.



Newton Example

Construct an interpolant for data points

$$\{(x_i, y_i)\} = \{(2, 14), (6, 24), (4, 25), (7, 15)\}$$

using the Newton basis.

- Four basis functions

$$\phi_0(x) = 1 \qquad \phi_1(x) = (x - 2)$$
$$\phi_2(x) = (x - 6)(x - 2) \qquad \phi_3(x) = (x - 4)(x - 6)(x - 2)$$

- Construct linear system

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 4 & 0 & 0 \\ 1 & 2 & -4 & 0 \\ 1 & 5 & 5 & 15 \end{bmatrix} c = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \quad y = \begin{bmatrix} 14 \\ 24 \\ 25 \\ 15 \end{bmatrix}$$

  and solve $Ac = y$ to find $c \approx \begin{bmatrix} 14 & 2.5 & -1.5 & -0.2667 \end{bmatrix}^T$
- Check $\text{cond}(A) \approx 17$
  - However, if we scaled all the abscissae $\tilde{x}_i = 1000 x_i$, the resulting $\text{cond}(\tilde{A}) \approx 4.6(10^{12})$ is just as bad as for monomial basis

Newton Interpolation Practice

For the data set

$$\{(x_i, y_i)\} = \{(1, 11), (3, 13), (2, 12)\}$$

and the Newton basis functions $\phi_j(x) = \prod_{i=0}^{j-1}(x - x_i)$, what is the form of the matrix A in $Ac = y$?

(A) $\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \\ 0 & 0 & -1 \end{bmatrix}$

(B) $\begin{bmatrix} 1 & 0 & 0 \\ 1 & 2 & 0 \\ 1 & 1 & -1 \end{bmatrix}$

(C) $\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 2 & -1 \end{bmatrix}$

(D) $\begin{bmatrix} 1 & 0 \\ 1 & 2 \end{bmatrix}$

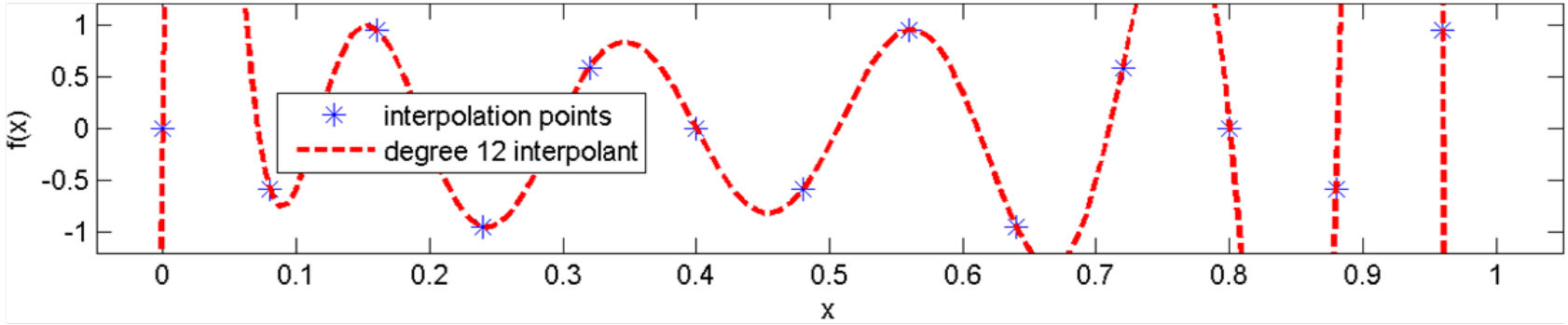
A B C D

Unique Polynomial Interpolants

Polynomial interpolant of degree n is unique: any basis will produce the same interpolant (assuming no computational errors).

- Let $q(x)$ and $p(x)$ be two polynomial interpolants of degree n such that interpolation condition holds: $p(x_i) = y_i = q(x_i)$ for all $i = 0, 1, \ldots, n$
- Then $u(x) = q(x) - p(x)$ is also a polynomial of degree n
  - Note that $u(x_i) = q(x_i) - p(x_i) = y_i - y_i = 0$ for all $i = 0, 1, \ldots, n$, so $u(x)$ has $n + 1$ zeros
  - The only such polynomial of degree n is $u(x) \equiv 0$, which implies $q(x) = p(x)$
- But $p(x) = \sum_{j=0}^{n} c_j \phi_j(x)$
  - For Lagrange basis $c_j = y_j$
  - For Newton and monomial bases we had to solve a linear system $Ac = y$ with $A \neq I$, so for these bases $c_j \neq y_j$
  - How can this be true if $p(x)$ is unique?
- Demonstration code: `interpolateSine`

Interpolation Error (part 1)



The plot above shows a polynomial interpolant of 13 data points built using the monomial basis. Based on the fact that the data values all lie in the interval $[1, +1]$, we do not really believe that the wild oscillations beyond this range shown by the interpolant accurately reflect the behaviour of the underlying process. Based on our definitions of various types of error, how would you classify this large apparent error?

(A) Computational error.

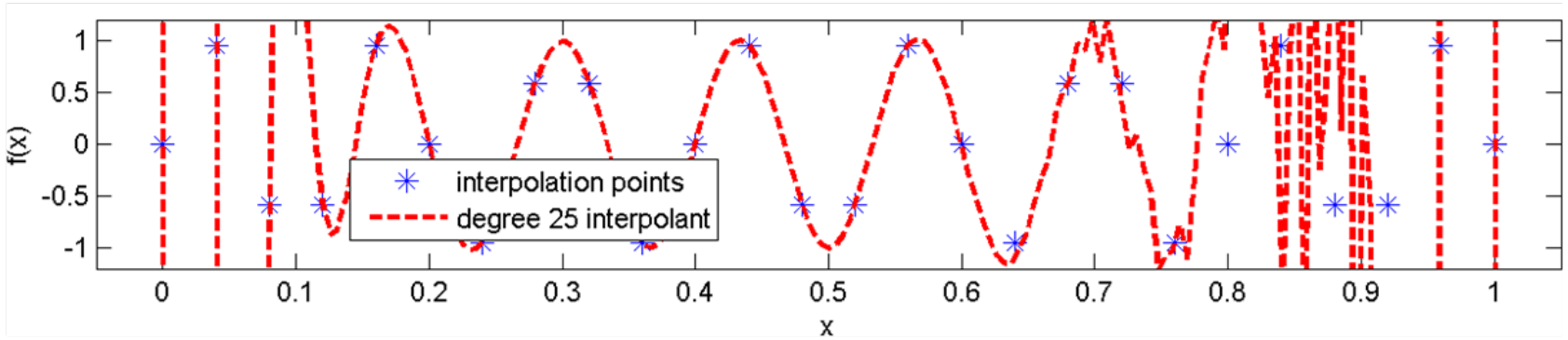
(B) Forward error.

(C) Relative error.

(D) None of the above.

A B C D

Interpolation Error (part 2)



The plot above shows a polynomial interpolant of 26 data points built using the monomial basis. Based on the fact that the data values all lie in the interval $[1, +1]$, we do not really believe that the wild oscillations beyond this range shown by the interpolant accurately reflect the behaviour of the underlying process. Based on our definitions of various types of error, how would you classify this large apparent error?

(A) Computational error.

(B) Forward error.

(C) Relative error.

(D) None of the above.

A B C D

Basic Polynomial Interpolation Summary

- We want to interpolate data $\{(x_i, y_i)\}_{i=0}^n$ using basis function set $\{\phi_j(x)\}_{j=0}^n$
  - Interpolant is $p_n(x) = \sum_{j=0}^n c_j \phi_j(x)$
  - Interpolation conditions $p_n(x_i) = y_i$ lead to square linear system $Ac = y$, where $a_{ij} = \phi_j(x_i)$
  - Solve linear system for coefficients c_j
- Interpolating polynomial is unique, but choice of basis set affects cost of construction and evaluation, accuracy, and difficulty of changing or adding data

basis set	construction cost	evaluation cost	bonus feature
monomial	$(1/3)n^3 + \mathcal{O}(n^2)$	n	simple
Lagrange	$n^2 + \mathcal{O}(n)$	$3n$	$c_j = y_j$
Newton	$n^2 + \mathcal{O}(n)$	n	adaptive n

Lagrange over Newton

The advantage of using the Lagrange basis functions rather than the Newton basis functions for polynomial interpolation is:

(A) The resulting linear system is lower triangular, and hence easier to solve.

(B) The basis functions do not need to recomputed if an additional data point is added.

(C) The resulting polynomial has smaller error.

(D) None of the above.

A B C D

Choosing Interpolant Bases I

Suppose that there are a dozen temperature sensors at fixed points along Interstate 5 between Blaine and Bellingham. Approximately every ten minutes each generates a new temperature reading, although readings from different sensors are not synchronized and rarely occur at the same time. We want an interpolant to estimate temperature as a function of position along the interstate. We would like it to incorporate the most recent reading from all sensors and be easy to update as new readings arrive. Which polynomial basis set would be best for this task?

(A) Monomial with $\phi_j(x) = x^j$, where x measures kilometers from Blaine.

(B) Monomial with $\phi_j(x) = (x - x_{\text{mid}})^j$, where x measures kilometers from Blaine and x_{mid} is half the distance between Blaine and Bellingham.

(C) Lagrange.

(D) Newton.

A B C D

Choosing Interpolant Bases II

We are constructing a plot of the temperature with respect to time outside Dempster for today, starting at 08:00 and continuing until 18:00. After every class we go outside and record the current temperature and the time at which that reading was made. We want to keep a plot of our interpolant—up to and including the most recent reading—on our web site, but we do not have long between classes so we need to minimize the amount of work done when each new reading becomes available. Which polynomial basis set would you recommend?

(A) Monomial with $\phi_j(t) = t^j$.

(B) Monomial with $\phi_j(t) = (t - 13:00)^j$

(C) Lagrange.

(D) Newton.

A B C D

Divided Differences

- An alternative method of determining the coefficients for a Newton basis interpolating polynomial
  - Used more often than the standard method of constructing and solving a linear system
  - Makes it easier to add and delete data points
- Have an interesting connection with function derivatives
  - A tool with which we will analyze interpolation error
- Are defined recursively

$$f[x_i] = y_i \qquad f[x_i, \ldots, x_j] = \frac{f[x_{i+1}, \ldots, x_j] - f[x_i, \ldots, x_{j-1}]}{x_j - x_i}$$

  - The coefficients for Newton interpolation are just $c_j = f[x_0, \ldots, x_j]$ (the diagonal elements in the table)
  - To add another data point ($n \to n+1$), just add another row to the table (assuming that the abscissae are unique)

Divided Difference Example

Given data $\{(x_i, y_i)\} = \{(0,0), (2,6), (1,0)\}$

- Divided difference table

i	x_i	$f[x_i]$	$f[x_{i-1}, x_i]$
0	$x_0 = 0$	$f[x_0] = y_0 = 0$	—
1	$x_1 = 2$	$f[x_1] = y_1 = 6$	$f[x_0, x_1] = \frac{f[x_1] - f[x_0]}{x_1 - x_0} = 3$
2	$x_2 = 1$	$f[x_2] = y_2 = 0$	$f[x_1, x_2] = \frac{f[x_2] - f[x_1]}{x_2 - x_1} = 6$

i	$f[x_{i-2}, x_{i-1}, x_i]$
0	—
1	—
2	$f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0} = 3$

- So the Newton interpolants are:

$$p_0(x) = 0$$
$$p_1(x) = 0 + 3(x - 0) = 3x$$
$$p_2(x) = 0 + 3(x - 0) + 3(x - 0)(x - 2) = 3x^2 - 3x$$

Divided Difference Example Continued

What if we add a new data point $(x_3, y_3) = (-1, 0)$?

- Add a new row and column to the divided difference table

x_i	$f[x_i]$	$f[x_{i-1}, x_i]$	$f[x_{i-2}, \ldots, x_i]$	$f[x_{i-3}, \ldots, x_i]$
0	0	—	—	—
$+2$	$+6$	$+3$	—	—
$+1$	0	$+6$	$+3$	—
-1	0	0	$+2$	$\frac{f[x_1,x_2,x_3]-f[x_0,x_1,x_2]}{x_3-x_0} = +1$

- Cubic Newton basis interpolant:

$$p_3(x) = 3x^2 - 3x + 1(x-0)(x-2)(x-1) = x^3 - x$$

Divided Difference Table Practice

Consider the data points

$$\{(x_i, y_i)\}_{i=0}^{n} = \{(1, 21), (2, 32), (4, 64), (8, 88)\}.$$

What are the missing entries in the divided difference table?

x_i	$f[x_i]$	$f[x_{i-1}, x_i]$	$f[x_{i-2}, \ldots, x_i]$	$f[x_{i-3}, \ldots, x_i]$
1	21	A		
2	32	11		
4	64	16	C	
8	88	B	$-5/3$	D

(A) $A = 11$, $B = 12$, $C = 5$, $D = -15$.

(B) A blank, $B = 6$, $C = 5/3$, $D = -10/21$.

(C) A blank, $B = 6$, $C = 5/2$, $D = -25/42$.

(D) A blank, $B = 6$, $C = 5/2$, $D = -25/6$.

A B C D

Divided Difference Interpolant Practice

Consider the data points

$$\{(x_i, y_i)\}_{i=0}^{n} = \{(+2, 1), (0, 3), (-2, 2), (+4, 12)\},$$

which generate the following divided difference table. What is the Newton interpolating polynomial?

x_i	$f[x_i]$	$f[x_{i-1}, x_i]$	$f[x_{i-2}, \ldots, x_i]$	$f[x_{i-3}, \ldots, x_i]$
2	1			
0	3	-1		
-1	2	1	$-2/3$	
4	12	2	$1/4$	$11/24$

(A) $12 + 2(x-2) + \frac{1}{4}(x-2)(x) + \frac{11}{24}(x-2)(x)(x+2)$

(B) $12 + 2(x-4) + \frac{1}{4}(x-4)(x+2) + \frac{11}{24}(x-4)(x+2)(x)$

(C) $1 - 1(x-2) - \frac{2}{3}(x-2)(x) + \frac{11}{24}(x-2)(x)(x+2)$

(D) $1 - 1(x-4) - \frac{2}{3}(x-4)(x+2) + \frac{11}{24}(x-4)(x+2)(x)$

A B C D

Divided Differences & Derivatives

Imagine that $y_i = f(x_i)$ for some unknown but sufficiently smooth function $f(x)$ (eg: $f(x)$ has enough derivatives)

- In these cases, we will write the data values as $\{(x_i, f(x_i))\}_{i=0}^{n}$
- There is a connection between the derivatives of f and the divided differences: if $\{x_i, \ldots, x_{i+k}\}$ are $k+1$ distinct points with $a = \min_\ell x_\ell$ and $b = \max_\ell x_\ell$, then there exists $\xi \in [a, b]$ such that

$$f[x_i, \ldots, x_{i+k}] = \frac{1}{k!} \frac{d^k f(\xi)}{dx^k} = \frac{f^{(k)}(\xi)}{k!}$$

  - For example, for $x_0 < x_1$ there exists ξ such that $x_0 \leq \xi \leq x_1$ and

$$f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0} = f'(\xi)$$

Divided Differences & Derivatives Practice

If you are given data samples

$$\{(x_i, f(x_i))\}_{i=0}^{n} = \{(1, 11), (5, 55), (3, 33), (2, 22)\}$$

which were drawn from some sufficiently smooth but unknown function $f(x)$, which of the following is true?

(A) $\exists \xi \in [1, 5]$ such that $f[1, 5, 3] = \frac{1}{2} f^{(2)}(\xi)$.

(B) $\exists \xi \in [1, 5]$ such that $f[1, 5, 3, 2] = \frac{1}{2} f^{(2)}(\xi)$.

(C) $\exists \xi \in [1, 3]$ such that $f[1, 5, 3] = \frac{1}{6} f^{(3)}(\xi)$.

(D) $\exists \xi \in [1, 3]$ such that $f[1, 5, 3] = \frac{1}{2} f^{(2)}(\xi)$.

A B C D

Divided Difference & Derivatives Example

- Divided difference table from earlier with data

$$\{(0,0),(2,6),(1,0),(-1,0)\}$$

actually used $y_i = f(x_i) = x_i^3 - x_i$ to generate the data values

  - So the true derivatives are $f'(x) = f^{(1)}(x) = 3x^2 - 1$, $f''(x) = f^{(2)}(x) = 6x$ and $f^{(3)}(x) = 6$
  - Observe

$$f[x_0, x_1] = 3 \Rightarrow \exists \xi \in [0,2] \text{ st } f'(\xi) = 3; \text{ eg: } \xi = \sqrt{4/3} \approx 1.15$$

$$f[x_1, x_2] = 6 \Rightarrow \exists \xi \in [1,2] \text{ st } f'(\xi) = 6; \text{ eg: } \xi = \sqrt{7/3} \approx 1.53$$

$$f[x_2, x_3] = 0 \Rightarrow \exists \xi \in [-1,+1] \text{ st } f'(\xi) = 0; \text{ eg: } \xi = \pm\sqrt{1/3} \approx \pm 0.58$$

$$f[x_0, x_1, x_2] = 3 \Rightarrow \exists \xi \in [0,2] \text{ st } 1/2 f''(\xi) = 3; \text{ eg: } \xi = 1$$

$$f[x_1, x_2, x_3] = 2 \Rightarrow \exists \xi \in [-1,2] \text{ st } 1/2 f''(\xi) = 2; \text{ eg: } \xi = \pm 2/3$$

$$f[x_0, x_1, x_2, x_3] = 1 \Rightarrow \exists \xi \in [-1,+2] \text{ st } 1/6 f^{(3)}(\xi) = 1; \text{ eg: any } \xi$$

Osculating Interpolation

What if we are given more than just the data value at each abscissae?

- For example, given $\{(x_i, y_i, y_i')\}_{i=0}^{q} = \{(x_i, f(x_i), f'(x_i))\}_{i=0}^{q}$
- We want the interpolant to match whatever data we have available; in this case

$$p(x_i) = \sum_{j=0}^{n} c_j \phi_j(x) = f(x_i) \qquad p'(x_i) = \sum_{j=0}^{n} c_j \phi_j'(x) = f'(x_i)$$

  - The resulting $p(x)$ is called an **osculating** interpolant
  - How many basis functions do we need?
- We can take this set of equations and form it into a linear system to solve for the coefficients
  - Works for any set of (linearly independent) basis functions
  - Works for any type of data; for example, given $f''(x_i)$ as well, or different numbers of derivatives and data values for each abscissa

Osculating Interpolation Linear System Practice

Given basis set $\{\phi_j(x)\}_{j=0}^{n}$ (where you can choose n) and the data set $\{(x_0, f(x_0), f'(x_0)), (x_1, f(x_1), f'(x_1))\}$, what linear system helps us construct an osculating interpolant?

(A)
$$\begin{bmatrix} \phi_0(x_0) & \phi_1(x_0) & \phi_2(x_0) & \phi_3(x_0) \\ \phi_0(x_1) & \phi_1(x_1) & \phi_2(x_1) & \phi_3(x_1) \\ \phi_0(x_0) & \phi_1(x_0) & \phi_2(x_0) & \phi_3(x_0) \\ \phi_0(x_1) & \phi_1(x_1) & \phi_2(x_1) & \phi_3(x_1) \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ f'(x_0) \\ f'(x_1) \end{bmatrix}$$

(B)
$$\begin{bmatrix} \phi_0(x_0) & \phi_1(x_0) & \phi_2(x_0) & \phi_3(x_0) \\ \phi_0'(x_0) & \phi_1'(x_0) & \phi_2'(x_0) & \phi_3'(x_0) \\ \phi_0(x_1) & \phi_1(x_1) & \phi_2(x_1) & \phi_3(x_1) \\ \phi_0'(x_1) & \phi_1'(x_1) & \phi_2'(x_1) & \phi_3'(x_1) \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f'(x_0) \\ f(x_1) \\ f'(x_1) \end{bmatrix}$$

(C)
$$\begin{bmatrix} \phi_0(x_0) & \phi_1(x_0) \\ \phi_0'(x_0) & \phi_1'(x_0) \\ \phi_0(x_1) & \phi_1(x_1) \\ \phi_0'(x_1) & \phi_1'(x_1) \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f'(x_0) \\ f(x_1) \\ f'(x_1) \end{bmatrix}$$

(D)
$$\begin{bmatrix} \phi_0(x_0) & \phi_1(x_0) & \phi_2(x_0) & \phi_3(x_0) \\ \phi_0(x_1) & \phi_1(x_1) & \phi_2(x_1) & \phi_3(x_1) \\ \phi_0'(x_0) & \phi_1'(x_0) & \phi_2'(x_0) & \phi_3'(x_0) \\ \phi_0'(x_1) & \phi_1'(x_1) & \phi_2'(x_1) & \phi_3'(x_1) \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ f'(x_0) \\ f'(x_1) \end{bmatrix}$$

Osculating Interpolant Practice

Given basis set $\{\phi_j(x)\}_{j=0}^n$ (where you can choose n) and the data set $\{(x_0, f'(x_0)), (x_1, f(x_1), f'(x_1)), (x_2, f(x_2))\}$, we solved the following linear system. What is the corresponding osculating interpolant?

$$\begin{bmatrix} \phi_0'(x_0) & \phi_1'(x_0) & \phi_2'(x_0) & \phi_3'(x_0) \\ \phi_0(x_1) & \phi_1(x_1) & \phi_2(x_1) & \phi_3(x_1) \\ \phi_0'(x_1) & \phi_1'(x_1) & \phi_2'(x_1) & \phi_3'(x_1) \\ \phi_0(x_2) & \phi_1(x_2) & \phi_2(x_2) & \phi_3(x_2) \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} f'(x_0) \\ f(x_1) \\ f'(x_1) \\ f(x_2) \end{bmatrix}$$

(A) $p(x) = \sum_{j=0}^{3} c_j \phi_j(x).$

(B) $p(x) = \sum_{j=0}^{1} c_j \phi_j(x) + \sum_{j=1}^{2} c_j \phi_j'(x).$

(C) $p(x) = c_0 \phi_0'(x) + c_1 \phi_1(x) + c_2 \phi_1'(x) + c_3 \phi_2(x).$

(D) $p(x) = \sum_{j=0}^{2} c_j \phi_j(x).$

A B C D

Osculating Interpolation & Divided Differences

We can also build an osculating interpolant for data of the form $\{(x_i, f(x_i), f'(x_i))\}_{i=0}^{q}$ with the Newton basis and a modified divided difference table

- Replicate abscissae: our new abscissae will be $\tilde{x}$

$$(\tilde{x}_0, \tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \ldots, \tilde{x}_{n-1}, \tilde{x}_n) = (x_0, x_0, x_1, x_1, \ldots, x_q, x_q),$$

  and use the standard Newton basis polynomial

$$p_n(x) = \sum_{j=0}^{n} f[\tilde{x}_0, \ldots, \tilde{x}_j] \left(\prod_{k=0}^{j-1} (x - \tilde{x}_k) \right)$$

- But we cannot compute $f[\tilde{x}_{k-1}, \tilde{x}_k]$ if $\tilde{x}_{k-1} = \tilde{x}_k$
  - Instead, take advantage of the relationship between divided differences and derivatives; for example

$$f[\tilde{x}_{k-1}, \tilde{x}_k] = \begin{cases} \frac{f[\tilde{x}_k] - f[\tilde{x}_{k-1}]}{\tilde{x}_k - \tilde{x}_{k-1}}, & \text{if } \tilde{x}_{k-1} \neq \tilde{x}_k; \\ \frac{f'(\tilde{x}_k)}{1!} & \text{if } \tilde{x}_{k-1} = \tilde{x}_k \end{cases}$$

Osculating Interpolation Example

Given data $x_0 = 20$, $y_0 = 100$, $y_0' = 10$, $y_0'' = 2$, $x_1 = 10$ and $y_1 = 0$, construct a divided difference table and then an osculating polynomial interpolant of maximum degree.

- Divided difference table (after replicating the abscissae as necessary)

$\tilde{x}_i$	$f[\tilde{x}_i]$	$f[\tilde{x}_{i-1}, \tilde{x}_i]$	$f[\tilde{x}_{i-2}, \ldots, \tilde{x}_i]$	$f[\tilde{x}_{i-3}, \ldots, \tilde{x}_i]$
20	100	—	—	—
20	100	10	—	—
20	100	10	1	—
10	0	10	0	0.1

- The resulting interpolant is determined just like the standard Newton basis interpolant from a divided difference table

$$
\begin{aligned}
p(x) &= \sum_{j=0}^{n} f[\tilde{x}_0, \ldots, \tilde{x}_j] \prod_{k=0}^{j-1} (x - \tilde{x}_k) \\
&= 100 + 10(x - 20) + (x - 20)^2 + 0.1(x - 20)^3
\end{aligned}
$$

Osculating Interpolation
Divided Different Table Practice

Fill in the divided difference table given data:

$$\{(x_i, y_i, y_i', y_i'')\}_{i=0}^{n} = \{(1.0, 1.1, 1.2, 1.4), (2.0, 2.1, 2.2, 2.4)\}$$

x_i	$f[\tilde{x}_i]$	$f[\tilde{x}_{i-1}, \tilde{x}_i]$	$f[\tilde{x}_{i-2}, \ldots, \tilde{x}_i]$	$f[\tilde{x}_{i-3}, \ldots, \tilde{x}_i]$	$f[\tilde{x}_{i-4}, \ldots, \tilde{x}_i]$
1	1.1				
1	1.1	B			
A	1.1	1.2	D		
2	2.1	C	$(1.0-1.2)/1$	-0.9	
2	2.1	2.2	$(2.2-1.0)/1$	1.4	2.3

(A) $A = 2,\ B = 0.0,\ C = 1.0,\ D = 1.4.$

(B) $A = 1,\ B = 1.2,\ C = 2.1,\ D = 1.4.$

(C) $A = 1,\ B = 1.2,\ C = 1.0,\ D = 0.7.$

(D) $A = 1,\ B = 0.0,\ C = 2.1,\ D = 0.7.$

A B C D

Osculating Interpolant Practice

What is the osculating interpolant generated by this divided difference table?

x_i	$f[\tilde{x}_i]$	$f[\tilde{x}_{i-1}, \tilde{x}_i]$	$f[\tilde{x}_{i-2}, \ldots, \tilde{x}_i]$	$f[\tilde{x}_{i-3}, \ldots, \tilde{x}_i]$	$f[\tilde{x}_{i-4}, \ldots, \tilde{x}_i]$
1	1.1				
1	1.1	1.2			
2	2.1	2.2	1.0		
2	2.1	2.2	2.0	1.0	
2	2.1	2.2	2.0	1.4	0.4

(A) $1.1+1.2(x-1)+1.0(x-1)(x-2)+1.0(x-1)^2(x-2)+0.4(x-1)^2(x-2)^2$

(B) $1.1+1.2(x-1)+1.0(x-1)^2+1.0(x-1)^2(x-2)+0.4(x-1)^2(x-2)^2$

(C) $2.1+2.2(x-1)+2.0(x-1)(x-2)+1.4(x-1)^2(x-2)+0.4(x-1)^2(x-2)^2$

(D) $2.1+2.2(x-1)+2.0(x-1)^2+1.4(x-1)^2(x-2)+0.4(x-1)^2(x-2)^2$

A B C D

(Forward) Error Analysis for Polynomial Interpolation Approximation

Assume that $\{y_i\}_{i=0}^n$ are drawn from some underlying but unknown function $f(x)$; in other words, $y_i = f(x_i)$

- We would like to estimate the error in the (unique) interpolating polynomial $p_n(x)$

$$e_n(x) = f(x) - p_n(x)$$

  and see how it depends on the choice of n and the properties of f
- To find $e(\tilde{x})$ for some $\tilde{x} \notin \{x_i\}_{i=0}^n$, pretend that we are adding the new data point $(\tilde{x}, f(\tilde{x}))$
  - Using the properties of the Newton basis and divided differences

$$f(\tilde{x}) = p_{n+1}(\tilde{x}) = p_n(\tilde{x}) + f[x_0, \ldots, x_n, \tilde{x}] \prod_{j=0}^{n} (\tilde{x} - x_j)$$

or by rearranging

$$e_n(\tilde{x}) = f(\tilde{x}) - p_n(\tilde{x}) = f[x_0, \ldots, x_n, \tilde{x}] \prod_{j=0}^{n} (\tilde{x} - x_j)$$

Polynomial Interpolation Error Analysis Continued

Error: $e_n(\tilde{x}) = f(\tilde{x}) - p_n(\tilde{x}) = f[x_0, \ldots, x_n, \tilde{x}] \prod_{j=0}^{n} (\tilde{x} - x_j)$

- Let $a = \min_i x_i$, $b = \max_i x_i$ and assume $\tilde{x} \in [a, b]$ (otherwise $p_n(\tilde{x})$ is "extrapolating")
- We had a relationship between divided differences and derivatives

$$\exists \xi \in [a, b] \text{ such that } f[x_0, \ldots, x_n, \tilde{x}] = \frac{f^{(n+1)}(\xi)}{(n+1)!}$$

- So we can take upper bounds to find

$$|e_n(\tilde{x})| \leq \max_{\xi \in [a,b]} \frac{|f^{(n+1)}(\xi)|}{(n+1)!} \max_{\zeta \in [a,b]} \left| \prod_{j=0}^{n} (\zeta - x_j) \right|$$

$$\|e_n\|_\infty \leq \frac{\|f^{(n+1)}\|_\infty}{(n+1)!} (b-a)^{n+1}$$

  - In CPSC 303, we will always use the **infinity norm** or **max norm** for functions: $\|g\|_\infty = \max_x |g(x)|$
- See Heath interactive modules on error bounds and convergence

Interpolation Error Example I

Consider interpolating $f(x) = \sin(2\pi x)$ on the interval $x \in [0, +1]$. From our error formula:

$$
\begin{aligned}
|e_0(x)| &\leq \frac{\|f'\|}{1!}(b-a) = \|2\pi\cos(2\pi x)\|(1) \leq 2\pi \\
|e_1(x)| &\leq \frac{\|f''\|}{2!}(b-a)^2 = \tfrac{1}{2}\|-4\pi^2\sin(2\pi x)\|(1)^2 \leq 2\pi^2 \\
|e_2(x)| &\leq \frac{\|f^{(3)}\|}{3!}(b-a)^3 = \tfrac{1}{6}\|-8\pi^3\cos(2\pi x)\|(1)^3 \leq \tfrac{4}{3}\pi^3 \\
|e_3(x)| &\leq \frac{\|f^{(4)}\|}{4!}(b-a)^4 = \tfrac{1}{24}\|16\pi^4\sin(2\pi x)\|(1)^4 \leq \tfrac{2}{3}\pi^4 \\
&\vdots \\
|e_n(x)| &\leq \frac{\|f^{(n+1)}\|}{(n+1)!}(b-a)^{n+1} = \frac{(2\pi)^{n+1}(1)^{n+1}}{(n+1)!}
\end{aligned}
$$

Since $(n+1)!$ will eventually dominate z^{n+1} for any fixed z, (in this case $z = 2\pi$) the error decreases as n increases and the interpolant gets better.

Interpolation Error Example II

In `interpolateSine.m` the actual function was $f(x) = \sin(20\pi x)$ so the same error bound works out to

$$|e_n(x)| \leq \frac{\|f^{(n+1)}\|}{(n+1)!}(b-a)^{n+1} = \frac{(20\pi)^{n+1}(1)^{n+1}}{(n+1)!}$$

- It is true that $(n+1)!$ will still eventually dominate $(20\pi)^{n+1}$, but n will have to be a lot bigger.
- In `interpolateSine.m` we used a monomial basis, so the linear system became very ill-conditioned around $n \approx 20$ (where n is still too small for the factorial to dominate).
- Therefore, we never got a good fit: by the time the factorial term dominates, the linear system is so ill-conditioned that roundoff error destroys any accuracy in the coefficients.
- We could overcome this problem with the Lagrange basis, although we will still need either small or large n to keep the error small.

Interpolation Error Example III

Now consider Runge's example $f(x) = (1 + 25x^2)^{-1}$ for $x \in [-1, +1]$. Although $(b - a) = 2$ and $\|f\| = 1$, the $\|f^{(n+1)}\|$ grow quickly (in fact, in a manner related to the factorial), so $|e_n(x)|$ grows rapidly with n (see `interpolateRunge.m` for confirmation)

$$
\begin{aligned}
|e_0(x)| &\leq \frac{\|f'\|}{1!}(b-a) = \| -50x(1-25x^2)^{-2}\|(2) \leq 100|x|\|f\|^2 \\
|e_1(x)| &\leq \frac{\|f''\|}{2!}(b-a)^2 \\
&= \frac{1}{2}\left\| \frac{5000x^2}{(1+25x^2)^3} - \frac{50}{(1+25x^2)^2} \right\| (2)^2 \leq 2500|x|^2\|f\|^3 - 25\|f\|^2 \\
|e_2(x)| &\leq \frac{\|f^{(3)}\|}{3!}(b-a)^3 \leq \tfrac{1}{6}\left(-750000|x|^3\|f\|^4 - 15000|x|\|f\|^3\right)(2)^3 \\
&\vdots
\end{aligned}
$$

Will Additional Data Reduce the Error?

Consider the following challenge:

- You have an interpolant $p_n(x)$ constructed from data samples $\{(x_i, f(x_i))\}_{i=0}^{n}$, where $x_i \in [a, b] = [0, 1]$
- You are given a new data sample $(x_{n+1}, f(x_{n+1}))$ with which you could form a new interpolant $p_{n+1}(x)$.
- Under what conditions on n, f, $a = 0$, $b = 1$, x_i and/or $f(x_i)$ can you guarantee that the bound on the error in $p_{n+1}(x)$ is smaller than the bound on the error in $p_n(x)$?

Chebyshev Points

Our error bound construction was not entirely fair.

$$\|e_n\|_\infty \leq \max_{\xi \in [a,b]} \frac{|f^{(n+1)}(\xi)|}{(n+1)!} \max_{\zeta \in [a,b]} \left| \prod_{j=0}^{n} (\zeta - x_j) \right| \leq \frac{\|f^{(n+1)}\|_\infty}{(n+1)!} (b-a)^{n+1}$$

- If we choose the x_j wisely, we can get a much lower bound on the term $\max_\zeta |\prod_j (\zeta - x_j)|$
- The best such x_j are called the Chebyshev points, which on the interval $[-1, +1]$ are

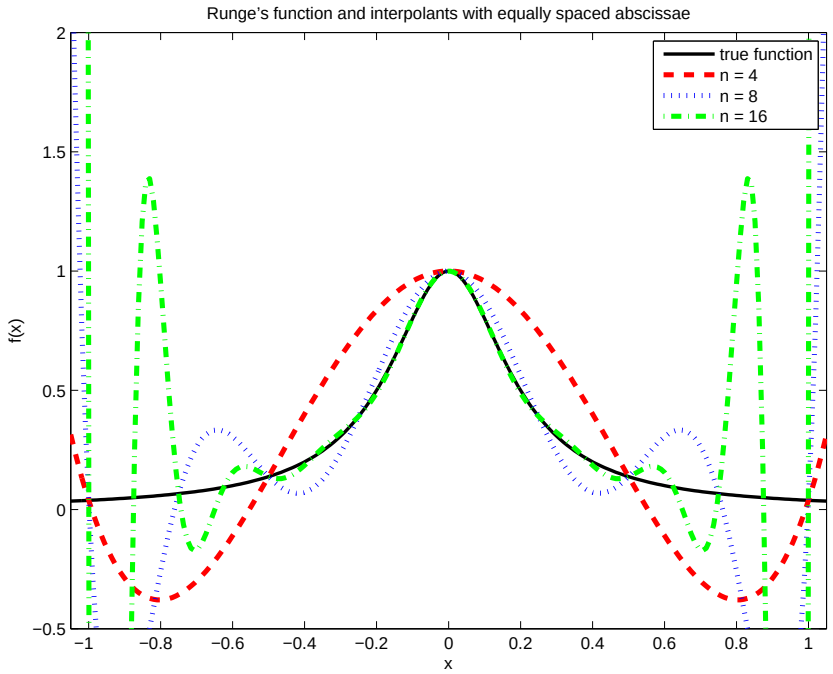
$$x_i = \cos\left(\frac{2i+1}{2(n+1)}\pi\right) \qquad i = 0, 1, 2, \ldots n$$

  - For another interval $[a, b]$ use $\tilde{x}_i = a + (1/2)(b-a)(x_i + 1)$
- Bound (for interval $[-1, +1]$) is $\max_\zeta |\prod_j (\zeta - x_j)| \leq 2^{-n}$ and

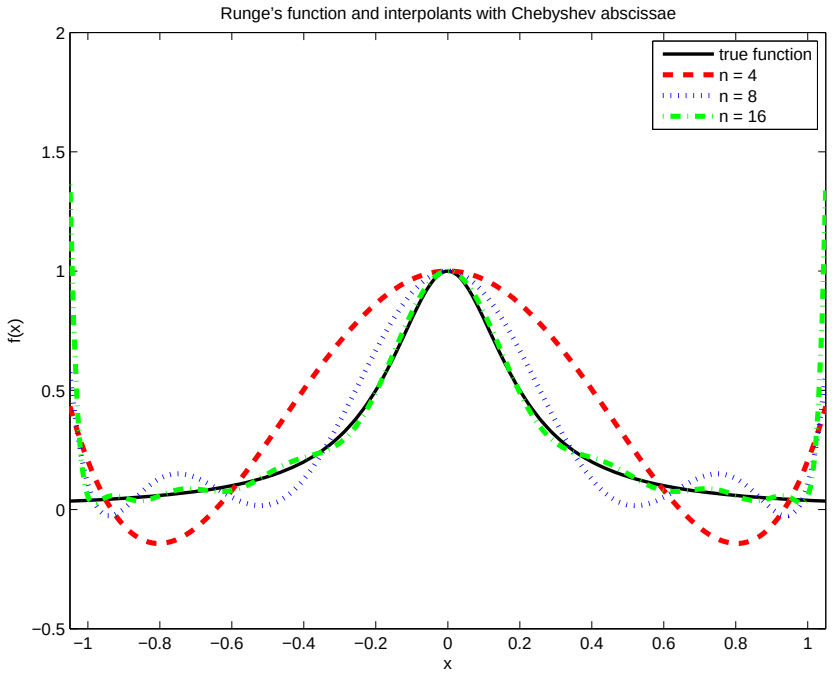
$$|e_n(\tilde{x})| \leq \max_{\xi \in [-1,+1]} \frac{|f^{(n+1)}(\xi)|}{2^n (n+1)!}$$

Example of Interpolation with Chebyshev Points

Consider Runge's function $f(x) = (1 + 25x^2)^{-1}$.



Interpolated with equally spaced abscissae



Interpolated with the Chebyshev abscissae

Despite the rapid growth in the magnitude of the derivatives of Runge's function, the error of the interpolants using the Chebyshev points is well controlled.