Lecture 15

Interpolation

Owen L. Lewis

Department of Mathematics and Statistics
University of New Mexico

Oct. 17, 2024

Course Notes

- Homework 8 will be assigned on Friday Oct 18. It will be due as normal on Friday Oct 25.

And now for something completely different. . .

Next Topics

1. Interpolation: Approximating a function $f(x)$ by a polynomial $p_n(x)$.
2. Least Squares: More linear algebra!
3. Differentiation: Approximating the derivative of a function $f(x)$.
4. Integration: Approximating an integral $\int_a^b f(x)\,dx$

Interpolation: Introduction

Objective

Approximate an unknown function $f(x)$ by an “easier” function $g(x)$ (perhaps a polynomial?).

Objective (alt)

Approximate some data by a function $g(x)$.

Types of approximating functions:

1. Polynomials
2. Piecewise polynomials
3. Rational functions
4. Trig functions
5. Others (inverse, exponential, Bessel, etc)

Interpolation: Introduction

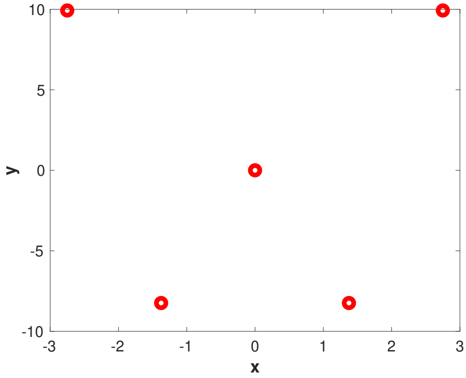
How do we approximate $f(x)$ by $g(x)$? In what sense is the approximation a good one?

1. Interpolation: $g(x)$ must have the same values of $f(x)$ at set of given points.
2. Least-squares: $g(x)$ must deviate as little as possible from $f(x)$ in the sense of a 2-norm: minimize $\int_a^b |f(t) - g(t)|^2 \, dt$
3. Chebyshev: $g(x)$ must deviate as little as possible from $f(x)$ in the sense of the ∞-norm: minimize $\max_{t \in [a,b]} |f(t) - g(t)|$.

Interpolation: Introduction

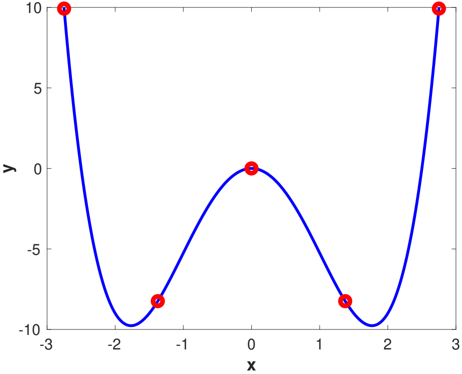
Given $n+1$ distinct points $x_0, \ldots, x_n$, and values $y_0, \ldots, y_n$, find a polynomial $p(x)$ of degree n so that

$$p(x_i) = y_i \quad i = 0, \ldots, n$$



Given $n+1$ distinct points $x_0, \ldots, x_n$, and values $y_0, \ldots, y_n$, find a polynomial $p(x)$ of degree n so that

$$p(x_i) = y_i \quad i = 0, \ldots, n$$



Interpolation: Introduction

Given $n+1$ distinct points $x_0, \ldots, x_n$, and values $y_0, \ldots, y_n$, find a polynomial $p(x)$ of degree n so that

$$p(x_i) = y_i \quad i = 0, \ldots, n$$

- A polynomial of degree n has $n+1$ degrees-of-freedom:

$$p(x) = a_0 + a_1 x + \cdots + a_n x^n$$

- $n+1$ constraints determine the polynomial uniquely:

$$p(x_i) = y_i, \quad i = 0, \ldots, n$$

Theorem (page 142 1stEd)

If points $x_0, \ldots, x_n$ are distinct, then for arbitrary $y_0, \ldots, y_n$, there is a *unique* polynomial $p(x)$ of degree at most n such that $p(x_i) = y_i$ for $i = 0, \ldots, n$.

How can you prove the polynomial is unique? (Hint: What if it isn't?)

Monomials

Obvious attempt: try picking

$$p(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

So for each x_i we have

$$p(x_i) = a_0 + a_1 x_i + a_2 x_i^2 + \cdots + a_n x_i^n = y_i$$

OR

$$\begin{aligned}
a_0 + a_1 x_0 + a_2 x_0^2 + \cdots + a_n x_0^n &= y_0 \\
a_0 + a_1 x_1 + a_2 x_1^2 + \cdots + a_n x_1^n &= y_1 \\
a_0 + a_1 x_2 + a_2 x_2^2 + \cdots + a_n x_2^n &= y_2 \\
a_0 + a_1 x_3 + a_2 x_3^2 + \cdots + a_n x_3^n &= y_3 \\
&\vdots \\
a_0 + a_1 x_n + a_2 x_n^2 + \cdots + a_n x_n^n &= y_n
\end{aligned}$$

Monomial: The problem

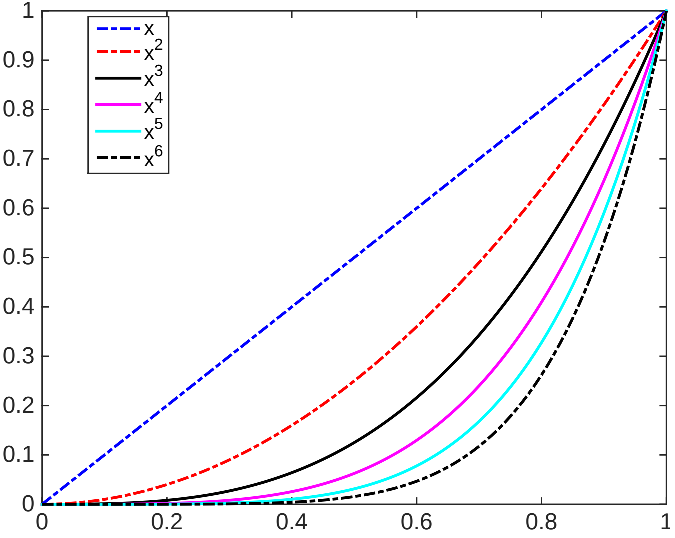
$$\begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ 1 & x_2 & x_2^2 & \dots & x_2^n \\ & & & \vdots & \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

The matrix above is known as the Vandermonde matrix.

Question

- Is this a "good" system to solve?

Ill-Conditioning!



Monomial: The problem

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ 1 & x_2 & x_2^2 & \dots & x_2^n \\ & & & \vdots & \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

As n gets large, the last column $\to [0, 0, \dots 0]^T$.
The matrix gets very close to singular: $cond(A) \to \infty$!

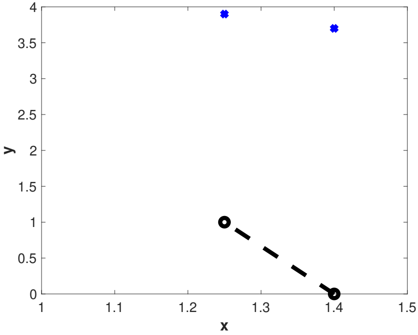
Back to the basics...

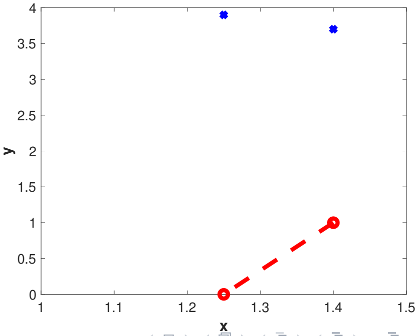
Example

Find the interpolating polynomial of least degree that interpolates

x	1.25	1.4
y	3.9	3.7

Graphically:





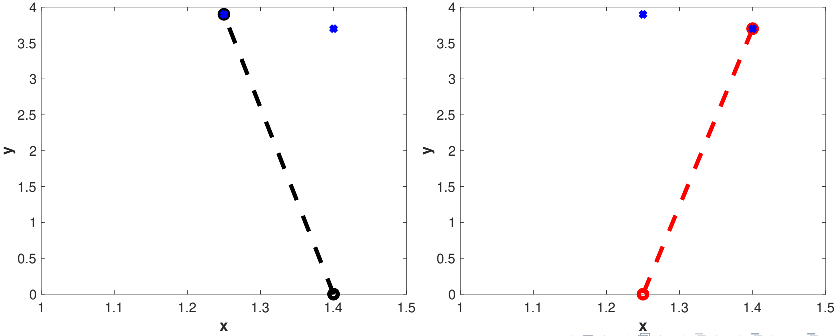
Back to the basics...

Example

Find the interpolating polynomial of least degree that interpolates

x	1.25	1.4
y	3.9	3.7

Graphically:



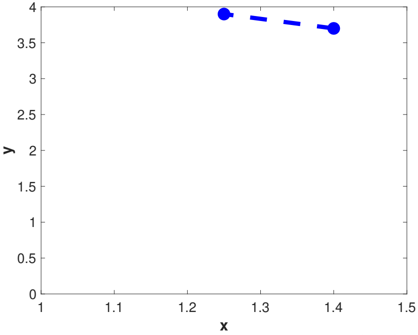
Back to the basics...

Example

Find the interpolating polynomial of least degree that interpolates

x	1.25	1.4
y	3.9	3.7

Graphically:



Back to the basics...

Example

Find the interpolating polynomial of least degree that interpolates

x	1.25	1.4
y	3.9	3.7

Directly

$$
\begin{aligned}
p(x) &= \left(\frac{x-1.4}{1.25-1.4}\right)3.9 + \left(\frac{x-1.25}{1.4-1.25}\right)3.7 \\
&= 3.7 + \left(\frac{3.9-3.7}{1.25-1.4}\right)(x-1.4) \\
&= 3.7 - \frac{4}{3}(x-1.4)
\end{aligned}
$$

Lagrange

What have we done? We've written $p(x)$ as

$$p(x) = \left(\frac{x - x_1}{x_0 - x_1} \right) y_0 + \left(\frac{x - x_0}{x_1 - x_0} \right) y_1$$

- the sum of two linear polynomials
- the first is zero at x_1 and 1 at x_0
- the second is zero at x_0 and 1 at x_1
- these are the two linear Lagrange basis functions:

$$\ell_0(x) = \frac{x - x_1}{x_0 - x_1} \qquad \ell_1(x) = \frac{x - x_0}{x_1 - x_0}$$

Notes

Lagrange

Example

Write the Lagrange basis functions for

x	$\frac{1}{3}$	$\frac{1}{4}$	1
y	2	-1	7

Directly

$$\ell_0(x) = \frac{(x-\frac{1}{4})(x-1)}{(\frac{1}{3}-\frac{1}{4})(\frac{1}{3}-1)}$$

$$\ell_1(x) = \frac{(x-\frac{1}{3})(x-1)}{(\frac{1}{4}-\frac{1}{3})(\frac{1}{4}-1)}$$

$$\ell_2(x) = \frac{(x-\frac{1}{3})(x-\frac{1}{4})}{(1-\frac{1}{3})(1-\frac{1}{4})}$$

Lagrange

The general Lagrange form is

$$\ell_k(x) = \prod_{i=0, i\neq k}^{n} \frac{x - x_i}{x_k - x_i}$$

The resulting interpolating polynomial is

$$p(x) = \sum_{k=0}^{n} \ell_k(x) y_k$$

Example

Find the equation of the parabola passing through the points (1,6), (-1,0), and (2,12)

$x_0 = 1, x_1 = -1, x_2 = 2; \qquad y_0 = 6, y_1 = 0, y_2 = 12;$

$$
\begin{aligned}
\ell_0(x) &= \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} &= \frac{(x+1)(x-2)}{(2)(-1)} \\
\ell_1(x) &= \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} &= \frac{(x-1)(x-2)}{(-2)(-3)} \\
\ell_2(x) &= \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} &= \frac{(x-1)(x+1)}{(1)(3)}
\end{aligned}
$$

$$
\begin{aligned}
p_2(x) &= y_0\ell_0(x) + y_1\ell_1(x) + y_2\ell_2(x) \\
&= -3 \times (x+1)(x-2) + 0 \times \frac{1}{6}(x-1)(x-2) \\
&\quad +4 \times (x-1)(x+1) \\
&= (x+1)[4(x-1) - 3(x-2)] \\
&= (x+1)(x+2)
\end{aligned}
$$

Summary so far:

- Monomials: $p(x) = a_0 + a_1 x + \cdots + a_n x^n$ results in poorly conditioned Vandermonde matrix that must be inverted.
- Monomials: but evaluating the Monomial interpolant is cheap (nested evaluation)

- Lagrange: $p(x) = \ell_0(x) y_0 + \cdots + \ell_n(x) y_n$ is very well behaved, in that it always (almost) exactly interpolates the (x_i, y_i).

  This is in contrast to the conditioning problems of monomials. Here, for large n, the ill-conditioning of the Vandermonde matrix leads to interpolants that do not interpolate the (x_i, y_i) well.

- When adding an additional data point, you cannot re-use previous polynomial easily (for monomials and Lagrange)

Lurking in the background: BASIS!

- In both cases what we have done is chosen a *basis* for our polynomial:

$$b_0(x), b_1(x), \ldots b_n(x).$$

  At this point, nothing depends on the y_k's.

- The interpolating polynomial will be created by linear combination:

$$p(x) = \sum_{k=0}^{n} a_k b_k(x)$$

- The trick is to choose the weights

$$a_0, a_1, \ldots a_n$$

  in order to get

$$p(x) = \sum_{k=0}^{n} a_k b_k(x)$$

  to match the y_k's.

Lurking in the background: BASIS!

- For "monomials" our basis is simple:

$$b_0(x) = 1, \quad b_1(x) = x, \ldots \quad b_n(x) = x^n.$$

  But this leads to problems when we try to find the a_k's.
- For Lagrange, our basis is more complex:

$$b_k(x) = \ell_k(x) = \prod_{i=0, i\neq k}^{n} \frac{x - x_i}{x_k - x_i},$$

  but finding the weights is trivial!

$$a_k(x) = y_k.$$

Notes

Recall Nested Form

Given a polynomial

$$p(x) = -5 + 4x - 7x^2 + 2x^3 + 3x^4$$

we can write this as

$$p(x) = -5 + x(4 + x(-7 + x(2 + 3x)));$$

evaluation can be done from the inside-out, for cheap (nested evaluation).

This polynomial can also be written as

$$p(x) = -5 + 2x - 4x(x-1) + 8x(x-1)(x+1) + 3x(x-1)(x+1)(x-2)$$

in nested form

$$p(x) = -5 + x(2 + (x-1)(-4 + (x+1)(8 + 3(x-2))))$$

Newton Polynomials

- Newton Polynomials are of the form

$$p_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + a_3(x - x_0)(x - x_1)(x - x_2) + \dots$$

- The basis used is thus

function	order
1	0
$x - x_0$	1
$(x - x_0)(x - x_1)$	2
$(x - x_0)(x - x_1)(x - x_2)$	3

- More stable that monomials
- Almost as computationally efficient (nested evaluation)
- Easier to add more data points

Newton Polynomials using Divided Differences

Consider the data

x_0	x_1	x_2
y_0	y_1	y_2

We want to find a_0, a_1, and a_2 in the following polynomial so that it fits the data:

$$p_2(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1)$$

Matching the data gives three equations to determine our three unknowns a_i:

$$\begin{aligned}
\text{at } x_0\text{: } y_0 &= a_0 + 0 + 0 \\
\text{at } x_1\text{: } y_1 &= a_0 + a_1(x_1 - x_0) + 0 \\
\text{at } x_2\text{: } y_2 &= a_0 + a_1(x_2 - x_0) + a_2(x_2 - x_0)(x_2 - x_1)
\end{aligned}$$

Newton Polynomials using Divided Differences

Or in matrix form:

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & x_1 - x_0 & 0 \\ 1 & x_2 - x_0 & (x_2 - x_0)(x_2 - x_1) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix}$$

$\Rightarrow$ lower triangular
$\Rightarrow$ only $\mathcal{O}(n^2)$ operations

Question

How many operations are needed to find the coefficients in the monomial basis?

Newton Polynomials using Divided Differences

Using Forward Substitution to solve this lower triangular system yields:

$$
\begin{aligned}
a_0 &= y_0 = f(x_0) \\
a_1 &= \frac{y_1 - a_0}{x_1 - x_0} \\
&= \frac{f(x_1) - f(x_0)}{x_1 - x_0} \\
a_2 &= \frac{y_2 - a_0 - (x_2 - x_0)a_1}{(x_2 - x_1)(x_2 - x_0)} \\
&= \frac{f(x_2) - f(x_0) - (x_2 - x_0)\frac{f(x_1)-f(x_0)}{x_1-x_0}}{(x_2 - x_1)(x_2 - x_0)} \\
&= \frac{\frac{f(x_2)-f(x_1)}{x_2-x_1} - \frac{f(x_1)-f(x_0)}{x_1-x_0}}{x_2 - x_0}
\end{aligned}
$$

Newton Polynomials using Divided Differences

From this we see a pattern. There are many terms of the form

$$\frac{f(x_j) - f(x_i)}{x_j - x_i}$$

These are called *divided differences* and are denoted with square brackets:

$$f[x_i, x_j] = \frac{f(x_j) - f(x_i)}{x_j - x_i}$$

Applying this to our results:

$$\begin{aligned} a_0 &= f(x_0) \\ a_1 &= f[x_0, x_1] \\ a_2 &= \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0} \\ &= f[x_0, x_1, x_2] \end{aligned}$$

Newton Polynomials using Divided Differences

example: long way

Example

For the data

x	1	-4	0
y	3	13	-23

Find the 2nd order interpolating polynomial using Newton.

We know

$$p_2(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1)$$

And that

$$\begin{aligned}
a_0 &= f[x_0] = f[1] = f(1) = 3 \\
a_1 &= f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0} = \frac{13 - 3}{-4 - 1} = -2 \\
a_2 &= f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0} \\
&= \frac{\frac{-23-13}{0--4} - \frac{13-3}{-4-1}}{0 - 1} \\
&= \frac{-9 + 2}{-1} = 7
\end{aligned}$$

So

$$p_2(x) = 3 - 2(x - 1) + 7(x - 1)(x + 4)$$

Divided Differences

Recursive Property

$$f[x_0, \ldots, x_k] = \frac{f[x_1, \ldots, x_k] - f[x_0, \ldots, x_{k-1}]}{x_k - x_0}$$

With the first two defined by

$$f[x_i] = f(x_i)$$

$$f[x_i, x_j] = \frac{f[x_j] - f[x_i]}{x_j - x_i}$$

Divided Differences

Invariance Theorem

$f[x_0, \dots, x_k]$ is invariant under all permutations of the arguments $x_0, \dots, x_k$

Simple "proof": $f[x_0, x_1, ..., x_k]$ is the coefficient of the x^k term in the polynomial interpolating f at $x_0, \dots, x_k$. But any permutation of the x_i still gives the same polynomial. That is, the order that you consider the interpolation points does not matter.

This says that we can also write

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

Divided Differences

the easy way: tables

We can compute the divided differences much easier using tables. To construct the divided difference table for $f(x)$ for the $x_0, \ldots, x_3$

x	$f[\cdot]$	$f[\cdot,\cdot]$	$f[\cdot,\cdot,\cdot]$	$f[\cdot,\cdot,\cdot,\cdot]$
x_0	$f[x_0]$			
		$f[x_0, x_1]$		
x_1	$f[x_1]$		$f[x_0, x_1, x_2]$	
		$f[x_1, x_2]$		$f[x_0, x_1, x_2, x_3]$
x_2	$f[x_2]$		$f[x_1, x_2, x_3]$	
		$f[x_2, x_3]$		
x_3	$f[x_3]$			

Divided Differences

the easy way: tables

We can compute the divided differences much easier using tables. To construct the divided difference table for $f(x)$ for the $x_0, \ldots, x_3$

x	$f[\cdot]$	$f[\cdot,\cdot]$	$f[\cdot,\cdot,\cdot]$	$f[\cdot,\cdot,\cdot,\cdot]$
x_0	$f[x_0]$			
		$f[x_0, x_1]$		
x_1	$f[x_1]$		$f[x_0, x_1, x_2]$	
		$f[x_1, x_2]$		$f[x_0, x_1, x_2, x_3]$
x_2	$f[x_2]$		$f[x_1, x_2, x_3]$	
		$f[x_2, x_3]$		
x_3	$f[x_3]$			

Divided Differences

the easy way: tables

We can compute the divided differences much easier using tables. To construct the divided difference table for $f(x)$ for the $x_0, \ldots, x_3$

x	$f[\cdot]$	$f[\cdot,\cdot]$	$f[\cdot,\cdot,\cdot]$	$f[\cdot,\cdot,\cdot,\cdot]$
x_0	$f[x_0]$			
		$f[x_0, x_1]$		
x_1	$f[x_1]$		$f[x_0, x_1, x_2]$	
		$f[x_1, x_2]$		$f[x_0, x_1, x_2, x_3]$
x_2	$f[x_2]$		$f[x_1, x_2, x_3]$	
		$f[x_2, x_3]$		
x_3	$f[x_3]$			

Divided Differences

the easy way: tables

We can compute the divided differences much easier using tables. To construct the divided difference table for $f(x)$ for the $x_0, \ldots, x_3$

x	$f[\cdot]$	$f[\cdot,\cdot]$	$f[\cdot,\cdot,\cdot]$	$f[\cdot,\cdot,\cdot,\cdot]$
x_0	$f[x_0]$			
		$f[x_0,x_1]$		
x_1	$f[x_1]$		$f[x_0,x_1,x_2]$	
		$f[x_1,x_2]$		$f[x_0,x_1,x_2,x_3]$
x_2	$f[x_2]$		$f[x_1,x_2,x_3]$	
		$f[x_2,x_3]$		
x_3	$f[x_3]$			

Divided Differences

the easy way: tables

We can compute the divided differences much easier using tables. To construct the divided difference table for $f(x)$ for the $x_0, \ldots, x_3$

x	$f[\cdot]$	$f[\cdot,\cdot]$	$f[\cdot,\cdot,\cdot]$	$f[\cdot,\cdot,\cdot,\cdot]$
x_0	$f[x_0]$			
		$f[x_0, x_1]$		
x_1	$f[x_1]$		$f[x_0, x_1, x_2]$	
		$f[x_1, x_2]$		$f[x_0, x_1, x_2, x_3]$
x_2	$f[x_2]$		$f[x_1, x_2, x_3]$	
		$f[x_2, x_3]$		
x_3	$f[x_3]$			

Divided Differences

the easy way: tables

We can compute the divided differences much easier using tables. To construct the divided difference table for $f(x)$ for the $x_0, \ldots, x_3$

x	$f[\cdot]$	$f[\cdot,\cdot]$	$f[\cdot,\cdot,\cdot]$	$f[\cdot,\cdot,\cdot,\cdot]$
x_0	$f[x_0]$			
		$f[x_0, x_1]$		
x_1	$f[x_1]$		$f[x_0, x_1, x_2]$	
		$f[x_1, x_2]$		$f[x_0, x_1, x_2, x_3]$
x_2	$f[x_2]$		$f[x_1, x_2, x_3]$	
		$f[x_2, x_3]$		
x_3	$f[x_3]$			

Divided Differences

the easy way: tables

We can compute the divided differences much easier using tables. To construct the divided difference table for $f(x)$ for the $x_0, \ldots, x_3$

x	$f[\cdot]$	$f[\cdot,\cdot]$	$f[\cdot,\cdot,\cdot]$	$f[\cdot,\cdot,\cdot,\cdot]$
x_0	$f[x_0]$			
		$f[x_0, x_1]$		
x_1	$f[x_1]$		$f[x_0, x_1, x_2]$	
		$f[x_1, x_2]$		$f[x_0, x_1, x_2, x_3]$
x_2	$f[x_2]$		$f[x_1, x_2, x_3]$	
		$f[x_2, x_3]$		
x_3	$f[x_3]$			