Set 5: Polynomial Interpolation – Part 2

Kyle A. Gallivan

Department of Mathematics

Florida State University

Foundations of Computational Math 1

Fall 2017

Overview

- Complexity measured in terms of number of computations, i.e., sequential computation
- Evaluation of a polynomial in mononmial form
- Interpolation polynomials
- For each form consider:
  - Complexity of constructing the polynomial, i.e., the parameters
  - Complexity of evaluating the polynomial at a point $x \neq x_i$
  - Complexity of updating the polynomial to include a new point

Horner's Rule

Assume the polynomial, $p_n(x)$, is given in terms of monomials, x^i

$$p_n(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \cdots + \alpha_n x^n$$

- $O(n)$ evaluation by rewriting: For example, let $n = 4$

$$p_4(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \alpha_3 x^3 + \alpha_4 x^4$$

$$p_4(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + x^3(\alpha_3 + x\alpha_4)$$

$$p_4(x) = \alpha_0 + \alpha_1 x + x^2(\alpha_2 + x(\alpha_3 + x\alpha_4))$$

$$p_4(x) = \alpha_0 + x(\alpha_1 + x(\alpha_2 + x(\alpha_3 + (x\alpha_4))))$$

- Adaptable to other similar forms, e.g., Newton form.
- Repeated application leads to an algorithm for the evaluation of derivatives at x.

Lagrange Form

- Storing only $x_i,\ y_i$, i.e., no preprocessing, yields an $O(n^2)$ to evaluate $p_n(x)$ via repeated linear interpolation, e.g., Aitken's method and Neville's method.
- The complexity of Lagrange form depends on form used
  - Barycentric form 1 exploits basic common structure of Lagrange characteristic/basis functions
  - Barycentric form 2 rescales form 1 and can result in significant reduction in computations

Lagrange Form

Recall, the standard form is sum of n-degree polynomials (using slightly modified notation)

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

$$\ell_i^{(n)}(x) = \frac{m_i^{(n)}(x)}{m_i^{(n)}(x_i)}$$

$$p_n(x) = \sum_{i=0}^{n} y_i \; \ell_i^{(n)}(x)$$

Lagrange Form

Lemma. *Given* (x_i, y_i) *for* $0 \leq i \leq n$ *and defining*

$$m_i^{(n)}(x) = \prod_{j=0, j\neq i}^{n} (x - x_j) \quad \textit{and} \quad \omega_j(x) = \prod_{i=0}^{j-1} (x - x_i)$$

we have

$$m_i^{(n)}(x) = \frac{\omega_{n+1}(x)}{(x - x_i)} \quad \textit{and} \quad m_i^{(n)}(x_i) = \omega'_{n+1}(x_i)$$

and the Lagrange characteristic functions can be written

$$\ell_i(x) = \frac{\omega_{n+1}(x)}{(x - x_i)\omega\prime_{n+1}(x_i)}.$$

Lagrange Form

Proof. Let $i = n$. We have

$$\begin{aligned}
\omega_{n+1}(x) &= \omega_n(x)(x - x_n) \\
&\to \omega'_{n+1}(x) = \omega'_n(x)(x - x_n) + \omega_n(x) \\
&\to \omega'_{n+1}(x_n) = \omega_n(x_n) \\
\therefore \ell_n(x) &= \frac{\omega_n(x)}{\omega_n(x_n)} = \frac{\omega_{n+1}(x)}{(x - x_n)\omega\prime_{n+1}(x_n)}.
\end{aligned}$$

This adapts trivially to $\ell_i(x)$ for $i \neq n$. □

Barycentric Form 1

Definition 5.1. (Berrut and Trefethen, Siam Review Vol. 46 No. 3)

Given (x_i, y_i) for $0 \leq i \leq n$, the Barycentric interpolation formula form 1 is

$$p_n(x) = \omega_{n+1}(x) \sum_{i=0}^{n} \frac{y_i}{(x - x_i)\omega'_{n+1}(x_i)} = \omega_{n+1}(x) \sum_{i=0}^{n} y_i \frac{\gamma_i}{(x - x_i)}$$

where

$$\gamma_i = 1/\omega'_{n+1}(x_i) \quad \text{and} \quad \omega_{n+1}(x) = \prod_{i=0}^{n} (x - x_i)$$

Barycentric Form 1

- Construction of $p_n(x)$ requires the computation of γ_i^{-1}, for $0 \leq i \leq n$ where

$$\gamma_i^{-1} = \omega'_{n+1}(x_i) = m_i^{(n)}(x_i) = \prod_{j=0, j\neq i}^{n} (x_i - x_j)$$

- Construction of $p_n(x)$ **does not depend on** y_i.
- Construction of $p_n(x)$ can be done in $O(n^2)$ computations.
- Structure can be exploited to keep the constant small.

Lagrange Complexity

Let $n = 4$

$$m_0^{(4)} = [(x_0 - x_1)(x_0 - x_2)(x_0 - x_3)(x_0 - x_4)]_R$$

$$m_1^{(4)} = [(x_1 - x_0)]_L \, [(x_1 - x_2)(x_1 - x_3)(x_1 - x_4)]_R$$

$$m_2^{(4)} = [(x_2 - x_0)(x_2 - x_1)]_L \, [(x_2 - x_3)(x_2 - x_4)]_R$$

$$m_3^{(4)} = [(x_3 - x_0)(x_3 - x_1)(x_3 - x_2)]_L \, [(x_3 - x_4)]_R$$

$$m_4^{(4)} = [(x_4 - x_0)(x_4 - x_1)(x_4 - x_2)(x_4 - x_3)]_L$$

Note. The terms in $[--]_L$ appear earlier with sign changes in $[--]_R$ terms.

Lagrange Complexity

Let $n = 4$ and update to $n = 5$

$$m_0^{(5)} = [(x_0 - x_1)(x_0 - x_2)(x_0 - x_3)(x_0 - x_4)]\,(x_0 - x_5)$$
$$m_1^{(5)} = [(x_1 - x_0)(x_1 - x_2)(x_1 - x_3)(x_1 - x_4)]\,(x_1 - x_5)$$
$$m_2^{(5)} = [(x_2 - x_0)(x_2 - x_1)(x_2 - x_3)(x_2 - x_4)]\,(x_2 - x_5)$$
$$m_3^{(5)} = [(x_3 - x_0)(x_3 - x_1)(x_3 - x_2)(x_3 - x_4)]\,(x_3 - x_5)$$
$$m_4^{(5)} = [(x_4 - x_0)(x_4 - x_1)(x_4 - x_2)(x_4 - x_3)]\,(x_4 - x_5)$$
$$m_5^{(5)} = [(x_5 - x_0)(x_5 - x_1)(x_5 - x_2)(x_5 - x_3)(x_5 - x_4)]$$

- The terms in brackets in the $m_i^{(5)}$, $0 \leq i \leq 4$ are the $m_i^{(4)}$ (known).
- The terms in $m_5^{(5)}$ are the terms applied to the earlier $m_i^{(4)}$ to get $m_i^{(5)}$ with sign changes.

Lagrange Complexity

We have $O(n)$ computations to update the parameters for $p_n(x)$ to those for $p_{n+1}(x)$ via the algorithm:

$p = 1$
for $i = 0, \ldots, n-1$
$\quad t = (x_i - x_n)$
$\quad m_i^{(n)} = t \times m_i^{(n-1)}$
$\quad p = -t \times p$
end
$m_n^{(n)} = p$

Barycentric Form 1

- Construction of $p_n(x)$ is $O(n^2)$: start with $m_0^{(1)} = x_0 - x_1$ and $m_1^{(1)} = x_1 - x_0$ and apply incremental algorithm.
- Construction of $p_n(x)$ **does not depend on** y_i.
- Update of $p_n(x)$ to $p_{n+1}(x)$ is $O(n)$
- Evaluation of $p_n(x)$ requires
  - $O(n)$ computations to evaluate $\omega_{n+1}(x)$
  - $O(n)$ computations to evaluate $\sum_{i=0}^{n}(y_i\gamma_i)(\omega_{n+1}(x)/(x - x_i))$
  - overflow? numerical stability for $x \approx x_i$?

Barycentric Form 2

Lemma. *If*

$$\ell_i(x) = \frac{\omega_{n+1}(x)}{(x - x_i)\omega'_{n+1}(x_i)}$$

then

$$\sum_{i=0}^{n} \ell_i(x) = \omega_{n+1}(x) \sum_{i=0}^{n} \frac{\gamma_i}{(x - x_i)} = 1$$

$$\omega_{n+1}(x) = \left[\sum_{i=0}^{n} \frac{\gamma_i}{(x - x_i)} \right]^{-1}$$

Barycentric Form 2

Definition 5.2. (Berrut and Trefethen, Siam Review Vol. 46 No. 3)

Given (x_i, y_i) for $0 \leq i \leq n$ and defining

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

the Barycentric interpolation formula form 2 or the true form is

$$p_n(x) = \{\sum_{i=0}^{n} y_i \frac{\gamma_i}{(x - x_i)}\} / \{\sum_{i=0}^{n} \frac{\gamma_i}{(x - x_i)}\}$$

where $\gamma_i = 1/\omega'_{n+1}(x_i)$.

Barycentric Form 2

- The term 'barycentric form' in the literature usually refers to Barycentric Form 2.
- Construction of $p_n(x)$ **does not depend on** y_i.
- Update of $p_n(x)$ to $p_{n+1}(x)$ is $O(n)$.
- Evaluation of $p_n(x)$ requires $O(n)$ computations.
- Main advantage: common factors in the γ_i can be cancelled!
- γ_i replaced by β_i which are often simpler and better scaled (although the scale may still be bad)
- Construction of $p_n(x)$ **may be** $O(n)$.

Barycentric Form 2

Consider the interval $[a, b]$ with equally spaced interpolation points $x_i = a + ih$ where $h = (b - a)/n$.

$$\gamma_i^{-1} = \left[\prod_{j=0}^{i-1}(x_i - x_j)\right]\left[\prod_{j=i+1}^{n}(x_i - x_j)\right]$$

$$= \left[\prod_{j=0}^{i-1}(a + ih - a - jh)\right]\left[\prod_{j=i+1}^{n}(a + ih - a - jh)\right]$$

$$\left[h^i \prod_{j=0}^{i-1}(i - j)\right]\left[h^{n-i}\prod_{j=i+1}^{n}(i - j)\right] = h^n\left[\prod_{j=1}^{i}(j)\right]\left[(-1)^{n-i}\prod_{j=1}^{n-i}(j)\right]$$

$$= (-1)^{n-i}h^n(i!)(n - i)!$$

Barycentric Form 2

$$\gamma_i = \frac{(-1)^{n-i}}{h^n (i!)(n-i)!} = \frac{(-1)^{n-i}}{h^n n!} \binom{n}{i}$$

$$= \frac{(-1)^n}{h^n n!} \left[(-1)^i \binom{n}{i} \right] = \frac{(-1)^n}{h^n n!} \beta_i$$

$$\beta_{i+1} = -\beta_i \frac{n-i}{i+1}$$

Barycentric Form

- γ_i can be replaced by β_i in the barycentric formula via cancellation of common factor.
- For equidistant points, β_i produced recursively implies construction of $p_n(x)$ requires $O(n)$ computations.
- Note the large range in magnitude of γ_i and β_i for equally spaced nodes (approximately 2^n)
- Reflects ill-conditioning of interpolation on equally spaced nodes
- on $[-1, 1]$ nonuniform points clustered at the endpoints with a density proportional to $1/\sqrt{1-x^2}$ as $n \to \infty$ are required to make the weights comparable in scale, i.e., they do not vary by an exponential in n. The Chebyshev points are a good example.

Barycentric Form and Chebyshev Points

Lemma. *If the nodes for interpolation are Chebyshev points of the first kind given by*

$$x_j = \cos \frac{(2j+1)\pi}{2n+2} \quad 0 \le j \le n$$

then the barycentric coefficients are

$$\beta_j = (-1)^j \sin \frac{(2j+1)\pi}{2n+2} \quad 0 \le j \le n$$

Barycentric Form and Chebyshev Points

Lemma. *If the nodes for interpolation are Chebyshev points of the second kind given by*

$$x_j = \cos \frac{j\pi}{n} \quad 0 \le j \le n$$

then the barycentric coefficients are

$$\beta_j = (-1)^j \delta_j, \quad \delta_j = \begin{cases} 1/2 & \text{if } j = 0 \text{ or } j = n \\ 1 & \text{otherwise} \end{cases}$$

Barycentric Form and Chebyshev Points

- Uniform and Chebyshev points of first, second, third, and fourth kind all have simple formulas for barycentric weights.
- Construction of $p_n(x)$ requires $O(n)$ computations in all of these cases.
- Construction of $p_n(x)$ **does not depend on** y_i therefore reusable for other functions.
- Update and evaluation of $p_n(x)$ requires $O(n)$ computations in all of these cases.
- Higham (IMA Journal of Numerical Analysis, Volume 24, 2004) has shown the very satisfactory stability properties of the barycentric form. (e.g., overflow and numerical stability when $x \approx x_i$?)
- Popular for numerical methods for solving PDEs

Barycentric Form Example

A simple MATLAB code (From Berrut and Trefethen, Siam Review Vol. 46 No. 3) using Chebyshev points of the second kind applied to the problem

$$f(x) = |x| + \frac{1}{2}x - x^2$$

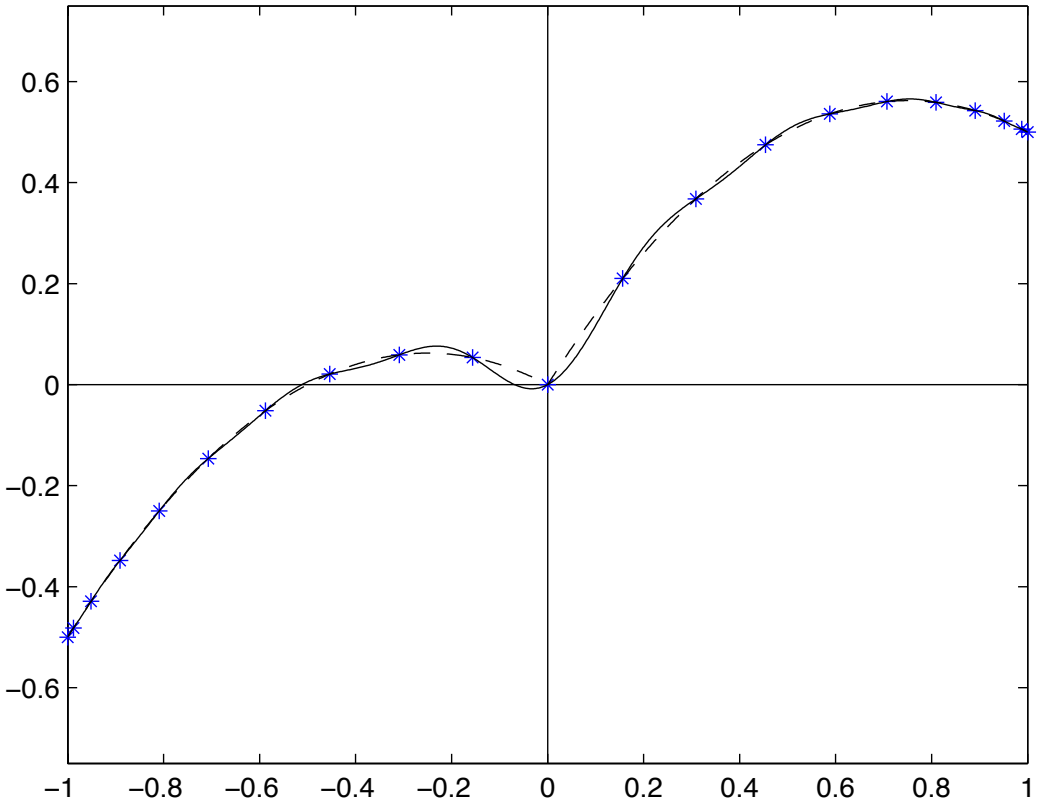
$$-1 \leq x \leq 1$$

$$x_k = \cos(k\pi/n), \quad 0 \leq k \leq n$$

- $n = 20$
- increasing n improves approximation

```
fun = @(z)  abs(z)+0.5*z-z.^2;
n=20;
% Chebyshev of the second kind Barycentric coefficients
c= [0.5; ones(n-1,1); 0.5].*(-1).^((0:n)');
% evaluate  f at the Chebyshev points
x = cos(pi*(0:n)'/n);   xx=linspace(-1,1,1000)';
f = fun(x);    ft=fun(xx);
numer=zeros(size(xx));    denom=zeros(size(xx));
for j=1:n+1
   xdiff=xx-x(j);
   temp=c(j)./xdiff;
   numer = numer+ temp*f(j);
   denom = denom+ temp;
end
ff=numer./denom;
plot(x,f,'*')  % interpolation points
plot(xx,ff,'-k') % interpolating polynomial
plot(xx,ft,'--k') % f(x)
```

Barycentric Example



$f(x)$: dotted line, $p(x)$: solid line, $*$: interpolation points.

Newton Complexity

With $\omega_{i+1}(x) = (x - x_0) \cdots (x - x_i)$, we have

$$p_n(x) = \sum_{i=0}^{n} y[x_0, \ldots, x_i]\omega_i(x)$$

$$D^k y_i = y[x_i, \ldots, x_{i+k}] = \frac{y[x_{i+1}, \ldots, x_{i+k}] - y[x_i, \ldots, x_{i+k-1}]}{x_{i+k} - x_i}$$

- Divided difference table requires $\frac{n(n+1)}{2}$ values
  $D^k y_i \;\; 1 \leq k \leq n, \;\; 0 \leq i \leq n - k$
- $O(n^2)$ computations to construct the table.
- adding (x_{n+1}, y_{n+1}) adds a column of $n + 1$ entries and requires $O(n)$ computations

Newton Complexity

- Evaluation of $p_n(x) = \sum_{i=0}^{n} y[x_0, \ldots, x_i]\omega_i(x)$ can be done in a nested fashion in $O(n)$.
- $p_n(x)$ can be specified in terms of multiple "paths" through the divided difference table, i.e., in terms of different sets of divided differences.
- order of points in divided difference does not matter

$$y[x_0, x_1] = \frac{(y_1 - y_0)}{(x_1 - x_0)} = \frac{-(y_1 - y_0)}{-(x_1 - x_0)} = \frac{(y_0 - y_1)}{(x_0 - x_1)} = y[x_1, x_0]$$

$$y[x_0, x_1, x_2] = y[x_1, x_0, x_2] = y[x_2, x_0, x_1]$$

Newton Form

i	0	1	2	3
x_i	0	1	3	4
y_i	-5	1	25^*	55
$y[-,-]$	6	12^*	30	
$y[-,-,-]$	2	6^*		
$y[-,-,-,-]$	1^*			

Newton Form

Create the quadratic polynomial $p_2(x)$ that interpolates (x_0, f_0),(x_1, f_1), and (x_2, f_2) using information from the left side of table.

$$\begin{aligned} p_2(x) &= y_0 + (x - x_0)y[x_0, x_1] + (x - x_0)(x - x_1)y[x_0, x_1, x_2] \\ &= -5 + 6x + 2x(x - 1) \\ &= -5 + 4x + 2x^2 \end{aligned}$$

Other forms possible – all equivalent by uniqueness.

Newton Form

Create the cubic polynomial $p_3(x)$ that interpolates (x_i, f_i), $0 \leq i \leq 3$.

Use information on left side of table:

$$\begin{aligned} p_3(x) &= y_0 + (x - x_0)y[x_0, x_1] + (x - x_0)(x - x_1)y[x_0, x_1, x_2] \\ &\quad + (x - x_0)(x - x_1)(x - x_2)y[x_0, x_1, x_2, x_3] \\ &= -5 + 6x + 2x(x - 1) + x(x - 1)(x - 3) \\ &= -5 + 7x - 2x^2 + x^3 \end{aligned}$$

Newton Form

Create the cubic polynomial $r_3(x)$ that interpolates (x_i, f_i), $0 \le i \le 3$.

Use information on right side of table:

$$
\begin{aligned}
r_3(x) &= y_3 + (x - x_3)y[x_3, x_2] + (x - x_3)(x - x_2)y[x_3, x_2, x_1] \\
&\quad + (x - x_3)(x - x_2)(x - x_1)y[x_3, x_2, x_1, x_0] \\
&= y_3 + (x - x_3)y[x_2, x_3] + (x - x_3)(x - x_2)y[x_1, x_2, x_3] \\
&\quad + (x - x_3)(x - x_2)(x - x_1)y[x_0, x_1, x_2, x_3] \\
&= 55 + 30(x - 4) + 6(x - 4)(x - 3) + 1(x - 4)(x - 3)(x - 1) \\
&= -5 + 7x - 2x^2 + x^3
\end{aligned}
$$

Newton Form

Create the cubic polynomial $q_3(x)$ that interpolates (x_i, f_i), $0 \leq i \leq 3$.

Use information marked by $*$ in table:

$$
\begin{aligned}
q_3(x) &= y_2 + (x - x_2)y[x_2, x_1] + (x - x_2)(x - x_1)y[x_2, x_1, x_3] \\
&\quad + (x - x_2)(x - x_1)(x - x_3)y[x_2, x_1, x_3, x_0] \\
&= y_2 + (x - x_2)y[x_1, x_2] + (x - x_2)(x - x_1)y[x_1, x_2, x_3] \\
&\quad + (x - x_2)(x - x_1)(x - x_3)y[x_0, x_1, x_2, x_3] \\
&= 25 + 12(x - 3) + 6(x - 3)(x - 1) + (x - 3)(x - 1)(x - 4) \\
&= -5 + 7x - 2x^2 + x^3
\end{aligned}
$$

Some Useful Divided Difference Identities

$$\omega'_{n+1}(x_i) = \prod_{k=0, k\neq i}^{n} (x_i - x_k)$$

$$f[x_0, \ldots, x_n] = \sum_{i=0}^{n} \frac{f(x_i)}{\omega'_{n+1}(x_i)}$$

$f[x_0, \ldots, x_n] = f[x_{i_0}, \ldots, x_{i_n}]$ for any permutation of indices

$$f[x_0, \ldots, x_n, x] = \sum_{i=0}^{n} \frac{f[x, x_i]}{\omega'_{n+1}(x_i)}$$

Many others, including nondistinct points, see Isaason and Keller (1966) and text.

An Alternative Divided Difference Computation

- Smoktunowicz et al. (Computing V79, pp. 33-52, 2007) have proposed an algorithm for polynomial interpolation and evaluating the divided differences.
- It is based on a parameterized approach that for particular values of the parameters is equal to Aitken repeated linear interpolation and for another set the evaluation of the divided differences.
- The algorithm is based on the identity

$$f[x_0, \ldots, x_n] = \sum_{i=0}^{n} \frac{f(x_i)}{\omega'_{n+1}(x_i)}$$

that shows the divided differences and the coefficients for the Lagrange form are very much related. The algorithm is a modification of the Lagrange algorithm discussed earlier.

An Alternative Divided Difference Computation

Consider $n = 3$

$$\begin{pmatrix} f_0 \\ f[x_0, x_1] \\ f[x_0, x_1, x_2] \\ f[x_0, x_1, x_2, x_3] \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ [\omega_2'(x_0)]^{-1} & [\omega_2'(x_1)]^{-1} & 0 & 0 \\ [\omega_3'(x_0)]^{-1} & [\omega_3'(x_1)]^{-1} & [\omega_3'(x_2)]^{-1} & 0 \\ [\omega_4'(x_0)]^{-1} & [\omega_4'(x_1)]^{-1} & [\omega_4'(x_2)]^{-1} & [\omega_4'(x_3)]^{-1} \end{pmatrix} \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{pmatrix}$$

- rows contain Lagrange coefficients for $p_0(x)$ to $p_3(x)$.
- earlier Lagrange coefficient algorithm produces each row in turn
- compute each element of the matrix vector product result along with each row

Smoktunowicz et al. Divided Difference Algorithm

Recall the basic facts and organize the computations as with Lagrange coefficients.

$$\omega_{k+1}(x) = (x - x_0)(x - x_1)\dots(x - x_k), \quad k = 0, \dots, n$$

$$\omega'_{n+1}(x_i) = \omega'_n(x_i)(x_i - x_n), \quad i < n$$

$$f[x_0, \dots, x_n] = \sum_{i=0}^{n} \frac{f(x_i)}{\omega'_{n+1}(x_i)}$$

$$= \left\{ \sum_{i=0}^{n-1} \left[\frac{f(x_i)}{\omega'_n(x_i)} \right] \frac{1}{(x_i - x_n)} \right\} + \frac{f_n}{\omega'_{n+1}(x_n)}$$

$$\omega'_{n+1}(x_n) = (x_n - x_0)(x_n - x_1)\dots(x_n - x_{n-1})$$

$$= (-1)^n (x_0 - x_n)(x_1 - x_n)\dots(x_{n-1} - x_n)$$

Smoktunowicz et al. Divided Difference Algorithm

Assume $f[x_0, \ldots, x_{n-1}]$ and

$$\begin{pmatrix} d_0 \\ d_1 \\ \vdots \\ d_{n-2} \\ d_{n-1} \end{pmatrix} = \begin{pmatrix} f_0/\omega_n'(x_0) \\ f_1/\omega_n'(x_1) \\ \vdots \\ f_{n-2}/\omega_n'(x_{n-2}) \\ f_{n-1}/\omega_n'(x_{n-1}) \end{pmatrix}$$

are known. $f[x_0, \ldots, x_n]$ can be computed in $O(n)$ computations and space. Therefore, computing $f[x_0, x_1]$ to $f[x_0, \ldots, x_N]$ requires $O(N^2)$ computations and $O(N)$ space.

Smoktunowicz et al. Divided Difference Algorithm

$p = (x_0 - x_n)$
$d_0 = d_0/p$
$s = d_0$
for $i = 1, \ldots, n-1$
$\quad t = (x_i - x_n)$
$\quad d_i = d_i/t$
$\quad p = t \times p$
$\quad s = d_i + s$
end
$d_n = (-1)^n \ \ f_n/p$
$f[x_0, \ldots, x_n] = s + d_n$

Newton and Barycentric Lagrange Complexity

- Construction of $p_n(x)$ is $O(n^2)$ for Newton and Lagrange.
- Construction of $p_n(x)$ is $O(n)$ for some Lagrange.
- Construction of $p_n(x)$ for Newton **depends on** y_i.
- Construction of $p_n(x)$ for Lagrange **does not depend on** y_i.
- Lagrange parameters can be reused for other y_i values.
- Update of $p_n(x)$ to $p_{n+1}(x)$ is $O(n)$ for both.
- Evaluation of $p_n(x)$ is $O(n)$ for both.
- Stability of Lagrange independent of node order.
- Stability of Newton depends on node order.