(a)+(c) We present a divided-difference table for the given problem, laid out vertically. Computations were done in Matlab.

\[
\begin{array}{cccccccc}
0.3000 & 0.3000 & 0.3200 & 0.3200 & 0.3500 & 0.3500 & 0.3300 & 0.3300 \\
0.2955 & 0.2955 & 0.3146 & 0.3146 & 0.3429 & 0.3429 & 0.3240 & 0.3240 \\
0.9553 & 0.9523 & 0.9492 & 0.9444 & 0.9394 & 0.9427 & 0.9460 & 0 \\
-0.1509 & -0.1541 & -0.1620 & -0.1667 & -0.1683 & -0.1652 & 0 & 0 \\
-0.1587 & -0.1581 & -0.1574 & -0.1573 & -0.1571 & 0 & 0 & 0 \\
0.0130 & 0.0134 & 0.0137 & 0.0137 & 0 & 0 & 0 & 0 \\
0.0079 & 0.0079 & 0.0079 & 0 & 0 & 0 & 0 & 0 \\
-0.0004 & -0.0005 & 0 & 0 & 0 & 0 & 0 & 0 \\
-0.0003 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

The relevant polynomials are therefore

\[
H_5(x) = 0.2955 + 0.9553(x - 0.3) - 0.1509(x - 0.3)^2 \\
- 0.1587(x - 0.3)^2(x - 0.32) + 0.0130(x - 0.3)^2(x - 0.32)^2 \\
+ 0.0079(x - 0.3)^2(x - 0.32)^2(x - 0.35)
\]

and

\[
H_7(x) = H_5(x) - 0.0004(x - 0.3)^2(x - 0.32)^2(x - 0.35)^2 - 0.0003(x - 0.3)^2(x - 0.32)^2(x - 0.35)^2(x - 0.33).
\]

(b) In general, the error for a degree-n interpolating polynomial is given by

\[
|P_n(x) - f(x)| = \frac{f^{(n+1)}(\xi)}{(n+1)!} \omega(x).
\]

Being lazy and simply using the bound \(|\sin(x)| \leq 1\), we get the bounds

\[
|H_5(0.34) - f(0.34)| \leq (0.04)^2(0.02)^2(0.01)^2/6! = 8.88 \times 10^{-14},
\]

and

\[
|H_7(0.34) - f(0.34)| \leq (0.04)^2(0.02)^2(0.01)^4/8! = 1.59 \times 10^{-19},
\]

as compared with the true errors of \(2.85 \times 10^{-14}\) and 0, respectively. In the latter case, the polynomial interpolation agrees with the true value to full double precision.
2. (a) The divided difference table is given in vertical layout below:

<table>
<thead>
<tr>
<th></th>
<th>$x_0$</th>
<th>$x_0$</th>
<th>$x_1$</th>
<th>$x_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x_0)$</td>
<td>$f(x_0)$</td>
<td>$f(x_1)$</td>
<td>$f(x_1)$</td>
<td></td>
</tr>
<tr>
<td>$f'(x_0)$</td>
<td>$\frac{f(x_0)-f(x_1)}{x_0-x_1}$</td>
<td>$f'(x_1)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$f'(x_0)(x_0-x_1)-f(x_0)+f(x_1)$</td>
<td>$f(x_0)-f(x_1)-f'(x_1)(x_0-x_1)(x_0-x_1)^2$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$f'(x_0)+f'(x_1) - \frac{2f(x_0)-f(x_1)}{(x_0-x_1)^2}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b) To show that the cubic Hermite polynomial $H_3(x)$ can be written as

$$P_3(x) = f[z_0] + f[z_0, z_1](x-x_0) + f[z_0, z_1, z_2](x-x_0)^2 + f[z_0, z_1, z_2, z_3](x-x_0)^3(x-x_1),$$

we check that the function values and first derivatives agree with $f$ at $x_0$ and $x_1$.

i. $P_3(x_0) = f[z_0] = x_0$.

ii. $P_3'(x_0) = f[z_0, z_1] = f'(x_0)$.

iii.

$$P_3(x_1) = f[z_0] + f[z_0, z_1](x_1-x_0) + f[z_0, z_1, z_2](x_1-x_0)^2$$
$$= f(x_0) + f'(x_0)(x_1-x_0) + f'(x_0)(x_0-x_1) - f(x_0) + f(x_1)$$
$$= f(x_1)$$

iv.

$$P_3'(x_1) = f[z_0, z_1] + 2f[z_0, z_1, z_2](x_1-x_0) + f[z_0, z_1, z_2, z_3](x_1-x_0)^2$$
$$= f'(x_0) - 2f'(x_0) + 2\frac{f(x_0)-f(x_1)}{x_0-x_1} + f'(x_0) + f'(x_1) + 2\frac{f(x_1)-f(x_0)}{x_0-x_1}$$
$$= f'(x_1).$$

Since Hermite interpolation is unique, we can conclude that $P_3(x) = H_3(x)$. 

3. (a) If \( P(x) = a_{2n+1}x^{2n+1} + \cdots + a_1x + a_0 \), setting \( P(x_j) = f(x_j) \) and \( P'(x_j) = f'(x_j) \) leads to the \( (2n+2) \times (2n+2) \) square linear system

\[
\begin{bmatrix}
1 & x_0 & x_0^2 & \cdots & x_0^{2n+1} \\
0 & 1 & 2x_0 & \cdots & (2n+1)x_0^{2n} \\
1 & x_1 & x_1^2 & \cdots & x_1^{2n+1} \\
0 & 1 & 2x_n & \cdots & (2n+1)x_n^{2n}
\end{bmatrix}
\begin{bmatrix}
a_0 \\
a_1 \\
\vdots \\
a_{2n+1}
\end{bmatrix}
= \begin{bmatrix}
f(x_0) \\
f'(x_0) \\
f(x_1) \\
\vdots \\
f'(x_n)
\end{bmatrix}.
\]

We know by using the Newton basis for Hermite interpolation that a solution exists for every right-hand side. Then since the matrix is square it must be invertible, and therefore the solution is unique.

Alternate solution: Assume that \( P(x) \) and \( H_{2n+1}(x) \) both agree with \( f \) and \( f' \) at \( x_0, \ldots, x_n \), and consider the polynomial \( D = H_{2n+1} - P \). Since \( D \) is a degree \( 2n+1 \) polynomial such that \( D(x_i) = D'(x_i) = 0 \) for \( 0 \leq i \leq n \), the polynomial \( D \) has \( 2n+2 \) zeros and must therefore be the zero polynomial. Since \( D = 0 \), we conclude that \( H_{2n+1}(x) = P(x) \) and therefore that the interpolating polynomial is unique. Since the system of linear equations is square, the matrix is invertible and so a solution always exists.

(b) As for the error term, define the polynomial \( g(t) \) by

\[
g(t) = f(t) - H_{2n+1}(t) - (f(x) - H_{2n+1}(x)) \prod_{i=0}^{n} \frac{(t - x_i)^2}{(x - x_i)^2}.
\]

Then \( g(x_i) = g'(x_i) = 0 \) for \( 0 \leq i \leq n \), and we can additionally check that \( g(x) = f(x) - H_{2n+1}(x) - [f(x) - H_{2n+1}(x)] = 0 \). Thus \( g \) has at least \( 2n + 3 \) zeros in the interval \([a, b]\), so by the Generalized Rolle’s Theorem there exists some \( \xi \in [a, b] \) such that

\[
0 = g^{(2n+2)}(\xi) = f^{(2n+2)}(\xi) - \frac{(2n+2)!}{\prod_{i=0}^{n} (x - x_i)^2} (f(x) - H_{2n+1}(x)).
\]

Rearranging gives

\[
f(x) = H_{2n+1}(x) + \frac{\prod_{i=0}^{n} (x - x_i)^2}{(2n+2)!} f^{(2n+2)}(\xi),
\]

as desired.
4. (a) \( f(x) = 2^x = e^{x \ln 2} \), so \( f^{(p)}(x) = (\ln 2)^p e^{x \ln 2} = (\ln 2)^p 2^x \).

(b) Define \( k = \ln 2 \). Then we get the (vertically-oriented) divided-difference table

\[
\begin{array}{cccccc}
0 & 0 & 0 & 2 & 2 & 2 \\
1 & 1 & 1 & 4 & 4 & 4 \\
\frac{k}{4} & k & \frac{3}{4} - \frac{k}{4} & 4k & 4k \\
-\frac{k^2}{2} - \frac{k}{4} + \frac{3}{8} & \frac{5}{4}k - \frac{3}{4} & 2k^2 - k + \frac{3}{8} \\
\frac{3}{8}k^2 - \frac{15}{16}k + \frac{9}{16} & \\
\end{array}
\]

Note that this table for \( p = 2 \) also contains inside it the tables for \( p = 0 \) and \( p = 1 \). We then get the polynomials

\[
H_0(x) = 1 + \frac{3}{2}x \\
H_1(x) = 1 + kx + \left( \frac{3}{4} - \frac{k}{2} \right) x^2 + \left( \frac{5}{4}k - \frac{3}{4} \right) x^2(x-2) \\
H_2(x) = 1 + kx + k^2x^2 + \left( -\frac{k^2}{2} - k + \frac{3}{8} \right) x^3 \\
+ \left( \frac{k^2}{4} + \frac{3}{4}k - \frac{9}{16} \right) x^3(x-2) \\
+ \left( \frac{3}{8}k^2 - \frac{15}{16}k + \frac{9}{16} \right) x^3(x-2)^2.
\]

(c) For general \( p \geq 1 \) and for \( 0 \leq x \leq 2 \) we get the error bound

\[
|f(x) - H_p(x)| \leq \frac{|f^{(2p+2)}(\xi)|}{(2p + 2)!} \prod_{j=0}^{1} |(x - x_j)|^{p+1} \\
\leq \frac{4 \ln(2)^{2p+2}}{(2p + 2)!} \\
\leq \frac{4 \ln(2)^{2p+2}}{\sqrt{2\pi(2p+2)(2(p+1)/e)^{2p+2}}} \\
= \frac{4}{\sqrt{2\pi(2p+2)}} \left( \frac{1}{p+1} \right)^{2p+2} \left( \frac{e \ln(2)}{2} \right)^{2p+2} \\
\leq \frac{4}{\sqrt{8\pi}} \left( \frac{e \ln(2)}{2} \right)^{2p+2} \left( \frac{1}{p+1} \right)^{2p+2} \\
\leq \left( \frac{1}{p+1} \right)^{2p+2}.
\]

The assumption \( p \geq 1 \) was used in the second-to-last inequality, substituting \( (2p + 2) \geq 4 \). Checking the case \( p = 0 \) separately, we find that

\[
\frac{4 \ln(2)^{2p+2}}{(2p + 2)!} = 2 \ln(2)^2 < 1 = \left( \frac{1}{p+1} \right)^{2p+2},
\]

which completes the proof.
(d) If we want to use Newton’s method to solve the equation

\[ g(y) = x \ln 2 - \ln y = 0, \]

then we get the fixed-function (and corresponding derivatives)

\[ h(y) = y(1 + x \ln 2 - \ln y) \]
\[ h'(y) = x \ln 2 - \ln y \]
\[ h''(y) = -1/y. \]

Expanding \( h \) as a Taylor series around \( y_\star = 2^x \), we get that

\[ h(y_\star + \delta) = h(y_\star) + \delta h'(y_\star) + \frac{\delta^2 h''(\xi)}{2} \]

for some \( \xi \in [y_\star, y_\star + \delta] \) and therefore (since \( h'(y_\star) = 0 \)) that

\[ |y_1 - y_\star| = \frac{\delta^2 |h''(\xi)|}{2} = \frac{|y_0 - y_\star|^2}{2\xi} \leq \frac{|y_0 - y_\star|^2}{2} \]

From the previous part we know that

\[ |2^x - y_0| \leq \frac{4}{\sqrt{2\pi(2p + 2)}} \left( \frac{1}{p + 1} \right)^{2p+2} \left( \frac{e \ln(2)}{2} \right)^{2p+2} \]
\[ = \frac{2}{\sqrt{5\pi}} \left( \frac{e \ln(2)}{10} \right)^{10} \]
\[ \leq 2^{-25} \]

so combining this bound with the result above gives that

\[ |2^x - y_1| \leq \frac{1}{2} (2^{-25})^2 = 2^{-51}. \]

... close enough? If you want to get the final bit of precision, you can use the fact that our bounds were not quite optimal when taken all at once: \( \omega(x) \) is maximized at \( x = 1 \), but in this case we would have \( 1/\xi \approx 1/2 \), getting us the final bit of precision. If we stray far from \( x = 1 \), on the other hand, then \( \omega(x) \) will decrease quickly.
5. Based on the formula (4.2)

\[ f'(x_j) = \sum_{k=0}^{n} f(x_k) L'_k(x_j) + \frac{f^{(n+1)}(\xi(x_j))}{(n+1)!} \prod_{k \neq j} (x_j - x_k), \]

we will achieve the tightest error bound by finding some number \( n \) and points \( \{x_0, \ldots, x_n\} \) such that the absolute value of \( \frac{f^{(n+1)}(\xi(x_j))}{(n+1)!} \prod_{k \neq j} (x_j - x_k) \) is minimized. We need to include the point \( x_0 = 3 \), and based on the formula it is strictly to our advantage to choose any new points as close to 3 as possible. The sign of the terms \( (x_j - x_k) \) is irrelevant, so without loss of generality we can add new points larger than 3 before adding ones smaller than 3. Based on the derivative bounds given in the problem, we can check all possibilities with the following table:

| \( n \) | \( \{x_0, \ldots, x_n\} \) | \[ \left| \frac{f^{(n+1)}(\xi(x_j))}{(n+1)!} \prod_{k \neq j} (x_j - x_k) \right| \]
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( {3} )</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>( {3, 4} )</td>
<td>( \frac{3}{2} )</td>
</tr>
<tr>
<td>3</td>
<td>( {3, 4, 2} )</td>
<td>( \frac{6}{6} = 1 )</td>
</tr>
<tr>
<td>4</td>
<td>( {3, 4, 2, 5} )</td>
<td>( \frac{12}{24} = 1 )</td>
</tr>
<tr>
<td>5</td>
<td>( {3, 4, 2, 5, 1} )</td>
<td>( \frac{234}{120} = \frac{23}{30} )</td>
</tr>
</tbody>
</table>

The last error bound is the smallest, so based on the five-point midpoint formula

\[ f'(x_0) = \frac{1}{12h} [f(x_0 - 2h) - 8f(x_0 - h) + 8f(x_0 + h) - f(x_0 + 2h)] + 30 f^{(5)}(\xi), \]

we get the approximation \( f'(3) \approx 0.2106 \), with error bounded by 23/30.
6. (a) First we prove the recurrence relation efficiently. Observe that the Lagrange basis polynomials satisfy a recursion

\[ L^n_k(x) = \prod_{\substack{i=0 \atop i \neq k}}^n \frac{x-x_i}{x_k-x_i} = \frac{x-x_n}{x_k-x_n} \prod_{\substack{i=0 \atop i \neq k}}^{n-1} \frac{x-x_i}{x_k-x_i} = \frac{x-x_n}{x_k-x_n} L^{n-1}_k(x) \]

when \(0 \leq k < n\). Evaluating at \(x=a\) gives the recurrence relation for \(m=0\):

\[ \delta^0_{nk} = \frac{a-x_n}{x_k-x_n} \delta^0_{n-1,k}. \]

Differentiating the recursion for Lagrange basis polynomials gives

\[ \frac{d}{dx} L^n_k(x) = \frac{1}{x_k-x_n} L^{n-1}_k(x) + \frac{x-x_n}{x_k-x_n} \frac{d}{dx} L^{n-1}_k(x), \]

and evaluating at \(x=a\) gives the recurrence relation for \(m=1\):

\[ \delta^1_{nk} = \frac{1}{x_k-x_n} \delta^0_{n-1,k} + \frac{a-x_n}{x_k-x_n} \delta^1_{n-1,k}. \]

In order to calculate higher derivatives, we observe that

\[ \left( \frac{d}{dx} \right)^m (f(x)g(x)) = \left( \frac{d}{dx} \right)^{m-1} (f'(x)g(x) + f(x)g'(x)) \]

\[ = \left( \frac{d}{dx} \right)^{m-2} (2f'(x)g(x) + f(x)g''(x)) \]

\[ = \cdots \]

\[ = mf'(x)g^{(m-1)}(x) + f(x)g^{(m)}(x) \]

whenever \(f\) is a linear function so that \(f'' = 0\). Thus

\[ \left( \frac{d}{dx} \right)^m L^n_k(x) = \frac{m}{x_k-x_n} \left( \frac{d}{dx} \right)^{m-1} L^{n-1}_k(x) + \frac{x-x_n}{x_k-x_n} \left( \frac{d}{dx} \right)^m L^{n-1}_k(x). \]

Evaluating at \(x=a\) gives the desired recurrence relation.

We also prove the recurrence relation as in lecture, because it gives a useful consequence. The key observation is that because the coefficients \(\delta^m_{nj}\) are \(m\)th derivatives of \(L^n_j(t)\) evaluated at \(t=a\), there is a Taylor expansion of the Lagrange basis polynomial:

\[ L^n_j(t) = \sum_{m=0}^n \frac{\delta^m_{nj}}{m!} (t-a)^m. \]

Since we also know from the explicit formula that

\[ L^n_j(t) = \frac{t-t_n}{t_j-t_n} L^{n-1}_j(t) \]

for \(0 \leq j < n\), we can equate two power series:

\[ \sum_{m=0}^n \frac{\delta^m_{nj}}{m!} (t-a)^m = \frac{t-a + a - t_n}{t_j-t_n} \sum_{m=0}^{n-1} \frac{\delta^{m-1}_{n-1,j}}{m!} (t-a)^m. \]

Shifting the index of summation gives

\[ \sum_{m=0}^n \frac{\delta^m_{nj}}{m!} (t-a)^m = \frac{1}{t_j-t_n} \sum_{m=1}^n \frac{\delta^{m-1}_{n-1,j}}{(m-1)!} (t-a)^m + \frac{a - t_n}{t_j-t_n} \sum_{m=0}^{n-1} \frac{\delta^{m-1}_{n-1,j}}{m!} (t-a)^m. \]
When two power series are equal, their coefficients must be equal. Thus equating coefficients gives the result.

A useful consequence of the key observation is the evaluation of integration weights. In approximating the integral

$$\int_a^b f(x) dx$$

a standard approach is to evaluate $f_j = f(x_j)$ at $n+1$ interpolation points $x_j$ and then approximate the integral by integrating the degree-$n$ polynomial interpolant

$$p(x) = \sum_{j=0}^n f_j L_j(x).$$

The resulting formula is

$$\int_a^b f(x) dx = \sum_{j=0}^n w_j f_j$$

where the weights are given by

$$w_j = \int_a^b L_j(x) dx.$$

These integrals can be complicated to evaluate for high-degree polynomials, but the key observation above yields the simple formula

$$w_j = \int_a^b \frac{\sum_{m=0}^n \delta_{nj}^m (x-a)^m}{m!} (b-a)^{m+1} dx$$

once the differentiation weights $\delta_{nj}^m$ are evaluated at $x = a$.

(b) The above recurrence relation relies on the inequality $k < n$ holding. This can be trivially enforced by re-indexing, so that $k = 0$. It is clear from the recurrence relation that $\delta^m_{n,k}$ depends on $\delta^m_{n-1,k}$ and $\delta^m_{n-1,k}$, which depend on $\delta^m_{n-2,k}$, $\delta^m_{n-2,k}$, and $\delta^m_{n-2,k}$. More generally, each set of the form $\delta^*_{n,k}$ depends on $\delta^*_{n-1,k}$. Thus, the recurrence relation can build all differentiation coefficients from those of the form

$$\delta^m_{1,k=0}(a) = \begin{cases} \frac{a-x_1}{x_0-x_1} & m = 0 \\ \frac{1}{x_0-x_1} & m = 1 \\ 0 & m > 1 \end{cases}$$

and

$$\delta^0_{n,k}(a) = \prod_{i=1}^{k} \frac{a-x_i}{x_k-x_i}.$$ 

These form the first row and zeroth column of a matrix containing all the differentiation coefficients, which can be constructed moving down the entries, using the recurrence relation. Alternatively, the zeroth column and zeroth row can be used to start, with

$$\delta^m_{0,0} = \begin{cases} 1 & m = 0 \\ 0 & m > 0 \end{cases}$$

With the first row and column predetermined this way, the coefficients can be created by applying the recurrence relation and extracting the last row of the matrix. Below is a MATLAB code that uses this implementation of the recurrence relation:
function [Deltas] = diff_coeffs(n, M, a, x)
    d_start = zeros(n+1, M+1);
    Deltas = d_start;
    d_start(1, 1) = 1;
    for k = 1:n+1
        d = d_start;
        xx = x;
        xx([1 k]) = xx([k 1]);
        for m = 2:M+1
            for nn = 2:n+1
                d(nn,m) = (d(nn-1,m-1)*m + d(nn-1,m)*(a-x(nn)))/(x(1)-x(nn))
            end
        end
        Deltas(k,:) = d(n+1,:)
    end

(c) The formula given,
\[ p^{(m)}(a) = \sum_{k=0}^{n} \delta_{nk}^m(a)f_k, \]

can be described as a vector-matrix multiplication of the row vector \( \vec{v} \) of \( f \) evaluated at all the points in the given \( x \) vector, which has entries \( v_j = f(jh) \), multiplied by the matrix \( D \) whose entries are given by
\[ D_{k,m} = \delta_{nk}^m(a) \]

Then the vector \( \vec{v}D \) gives the approximations of the derivatives of the function \( f \). Setting \( h = 0.1 \), \( n = 4 \), \( a = 0.5 \), and \( M = 5 \) we get the following errors:
\[
\begin{bmatrix}
1.28670303 \times 10^{-05} & 2.96019564 \times 10^{-04} & 4.92727672 \times 10^{-03} & 5.72299634 \times 10^{-02} & 4.25281314 \times 10^{-01} \\
\end{bmatrix}
\]

Setting \( h = 0.0666666667 \), i.e. dividing \( h \) by 1.5, we get the errors
\[
\begin{bmatrix}
5.63954233 \times 10^{-05} & 8.36240212 \times 10^{-04} & 9.79325784 \times 10^{-03} & 8.55452261 \times 10^{-02} & 5.05243784 \times 10^{-01} \\
\end{bmatrix}
\]

Thus the ratio of each pair of errors is
\[
\begin{bmatrix}
4.3829401 & 2.82494914 & 1.98755994 & 1.4947629 & 1.18802253 \\
\end{bmatrix}
\]

As the errors should be on the order \( h^{n-m} \), this ratio should ideally be \( 1.5^{n-m} \), or
\[
\begin{bmatrix}
5.0625 & 3.375 & 2.25 & 1.5 & 1 \\
\end{bmatrix}
\]

The results are close enough to claim order \( h^{n-m} \) convergence.

(d) In approaching this problem, it helps to see the matrix \( A_m \) as an operator that takes the \( m \)th derivative of \( p \) when it multiplies \( f \). Thus, it is reasonable to see that \( m \) applications of \( A_1 \) should be equivalent to taking the first derivative \( m \) times, giving, finally, the \( m \)th derivative. Thus, intuition would suggest that \( A_m = A_1^m \). To prove this:

It is clearly true that \( A_m = A_1^m \) when \( m = 1 \), as this is simply \( A_1 = A_1^1 \), and any matrix is equal to itself to the first power. Assume that \( A_b = A_1^b \) for some integer \( b \). Now, setting \( m = 1 \) in the given interpolation formula,
\[
p'(a) = \sum_{k=0}^{n} \delta_{nk}^1(a)f_k
\]
This can be applied to the case where \( f = \left( \frac{d}{dx} \right)^b L^n_j \), and \( a = x_i \). Since \( f \) is a polynomial of degree less than \( n \), interpolation is exact, and \( p = f \). Then \( f_k = p(x_k) = \left( \frac{d}{dx} \right)^b L^n_j(x)|_{x=x_k} = \delta^n_{nj}(x_k) \).

Therefore,

\[
\left( \frac{d}{dx} \right)^{b+1} L^n_j(x)|_{x=x_i} = p'(x_i) = p'(a) = \sum_{k=0}^{n} \delta^{1}_{nk}(a)f_k = \sum_{k=0}^{n} \delta^{1}_{nk}(x)\delta^{b}_{nj}(x_k)
\]

Observe that matrix multiplication of \( A_1 \) and \( A_b \) gives

\[
(A_1 A_b)_{ij} = \sum_{k=0}^{n} (A_1)_{ik}(A_b)_{kj} = \sum_{k=0}^{n} \delta^{1}_{nk}(x)\delta^{b}_{nj}(x_k)
\]

Furthermore,

\[
\left( \frac{d}{dx} \right)^{b+1} L^n_j(x)|_{x=x_i} = \delta^{b+1}_{nj}(x_i) = (A_{b+1})_{ij}
\]

Therefore

\[
(A_{b+1})_{ij} = (A_1 A_b)_{ij}
\]

Since it is assumed that \( A_b = A_1^b \), then

\[
(A_{b+1})_{ij} = (A_1 A_b)_{ij} = (A_1 A_1^b)_{ij} = (A_1^{b+1})_{ij}
\]

Since every element agrees, \( A_{b+1} = A_1^{b+1} \), and, by induction, \( A_m = A_1^n \) for every integer \( m \).