MA 322: Scientific Computing

Lecture - 6

Interpolating polynomial in monomial basis

- Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.

Interpolating polynomial in monomial basis

- Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.
- Consider the monomial basis $1, x, \ldots, x^n$, which corresponds to the choice $\phi_j(x) = x^j$ for $j = 0 : n$.

Interpolating polynomial in monomial basis

- Consider the nodes $[x_0, \dots, x_n]$ and the values $[f_0, \dots, f_n]$.
- Consider the monomial basis $1, x, \dots, x^n$, which corresponds to the choice $\phi_j(x) = x^j$ for $j = 0 : n$. Then $p(x) = a_0 + a_1 x + \cdots + a_n x^n$ is obtained by solving the Vandermonde system

$$\begin{bmatrix} 1 & x_0 & \cdots & x_0^n \\ 1 & x_1 & \cdots & x_1^n \\ \vdots & \vdots & \dots & \vdots \\ 1 & x_n & \cdots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_n \end{bmatrix}.$$

Interpolating polynomial in monomial basis

- Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.
- Consider the monomial basis $1, x, \ldots, x^n$, which corresponds to the choice $\phi_j(x) = x^j$ for $j = 0 : n$. Then $p(x) = a_0 + a_1 x + \cdots + a_n x^n$ is obtained by solving the Vandermonde system

$$\begin{bmatrix} 1 & x_0 & \cdots & x_0^n \\ 1 & x_1 & \cdots & x_1^n \\ \vdots & \vdots & \ldots & \vdots \\ 1 & x_n & \cdots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_n \end{bmatrix}.$$

Drawbacks:

- The Vandermonde matrix is extremely ill-conditioned.

Interpolating polynomial in monomial basis

- Consider the nodes $[x_0, \dots, x_n]$ and the values $[f_0, \dots, f_n]$.
- Consider the monomial basis $1, x, \dots, x^n$, which corresponds to the choice $\phi_j(x) = x^j$ for $j = 0 : n$. Then $p(x) = a_0 + a_1 x + \cdots + a_n x^n$ is obtained by solving the Vandermonde system

$$\begin{bmatrix} 1 & x_0 & \cdots & x_0^n \\ 1 & x_1 & \cdots & x_1^n \\ \vdots & \vdots & \dots & \vdots \\ 1 & x_n & \cdots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_n \end{bmatrix}.$$

Drawbacks:

- The Vandermonde matrix is extremely ill-conditioned.
- The solution of the system requires large numbers of operations.

Interpolating polynomial in monomial basis

- Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.
- Consider the monomial basis $1, x, \ldots, x^n$, which corresponds to the choice $\phi_j(x) = x^j$ for $j = 0 : n$. Then $p(x) = a_0 + a_1 x + \cdots + a_n x^n$ is obtained by solving the Vandermonde system

$$\begin{bmatrix} 1 & x_0 & \cdots & x_0^n \\ 1 & x_1 & \cdots & x_1^n \\ \vdots & \vdots & \ldots & \vdots \\ 1 & x_n & \cdots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_n \end{bmatrix}.$$

Drawbacks:

- The Vandermonde matrix is extremely ill-conditioned.
- The solution of the system requires large numbers of operations.
- Any additional new data (x_{n+1}, f_{n+1}) requires re-computation.

Example

Consider the data set $(x_0, f_0), (x_1, f_1), (x_2, f_2)$. Then there is a unique interpolating polynomial $p(x) = a_0 + a_1 x + a_2 x^2$ of degree ≤ 2.

Example

Consider the data set $(x_0, f_0), (x_1, f_1), (x_2, f_2)$. Then there is a unique interpolating polynomial $p(x) = a_0 + a_1 x + a_2 x^2$ of degree ≤ 2. The Vandermonde system is given by

$$\begin{bmatrix} 1 & x_0 & x_0^2 \\ 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \end{bmatrix}.$$

Example

Consider the data set $(x_0, f_0), (x_1, f_1), (x_2, f_2)$. Then there is a unique interpolating polynomial $p(x) = a_0 + a_1 x + a_2 x^2$ of degree ≤ 2.
The Vandermonde system is given by

$$\begin{bmatrix} 1 & x_0 & x_0^2 \\ 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \end{bmatrix}.$$

For the data set $(-2, -27), (0, -1), (1, 0)$, we have

$$\begin{bmatrix} 1 & -2 & 4 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} -27 \\ -1 \\ 0 \end{bmatrix}.$$

Example

Consider the data set $(x_0, f_0), (x_1, f_1), (x_2, f_2)$. Then there is a unique interpolating polynomial $p(x) = a_0 + a_1 x + a_2 x^2$ of degree ≤ 2. The Vandermonde system is given by

$$\begin{bmatrix} 1 & x_0 & x_0^2 \\ 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \end{bmatrix}.$$

For the data set $(-2, -27), (0, -1), (1, 0)$, we have

$$\begin{bmatrix} 1 & -2 & 4 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} -27 \\ -1 \\ 0 \end{bmatrix}.$$

Solving the system, we have $[a_0, a_1, a_2] = [-1, 5, -4]$ so that the interpolating polynomial is given by

$$p(x) = -1 + 5x - 4x^2.$$

Interpolating polynomial in Lagrange basis

- Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.

Interpolating polynomial in Lagrange basis

- Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.
- Consider the bases $\phi_0(x), \ldots, \phi_n(x)$ and $\ell_0(x), \ldots, \ell_n(x)$ given by

Interpolating polynomial in Lagrange basis

- Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.

- Consider the bases $\phi_0(x), \ldots, \phi_n(x)$ and $\ell_0(x), \ldots, \ell_n(x)$ given by

$$\phi_0(x) \quad := \quad (x - x_1)(x - x_2) \cdots (x - x_n) \text{ and } \ell_0(x) = \frac{\phi_0(x)}{\phi_0(x_0)}$$

Interpolating polynomial in Lagrange basis

- Consider the nodes $[x_0, \dots, x_n]$ and the values $[f_0, \dots, f_n]$.
- Consider the bases $\phi_0(x), \dots, \phi_n(x)$ and $\ell_0(x), \dots, \ell_n(x)$ given by

$$
\begin{aligned}
\phi_0(x) &:= (x - x_1)(x - x_2)\cdots(x - x_n) \text{ and } \ell_0(x) = \frac{\phi_0(x)}{\phi_0(x_0)} \\
\phi_1(x) &:= (x - x_0)(x - x_2)\cdots(x - x_n) \text{ and } \ell_1(x) := \frac{\phi_1(x)}{\phi_1(x_1)}
\end{aligned}
$$

Interpolating polynomial in Lagrange basis

- Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.
- Consider the bases $\phi_0(x), \ldots, \phi_n(x)$ and $\ell_0(x), \ldots, \ell_n(x)$ given by

$$
\begin{aligned}
\phi_0(x) &:= (x-x_1)(x-x_2)\cdots(x-x_n) \text{ and } \ell_0(x) = \frac{\phi_0(x)}{\phi_0(x_0)} \\
\phi_1(x) &:= (x-x_0)(x-x_2)\cdots(x-x_n) \text{ and } \ell_1(x) := \frac{\phi_1(x)}{\phi_1(x_1)} \\
&\vdots \\
\phi_n(x) &:= (x-x_0)(x-x_1)\cdots(x-x_{n-1}) \text{ and } \ell_n(x) := \frac{\phi_n(x)}{\phi_n(x_n)}
\end{aligned}
$$

Interpolating polynomial in Lagrange basis

- Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.

- Consider the bases $\phi_0(x), \ldots, \phi_n(x)$ and $\ell_0(x), \ldots, \ell_n(x)$ given by

$$
\begin{aligned}
\phi_0(x) &:= (x - x_1)(x - x_2)\cdots(x - x_n) \text{ and } \ell_0(x) = \frac{\phi_0(x)}{\phi_0(x_0)} \\
\phi_1(x) &:= (x - x_0)(x - x_2)\cdots(x - x_n) \text{ and } \ell_1(x) := \frac{\phi_1(x)}{\phi_1(x_1)} \\
&\vdots \\
\phi_n(x) &:= (x - x_0)(x - x_1)\cdots(x - x_{n-1}) \text{ and } \ell_n(x) := \frac{\phi_n(x)}{\phi_n(x_n)}
\end{aligned}
$$

- The basis $\ell_0(x), \ldots, \ell_n(x)$ is called the Lagrange basis of $\mathcal{P}_n$.

Interpolating polynomial in Lagrange basis

- Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.
- Consider the bases $\phi_0(x), \ldots, \phi_n(x)$ and $\ell_0(x), \ldots, \ell_n(x)$ given by

$$
\begin{aligned}
\phi_0(x) &:= (x - x_1)(x - x_2)\cdots(x - x_n) \text{ and } \ell_0(x) = \frac{\phi_0(x)}{\phi_0(x_0)} \\
\phi_1(x) &:= (x - x_0)(x - x_2)\cdots(x - x_n) \text{ and } \ell_1(x) := \frac{\phi_1(x)}{\phi_1(x_1)} \\
&\vdots \\
\phi_n(x) &:= (x - x_0)(x - x_1)\cdots(x - x_{n-1}) \text{ and } \ell_n(x) := \frac{\phi_n(x)}{\phi_n(x_n)}
\end{aligned}
$$

- The basis $\ell_0(x), \ldots, \ell_n(x)$ is called the Lagrange basis of $\mathcal{P}_n$.
- Then it follows that

$$
\ell_j(x) = \prod_{i \neq j} \frac{(x - x_i)}{(x_j - x_i)}, \quad j = 0 : n.
$$

Interpolating polynomial in Lagrange basis (cont.)

Note that $\ell_j(x_i) = \delta_{ij}$, where δ_{ij} is the Dirac delta function. Hence it follows that the polynomial

$$p(x) := f_0\ell_0(x) + \cdots + f_n\ell_n(x)$$

interpolates the data set $(x_0, f_0), \ldots, (x_n, f_n)$ and is called the Lagrange interpolating polynomial.

Interpolating polynomial in Lagrange basis (cont.)

Note that $\ell_j(x_i) = \delta_{ij}$, where δ_{ij} is the Dirac delta function. Hence it follows that the polynomial

$$p(x) := f_0\ell_0(x) + \cdots + f_n\ell_n(x)$$

interpolates the data set $(x_0, f_0), \ldots, (x_n, f_n)$ and is called the Lagrange interpolating polynomial.

Drawbacks:

- Cannot accommodate new data (x_{n+1}, f_{n+1}).

Interpolating polynomial in Lagrange basis (cont.)

Note that $\ell_j(x_i) = \delta_{ij}$, where δ_{ij} is the Dirac delta function. Hence it follows that the polynomial

$$p(x) := f_0\ell_0(x) + \cdots + f_n\ell_n(x)$$

interpolates the data set $(x_0, f_0), \ldots, (x_n, f_n)$ and is called the Lagrange interpolating polynomial.

Drawbacks:

- Cannot accommodate new data (x_{n+1}, f_{n+1}).
- Any additional new data requires re-computation.

Interpolating polynomial in Lagrange basis (cont.)

Note that $\ell_j(x_i) = \delta_{ij}$, where δ_{ij} is the Dirac delta function. Hence it follows that the polynomial

$$p(x) := f_0\ell_0(x) + \cdots + f_n\ell_n(x)$$

interpolates the data set $(x_0, f_0), \ldots, (x_n, f_n)$ and is called the Lagrange interpolating polynomial.

Drawbacks:

- Cannot accommodate new data (x_{n+1}, f_{n+1}).
- Any additional new data requires re-computation.

Example: For the data set $(-2, -27), (0, -1), (1, 0)$, we have

Interpolating polynomial in Lagrange basis (cont.)

Note that $\ell_j(x_i) = \delta_{ij}$, where δ_{ij} is the Dirac delta function. Hence it follows that the polynomial

$$p(x) := f_0\ell_0(x) + \cdots + f_n\ell_n(x)$$

interpolates the data set $(x_0, f_0), \ldots, (x_n, f_n)$ and is called the Lagrange interpolating polynomial.

Drawbacks:

- Cannot accommodate new data (x_{n+1}, f_{n+1}).
- Any additional new data requires re-computation.

Example: For the data set $(-2, -27), (0, -1), (1, 0)$, we have

$$p(x) = -27\frac{(x-0)(x-1)}{(-2-0)(-2-1)} - 1\frac{(x+2)(x-1)}{(0+2)(0-1)} + 0\frac{(x+2)(x-0)}{(1+2)(1-0)}$$

Interpolating polynomial in Lagrange basis (cont.)

Note that $\ell_j(x_i) = \delta_{ij}$, where δ_{ij} is the Dirac delta function. Hence it follows that the polynomial

$$p(x) := f_0\ell_0(x) + \cdots + f_n\ell_n(x)$$

interpolates the data set $(x_0, f_0), \ldots, (x_n, f_n)$ and is called the Lagrange interpolating polynomial.

Drawbacks:

- Cannot accommodate new data (x_{n+1}, f_{n+1}).
- Any additional new data requires re-computation.

Example: For the data set $(-2, -27), (0, -1), (1, 0)$, we have

$$\begin{aligned} p(x) &= -27\frac{(x-0)(x-1)}{(-2-0)(-2-1)} - 1\frac{(x+2)(x-1)}{(0+2)(0-1)} + 0\frac{(x+2)(x-0)}{(1+2)(1-0)} \\ &= -\frac{9}{2}x(x-1) + \frac{1}{2}(x+2)(x-1) \end{aligned}$$

Interpolating polynomial in Lagrange basis (cont.)

Note that $\ell_j(x_i) = \delta_{ij}$, where δ_{ij} is the Dirac delta function. Hence it follows that the polynomial

$$p(x) := f_0\ell_0(x) + \cdots + f_n\ell_n(x)$$

interpolates the data set $(x_0, f_0), \ldots, (x_n, f_n)$ and is called the Lagrange interpolating polynomial.

Drawbacks:

- Cannot accommodate new data (x_{n+1}, f_{n+1}).
- Any additional new data requires re-computation.

Example: For the data set $(-2, -27), (0, -1), (1, 0)$, we have

$$\begin{aligned} p(x) &= -27\frac{(x-0)(x-1)}{(-2-0)(-2-1)} - 1\frac{(x+2)(x-1)}{(0+2)(0-1)} + 0\frac{(x+2)(x-0)}{(1+2)(1-0)} \\ &= -\frac{9}{2}x(x-1) + \frac{1}{2}(x+2)(x-1) = -1 + 5x - 4x^2 \end{aligned}$$

Interpolating polynomial in Newton basis

Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.

Interpolating polynomial in Newton basis

Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.

Define $\mathrm{N}_0(x) := 1$ and $\mathrm{N}_j(x) := (x - x_0) \cdots (x - x_{j-1})$ for $j = 1 : n$.

Interpolating polynomial in Newton basis

Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.

Define $\mathrm{N}_0(x) := 1$ and $\mathrm{N}_j(x) := (x - x_0) \cdots (x - x_{j-1})$ for $j = 1 : n$.

Fact: The Newton polynomials $\mathrm{N}_0(x), \ldots, \mathrm{N}_n(x)$ form a basis of $\mathcal{P}_n$ which is referred to as Newton basis.

Interpolating polynomial in Newton basis

Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.

Define $\mathrm{N}_0(x) := 1$ and $\mathrm{N}_j(x) := (x - x_0) \cdots (x - x_{j-1})$ for $j = 1 : n$.

Fact: The Newton polynomials $\mathrm{N}_0(x), \ldots, \mathrm{N}_n(x)$ form a basis of $\mathcal{P}_n$ which is referred to as Newton basis.

Let $p(x) := a_0\mathrm{N}_0(x) + \cdots + a_n\mathrm{N}_n(x)$. The interpolation conditions $p(x_j) = f_j$ for $j = 0 : n$ yield the lower triangular system

Interpolating polynomial in Newton basis

Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.

Define $\mathrm{N}_0(x) := 1$ and $\mathrm{N}_j(x) := (x - x_0)\cdots(x - x_{j-1})$ for $j = 1 : n$.

Fact: The Newton polynomials $\mathrm{N}_0(x), \ldots, \mathrm{N}_n(x)$ form a basis of $\mathcal{P}_n$ which is referred to as Newton basis.

Let $p(x) := a_0\mathrm{N}_0(x) + \cdots + a_n\mathrm{N}_n(x)$. The interpolation conditions $p(x_j) = f_j$ for $j = 0 : n$ yield the lower triangular system

$$\begin{bmatrix} \mathrm{N}_0(x_0) & 0 & 0 & \cdots & 0 \\ \mathrm{N}_0(x_1) & \mathrm{N}_1(x_1) & 0 & \cdots & 0 \\ \mathrm{N}_0(x_2) & \mathrm{N}_1(x_2) & \mathrm{N}_2(x_2) & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathrm{N}_0(x_n) & \mathrm{N}_1(x_n) & \mathrm{N}_2(x_n) & \cdots & \mathrm{N}_n(x_n) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix}.$$

Interpolating polynomial in Newton basis

Consider the nodes $[x_0, \ldots, x_n]$ and the values $[f_0, \ldots, f_n]$.

Define $\mathrm{N}_0(x) := 1$ and $\mathrm{N}_j(x) := (x - x_0) \cdots (x - x_{j-1})$ for $j = 1 : n$.

Fact: The Newton polynomials $\mathrm{N}_0(x), \ldots, \mathrm{N}_n(x)$ form a basis of $\mathcal{P}_n$ which is referred to as Newton basis.

Let $p(x) := a_0 \mathrm{N}_0(x) + \cdots + a_n \mathrm{N}_n(x)$. The interpolation conditions $p(x_j) = f_j$ for $j = 0 : n$ yield the lower triangular system

$$\begin{bmatrix} \mathrm{N}_0(x_0) & 0 & 0 & \cdots & 0 \\ \mathrm{N}_0(x_1) & \mathrm{N}_1(x_1) & 0 & \cdots & 0 \\ \mathrm{N}_0(x_2) & \mathrm{N}_1(x_2) & \mathrm{N}_2(x_2) & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathrm{N}_0(x_n) & \mathrm{N}_1(x_n) & \mathrm{N}_2(x_n) & \cdots & \mathrm{N}_n(x_n) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix}.$$

Solving the triangular system for $a_j, j = 0 : n$, we obtain the interpolating polynomial $p(x) = a_0 \mathrm{N}_0(x) + \cdots + a_n \mathrm{N}_n(x)$ in Newton's form.

Interpolating polynomial in Newton basis (cont.)

Advantage: Can accommodate new data (x_{n+1}, f_{n+1}) with additional computational cost.

Interpolating polynomial in Newton basis (cont.)

Advantage: Can accommodate new data (x_{n+1}, f_{n+1}) with additional computational cost.

Example: For the data set $(x_0, f_0), (x_1, f_1), (x_2, f_2)$, we have

Interpolating polynomial in Newton basis (cont.)

Advantage: Can accommodate new data (x_{n+1}, f_{n+1}) with additional computational cost.

Example: For the data set $(x_0, f_0), (x_1, f_1), (x_2, f_2)$, we have

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & (x_1 - x_0) & 0 \\ 1 & (x_2 - x_0) & (x_2 - x_0)(x_2 - x_1) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \end{bmatrix}.$$

For the data set $(-2, -27), (0, -1), (1, 0)$, we have

Interpolating polynomial in Newton basis (cont.)

Advantage: Can accommodate new data (x_{n+1}, f_{n+1}) with additional computational cost.

Example: For the data set $(x_0, f_0), (x_1, f_1), (x_2, f_2)$, we have

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & (x_1 - x_0) & 0 \\ 1 & (x_2 - x_0) & (x_2 - x_0)(x_2 - x_1) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \end{bmatrix}.$$

For the data set $(-2, -27), (0, -1), (1, 0)$, we have

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 2 & 0 \\ 1 & 3 & 3 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} -27 \\ -1 \\ 0 \end{bmatrix}$$

Interpolating polynomial in Newton basis (cont.)

Advantage: Can accommodate new data (x_{n+1}, f_{n+1}) with additional computational cost.

Example: For the data set $(x_0, f_0), (x_1, f_1), (x_2, f_2)$, we have

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & (x_1 - x_0) & 0 \\ 1 & (x_2 - x_0) & (x_2 - x_0)(x_2 - x_1) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \end{bmatrix}.$$

For the data set $(-2, -27), (0, -1), (1, 0)$, we have

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 2 & 0 \\ 1 & 3 & 3 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} -27 \\ -1 \\ 0 \end{bmatrix} \implies [a_0, a_1, a_2] = [-27, 13, -4].$$

Hence

Interpolating polynomial in Newton basis (cont.)

Advantage: Can accommodate new data (x_{n+1}, f_{n+1}) with additional computational cost.

Example: For the data set $(x_0, f_0), (x_1, f_1), (x_2, f_2)$, we have

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & (x_1 - x_0) & 0 \\ 1 & (x_2 - x_0) & (x_2 - x_0)(x_2 - x_1) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \end{bmatrix}.$$

For the data set $(-2, -27), (0, -1), (1, 0)$, we have

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 2 & 0 \\ 1 & 3 & 3 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} -27 \\ -1 \\ 0 \end{bmatrix} \implies [a_0, a_1, a_2] = [-27, 13, -4].$$

Hence $p(x) = -27 + 13(x + 2) - 4(x + 2)x$

Interpolating polynomial in Newton basis (cont.)

Advantage: Can accommodate new data (x_{n+1}, f_{n+1}) with additional computational cost.

Example: For the data set $(x_0, f_0), (x_1, f_1), (x_2, f_2)$, we have

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & (x_1 - x_0) & 0 \\ 1 & (x_2 - x_0) & (x_2 - x_0)(x_2 - x_1) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \end{bmatrix}.$$

For the data set $(-2, -27), (0, -1), (1, 0)$, we have

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 2 & 0 \\ 1 & 3 & 3 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} -27 \\ -1 \\ 0 \end{bmatrix} \implies [a_0, a_1, a_2] = [-27, 13, -4].$$

Hence $p(x) = -27 + 13(x+2) - 4(x+2)x = -1 + 5x - 4x^2.$