Approximation and Interpolation

**Approximation:** 
If $f$ and $g$ are functions of $x$ defined on interval $[a, b]$ then the inner product of $f$ and $g$ is defined as

$$ (f, g) = \int_a^b f(x)g(x) \, dx $$

Square length of $f$, $(f, f) = \left| f^2 \right| = \int_a^b f^2 \, dx \geq 0$

Length of $f$, $|f| = \sqrt{(f, f)} = \sqrt{\int_a^b f^2 \, dx} \geq 0$

**Least square approximation of a function:**

If $f$ is a function of $x$ is given in $[a, b]$

Let $\phi_0(x), \phi_1(x), \ldots$ be set of base functions (BF) which are independent and complete.

(“Complete” means all terms included as in series expression and any function can be expressed in terms of base functions. A base function can not be expressed in terms of other functions.)

**Example**

$\phi_i(x) = x^i$ (i=0,1,...)

$\phi_0(x) = 1$

$\phi_1(x) = x$

$\phi_2(x) = x^2$, 

etc

Expression of $f$ in terms of BF’s.

$$ f(x) = c_0 \phi_0(x) + c_1 \phi_1(x) + \ldots, \text{ where } c_i \text{’s are constants and } \phi_i(x) \text{ ’s are BF’s.} $$

$$ f(x) = \sum_{i=0}^{\infty} c_i \phi_i(x) \quad (1) $$

infinite sum, exact representation of “$f$” in terms of BF’s.

**Truncation of Eqn. (1)**

$$ f(x) \approx f^t(x) = c_0 \phi_0(x) + c_1 \phi_1(x) + \ldots + c_n \phi_n(x) = \sum_{k=0}^{n} c_k \phi_k(x) \quad \text{or} $$

$$ f^t(x) = \sum_{k=0}^{n} c_k \phi_k $$

approximates “$f$” in $[a, b]$. Where $\phi_k(x)$’s are known and $c_k$’s need to be determined.
**Determination of $c_k$'s:** Square Error or residual $r$ will be minimized to determine $c_k$'s as follows,

$$r(x) = f(x) - f^*(x)$$

So, square length of $r$ is $(r,r)$:

$$I = |r|^2 = (r, r) = \int_a^b r^2 \, dx \rightarrow \text{square error or residual} \quad (2)$$

Determine $c_k$'s so that the square error is minimum

$$\frac{\partial I}{\partial c_i} = 0 \quad (i=0, \ldots, n)$$

$$\frac{\partial I}{\partial c_i} = 0 \Rightarrow \frac{\partial}{\partial c_i} \left( \int_a^b r^2 \, dx \right) = \int_a^b 2r \cdot \frac{\partial r}{\partial c_i} \, dx = 0$$

where $\frac{\partial r}{\partial c_i} = -\frac{\partial f^*}{\partial c_i}$, disregard “-” sign since RHS=0.

$$\int_a^b r^* \frac{\partial f^*}{\partial c_i} \, dx = 0$$

where $\frac{\partial f^*}{\partial c_i} = \phi_i$ since $f^*(x) = c_0 \phi_0(x) + c_1 \phi_1(x) + \ldots + c_n \phi_n(x)$

$$\int_a^b r^* \phi_i \, dx = \int_a^b (f - f^*) \cdot \phi_i \, dx = 0 \Rightarrow \int_a^b f^* \phi_i \, dx = \int_a^b f \phi_i \, dx$$

and $f^*(x) = \sum_{k=0}^{n} c_k \phi_k$

$$\sum_{k=0}^{n} c_k \int_a^b \phi_k \phi_i \, dx = \int_a^b f \phi_i \, dx$$

Here $d_{ik} = \int_a^b \phi_i \phi_k \, dx$ and $p_i = \int_a^b f \phi_i \, dx$

Therefore,

$$\sum_{k=0}^{n} d_{ik} c_k = p_i$$

in matrix form,

$$\begin{bmatrix} d_{00} & \cdots & d_{0n} \\ \vdots & \ddots & \vdots \\ d_{n0} & \cdots & d_{nn} \end{bmatrix} \begin{bmatrix} c_0 \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} p_0 \\ \vdots \\ p_n \end{bmatrix} \quad (3)$$

where $d_{ik} = (\phi_i, \phi_k) = \int_a^b \phi_i \phi_k \, dx$ and $p_i = (f, \phi_i) = \int_a^b f \phi_i \, dx$

Solution of Eqn (3) determines $c_i$'s.

In compact form $\Rightarrow \, D \, c = P$. Then $c = D^{-1} \, P$

Note that $D$ is a symmetric matrix since $d_{ik} = \int_a^b \phi_i \phi_k \, dx = \int_a^b \phi_k \phi_i \, dx = d_{ki}$

Such as $d_{12} = d_{21}$ etc.
Special Case:

If BF’s are orthogonal that means:
\[
\begin{align*}
(\phi_i, \phi_k) &\neq 0 & \text{when } i = k \\
(\phi_i, \phi_k) &\neq 0 & \text{when } i \neq k
\end{align*}
\]

\(D\) would be a diagonal matrix such as
\[
\begin{bmatrix}
d_{00} & \cdots & 0 \\
\vdots & \ddots & \vdots \\
0 & \cdots & d_{nn}
\end{bmatrix}
\begin{bmatrix}
c_0 \\
\vdots \\
c_n
\end{bmatrix}
= 
\begin{bmatrix}
p_0 \\
\vdots \\
p_n
\end{bmatrix}
\rightarrow c_0 = \frac{p_0}{d_{00}},

\begin{align*}
c_1 &= \frac{p_1}{d_{11}}, \ldots, \\
c_n &= \frac{p_n}{d_{nn}}
\end{align*}
\]

LSQ Approximation of a Discrete Form

Discrete data is given as
\[
\begin{array}{c|c}
x & f \\
\hline
x_0 & f_0 \\
x_1 & f_1 \\
\vdots & \vdots \\
x_m & f_m
\end{array}
\]

Base functions are chosen and
\(f(x) \cong f^*(x) = c_0 \phi_0(x) + c_1 \phi_1(x) + \ldots + c_n \phi_n(x)\)

\(c_k\)'s are obtained from
\[
\begin{bmatrix}
d_{00} & \cdots & d_{0n} \\
\vdots & \ddots & \vdots \\
0 & \cdots & d_{nn}
\end{bmatrix}
\begin{bmatrix}
c_0 \\
\vdots \\
c_n
\end{bmatrix}
= 
\begin{bmatrix}
p_0 \\
\vdots \\
p_n
\end{bmatrix}
\]

where
\[
d_{ik} = (\phi_i, \phi_k) = \sum_{p=0}^{m} \phi_i^p \phi_k^p
\]

and
\[
p_i = (f, \phi_i) = \sum_{p=0}^{m} f_p \phi_i^p
\]
in which
\[
\phi_i^p = \phi_i(x_p)
\]

In tabular form:
\[
\begin{array}{cccccccc}
x & f & \phi_0 & \phi_1 & \ldots & \phi_n \\
\hline
x_0 & f_0 & \phi_0(x_0) & x & \ldots & x \\
x_1 & f_1 & x & x & \ldots & x \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
x_m & f_m & \phi_0(x_m) & x & \ldots & x
\end{array}
\]
A Special Case

\( f^* \) is chosen as nth order polynomial, 
\( f(x) \approx f^*(x) = c_0 + c_1 x + \ldots + c_n x^n \)

\[
\begin{array}{cccc}
 x & f & x & x^n \\
 x_0 & f_0 & 1 & x_0^n \\
x_1 & f_1 & 1 & x_1^n \\
\vdots & \vdots & \ddots & \vdots \\
x_m & f_m & 1 & x_m^n \\
\end{array}
\]

\[
Dc = P \Rightarrow c = D^{-1}P, \quad d_{ik} = \sum_{p=0}^{m} x_i^k x_p^k \quad \text{and} \quad p_i = \sum_{p=0}^{m} x_i^p f_p
\]

\[
d_{00} = 1 + 1 + \ldots + 1 = m + 1
\]

\[
d_{01} = 1 \cdot (x_0) + 1 \cdot (x_1) + \ldots + 1 \cdot (x_m) = \sum_{p=0}^{m} x_p = d_{10}
\]

\[
d_{11} = (x_0) \cdot (x_0) + (x_1) \cdot (x_1) + \ldots + (x_m) \cdot (x_m) = \sum_{p=0}^{m} x_p^2
\]

Etc.

\[
p_0 = 1 \cdot (f_0) + 1 \cdot (f_1) + \ldots + 1 \cdot (f_m) = \sum_{p=0}^{m} f_p
\]

\[
p_1 = (x_0) \cdot (f_0) + (x_1) \cdot (f_1) + \ldots + (x_m) \cdot (f_m) = \sum_{p=0}^{m} x_p f_p
\]

Etc.

This is known as Least Square Approximation, LSQ (Curve Fitting)

**EXAMPLE**

We want to approximate the given data by a second-order polynomial.

\( f^*(x) = P_2(x) = c_0 + c_1 x + c_2 x^2 \)

Find \( c_0, c_1 \) and \( c_2 \)

\[
\begin{array}{ccc}
 i & x & f \\
 0 & 0 & 1.016 \\
 1 & 0.2 & 0.768 \\
 2 & 0.4 & 0.648 \\
 3 & 0.7 & 0.401 \\
 4 & 0.9 & 0.272 \\
 5 & 1.0 & 0.193 \\
\end{array}
\]

\[
\sum_{k=0}^{n} d_{ik} c_k = p_i \quad i = 0, \ldots, n \quad \text{and} \quad d_{ik} = \sum_{p=0}^{m} x_i^k x_p^k \quad \text{and} \quad p_i = \sum_{p=0}^{m} x_i^p f_p, \quad \text{where} \ n = 2 \ \text{and} \ m = 5
\]

\[
d_{00} c_0 + d_{01} c_1 + d_{02} c_2 = \sum_{p=0}^{m} f_p \quad \text{for} \ i = 0
\]
\[d_{10}c_0 + d_{11}c_1 + d_{12}c_2 = \sum_{p=0}^{5} x_p f_p \quad \text{for } i=1\]

\[d_{20}c_0 + d_{21}c_1 + d_{22}c_2 = \sum_{p=0}^{5} x_p^2 f_p \quad \text{for } i=2\]

\[
\begin{pmatrix}
\sum_{p=0}^{5} c_0 + \sum_{p=0}^{5} x_p c_1 + \sum_{p=0}^{5} x_p^2 c_2 = \sum_{p=0}^{5} f_p \\
\sum_{p=0}^{5} x_p^2 c_0 + \sum_{p=0}^{5} x_p x_p^2 c_1 + \sum_{p=0}^{5} x_p^2 x_p^2 c_2 = \sum_{p=0}^{5} x_p f_p
\end{pmatrix}
\text{for } i=0
\]

\[
\begin{pmatrix}
6.0 & 3.20 & 2.50 \\
3.20 & 2.50 & 2.144 \\
2.50 & 2.144 & 1.9234
\end{pmatrix}
\begin{pmatrix}
c_0 \\
c_1 \\
c_2
\end{pmatrix}
=\begin{pmatrix}
3.298 \\
1.1313 \\
0.7442
\end{pmatrix}
\]

\[c_0 = 0.9986\]

\[c_1 = -1.006 \quad \text{note that } P_2(x_i) \neq f_i\]

\[c_2 = 0.2100\]

\[P_2(x) = 0.9986 - 1.006x + 0.21x^2\]

**Collocation Method (CM)**

<table>
<thead>
<tr>
<th>x</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_0</td>
<td>f_0</td>
</tr>
<tr>
<td>x_1</td>
<td>f_1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>x_m</td>
<td>f_m</td>
</tr>
</tbody>
</table>

# of data points = m+1
I: \([x_0, x_m]\)

In CM, it is taken that \# of data points = \# of BF’s. \(\phi_0, \phi_1, ..., \phi_m \rightarrow \) Base Functions, BF’s.
- Write \(f(x) \cong f^*(x)\), 
  where \(f^*(x) = c_0\phi_0(x) + c_1\phi_1(x) + ... + c_m\phi_m(x)\)
- then calculate \(c\) values so that \(f^*\) produces the functions values at \(x_0, x_1, ..., x_m\)

\[f^*(x_i) = f_i, \ i = 0, ..., m\]  (2)

From (1) and (2),
\[\phi_0 c_0 + \phi_1 c_1 + ... + \phi_m c_m = f_i\]
\[\phi_i^* = \phi_k(x_i) = a_{ik}\]  (3)

Write (3) at \(x_0, x_1, ..., x_m\) and combine.

\[
\begin{bmatrix}
  a_{00} & \cdots & a_{0m} \\
  \vdots & \ddots & \vdots \\
  a_{m0} & \cdots & a_{mm}
\end{bmatrix}
\begin{bmatrix}
  c_0 \\
  \vdots \\
  c_m
\end{bmatrix}
= \begin{bmatrix}
  f_0 \\
  \vdots \\
  f_m
\end{bmatrix}, \quad \begin{bmatrix}
  a_{00} & \cdots & a_{0m} \\
  \vdots & \ddots & \vdots \\
  a_{m0} & \cdots & a_{mm}
\end{bmatrix}
\begin{bmatrix}
  c_0 \\
  \vdots \\
  c_m
\end{bmatrix}
= \begin{bmatrix}
  f_0 \\
  \vdots \\
  f_m
\end{bmatrix}
\]

\[AC = f \Rightarrow C = A^{-1}f\]  determine \(c\) values.

<table>
<thead>
<tr>
<th>(x)</th>
<th>(f)</th>
<th>(\phi_0)</th>
<th>(\phi_1)</th>
<th>(\phi_m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x_0)</td>
<td>(f_0)</td>
<td>(*)</td>
<td>(*)</td>
<td>(*)</td>
</tr>
<tr>
<td>(x_1)</td>
<td>(f_1)</td>
<td>(*)</td>
<td>(*)</td>
<td>(*)</td>
</tr>
<tr>
<td>(x_m)</td>
<td>(f_m)</td>
<td>(*)</td>
<td>(*)</td>
<td>(*)</td>
</tr>
</tbody>
</table>

**Example for Collocation Method**

For the following given \(x\) and \(f\) values, the second order polynomial is needed to be fitted. So the base functions are \(1, x\) and \(x^2\).

<table>
<thead>
<tr>
<th>(x_i)</th>
<th>(f_i)</th>
<th>(\phi_0(x_i) = 1)</th>
<th>(\phi_1(x_i) = x_i)</th>
<th>(\phi_2(x_i) = x_i^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>31</td>
<td>1</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>121</td>
<td>1</td>
<td>3</td>
<td>9</td>
</tr>
</tbody>
</table>

\(f \cong P_2(x) = c_0 + c_1x + c_2x^2\) in \(1 \leq x \leq 3\)

\[f^*(x_i) = P_2(x_i) = f_i\] and \(\phi_i = \phi_k(x_i) = a_{ik}\)

\[
\begin{bmatrix}
  1 & 1 & 1 \\
  1 & 2 & 4 \\
  1 & 4 & 9
\end{bmatrix}
\begin{bmatrix}
  c_0 \\
  c_1 \\
  c_2
\end{bmatrix}
= \begin{bmatrix}
  5 \\
  32 \\
  121
\end{bmatrix}
\Rightarrow \begin{bmatrix}
  c_0 \\
  c_1 \\
  c_2
\end{bmatrix}
= \begin{bmatrix}
  43 \\
  -70 \\
  32
\end{bmatrix}
\quad \text{then } P_2(x) = 43 - 70x + 32x^2
Lagrange form of an mth order approximated polynomial:

\[
\begin{array}{c|c}
 x_i & f_i \\
 x_0 & f_0 \\
 \vdots & \vdots \\
 x_m & f_m \\
\end{array}
\]

\[f(x) \cong P_m(x) \quad \text{and} \quad f(x_i) = P_m(x_i) \quad \text{and} \quad P_m(x) = L_0(x)f_0 + \cdots + L_m(x)f_m\]

Where \(L_0(x), L_1(x), \ldots, L_m(x)\) are mth order polynomials. And,

\[L_i(x) = \prod_{j=0}^{m} \frac{(x - x_j)}{(x_i - x_j)}.\]

For example, \(L_0(x) = \frac{(x - x_1)(x - x_2)\cdots(x - x_m)}{(x_0 - x_1)(x_0 - x_2)\cdots(x_0 - x_m)}\),

\[L_1(x) = \frac{(x - x_0)(x - x_2)\cdots(x - x_m)}{(x_1 - x_0)(x_1 - x_2)\cdots(x_1 - x_m)}, \ldots \text{ etc.}\]

Note that, \(L_i(x_i) = 1\) and \(L_i(x_j) = 0\) at \(x_0, x_1, \ldots, x_m\) and \(\sum_{i=0}^{m} L_i(x) = 1\) for any \(x\) value.

**Example:**

For the given data, \(P_2(x)\) producing the function values at \(x=1,2,3\) is to be fitted by using Lagrange interpolating polynomial (LIP) (i.e. lagrange formula (LF)).

\[
\begin{array}{c|c|c}
i & x_i & f_i \\
0 & 1 & 5 \\
1 & 2 & 31 \\
2 & 3 & 121 \\
\end{array}
\]

\[P_2(x) = L_0(x)f_0 + L_1(x)f_1 + L_2(x)f_2 \quad \text{or} \quad P_2(x) = L_0(x)(5) + L_1(x)(31) + L_2(x)(121).\]

Where,

\[L_0(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)}, \quad L_1(x) = \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} \quad \text{and} \quad L_2(x) = \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)}\]

\[L_0(x) = \frac{(x - 2)(x - 3)}{(1 - 2)(1 - 3)}, \quad L_1(x) = \frac{(x - 1)(x - 3)}{(2 - 1)(2 - 3)} \quad \text{and} \quad L_2(x) = \frac{(x - 1)(x - 2)}{(3 - 1)(3 - 2)}.\]
P_2(x) = 43 - 70x + 32x^2

Newton Forward Difference Interpolating Polynomial (NFDIP) for P_m(x):

Construction of Difference Table

<table>
<thead>
<tr>
<th>i</th>
<th>0</th>
<th>1</th>
<th>m</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x_0</td>
<td>x_1</td>
<td>...</td>
</tr>
<tr>
<td>f</td>
<td>f_0</td>
<td>f_1</td>
<td>...</td>
</tr>
</tbody>
</table>

It is based on differences in f values. Given values of x and f are equally spaced.

\[ x_2 = x_1 + (x_1 - x_0) \quad \text{let } x_1 - x_0 = h \]
\[ x_2 = x_1 + h \]
\[ x_{i+1} = x_i + h \]

Define \( \Delta f(x) = f(x + h) - f(x) \) or \( \Delta f_i = f(x_i + h) - f(x_i) = f_{i+1} - f_i \)

\( \Delta f_i = f_{i+1} - f_i \) \quad \text{where } \Delta \text{ is called as difference operator.}

\( \Delta(\Delta f_i) = \Delta^2 f_i = \Delta f_{i+1} - \Delta f_i \)
\( \Delta^2 f_i = f_{i+2} - f_{i+1} - (f_{i+1} - f_i) \)
\[ = f_{i+2} - 2f_{i+1} + f_i \]

\( \Delta^3 f_i = f_{i+n} - \binom{n}{1} f_{i+n-1} + \binom{n}{2} f_{i+n-2} + ... + (-1)^n f_i \)

### Difference table

<table>
<thead>
<tr>
<th>x_i</th>
<th>f_i</th>
<th>( \Delta f_i )</th>
<th>( \Delta^2 f_i )</th>
<th>( \Delta^3 f_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_0</td>
<td>f_0</td>
<td>f_1 - f_0</td>
<td>f_1 - 2f_0 + f_0</td>
<td>f_2 - 3f_1 + 3f_0 - f_1</td>
</tr>
<tr>
<td>x_1</td>
<td>f_1</td>
<td>f_2 - f_1</td>
<td>f_2 - 2f_1 + f_0</td>
<td></td>
</tr>
<tr>
<td>x_2</td>
<td>f_2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
NFDIP

\[
\begin{array}{|c|c|c|c|c|}
\hline
x_i & f_i & \Delta f_i & \Delta^2 f_i & \Delta^3 f_i \\
\hline
0 & 1 & & & \\
1 & 5 & 4 & 22 & 42 \\
2 & 31 & 26 & 64 & 66 \\
3 & 121 & 90 & 130 & \\
4 & 341 & 220 & & \\
\hline
\end{array}
\]

\[
P_m(x) = \left(\frac{s}{m+1}\right) \Delta^m f_0, \text{ where } s = \frac{x-x_0}{h} \text{ is nondimensional}
\]

measured from \(x=x_0\). error, \(e=\left(\frac{s}{m+1}\right) \Delta^m f_0 \)

\[
\binom{s}{k} = \frac{s(s-1)(s-2)\cdots(s-k+1)}{k!} \text{ binomial coefficient.}
\]

\[
\begin{align*}
\binom{s}{0} &= 1, \\
\binom{s}{1} &= s, \\
\binom{s}{2} &= \frac{s(s-1)}{2} = \frac{s(s-1)(s-2)}{6} \\
\end{align*}
\]

and \(\Delta f_0, \Delta^2 f_0, \ldots, \Delta^m f_0\) are forward differences at \(x=x_0\).

**Example:**

Where 1,4,24,42,44 are forward differences at \(x=0\) and 5,26,64,66 are forward differences at \(x=1\).

\[
P_m(x) = \left(\frac{s}{m+1}\right) \Delta^m f_0 
\]

a) Find \(P_2(x)\) by using \(x=1,2,3\).

\[
P_2(x) = f_0 + \left(\frac{s}{1}\right) \Delta f_0 + \left(\frac{s}{2}\right) \Delta^2 f_0 \quad \text{or} \\
P_2(x) = f_0 + s\Delta f_0 + \frac{s(s-1)}{2} \Delta^2 f_0. \quad (*)
\]

\[
s = \frac{x-x_0}{h} = \frac{x-1}{1} = x - 1 \quad \text{.} \quad P_2(x) = 43 - 70x + 32x^2
\]

b) Find \(P_3(x)\) by using \(x=0,1,2,3\).

\[
P_3(x) = f_0 + s\Delta f_0 + \frac{s(s-1)}{2} \Delta^2 f_0 + \frac{s(s-1)(s-2)}{3!} \Delta^3 f_0
\]

where \(f_0 = 1\), \(\Delta f_0 = 4\), \(\Delta^2 f_0 = 24\) and \(\Delta^3 f_0 = 42\)

\[
s = \frac{x-x_0}{h} = \frac{x-0}{1} = x \text{ then, } P_3(x) = 1 + 7x - 10x^2 + 7x^3.
\]
c) Find the approximate value of \( f(1.5) \) by using \( x=1, 2, 3 \) and corresponding \( f \) values without finding polynomial explicitly.

\[
f \simeq P_2(x), \quad P_2(x) = f_0 + s\Delta f_0 + \frac{s(s-1)}{2} \Delta^2 f_0, \quad s=x-1=1.5-1=0.5
\]

\[
f(1.5) \simeq P_2(1.5) = 5 + 26s + 64 \frac{s(s-1)}{2} = 10.
\]

Error expression is given by first neglected term in \( P_2(x) = f_0 + \left(\frac{s}{1}\right)\Delta f_0 + \left(\frac{s}{2}\right)\Delta^2 f_0 \)

\[
e(x=1.5) = \left(\frac{s}{3}\right)\Delta^3 f_0 = \frac{s(s-1)(s-2)}{6} = 4.125.
\]

**INTERPOLATING BY SPLINE FUNCTIONS**

Polynomial interpolation for a set of \( N+1 \) points \( \{(x_k, f_k)\} \rightarrow \) is frequently unsatisfactory. To get an satisfactory result, piecewise polynomial interpolation is introduced.

**Simplest case**

Piecewise linear interpolation looks like broken line.

\[
s_k(x) = \frac{x-x_{k+1}}{x_k-x_{k+1}} f_k + \frac{x-x_k}{x_{k+1}-x_k} f_{k+1} \quad \text{for} \quad x_k \leq x \leq x_{k+1}
\]

or \( s_k(x) = f_k + d_k (x-x_k) \) where \( d_k = \frac{(f_{k+1} - f_k)}{(x_{k+1} - x_k)} \)

\[
s(x) = \begin{cases} 
  f_0 + d_0 (x-x_0) & \text{for} \quad x \in [x_0, x_1] \\
  f_1 + d_1 (x-x_1) & \text{for} \quad x \in [x_1, x_2] \\
  \vdots & \\
  f_k + d_k (x-x_k) & \text{for} \quad x \in [x_k, x_{k+1}] \\
  \vdots & \\
  f_{N-1} + d_{N-1} (x-x_{N-1}) & \text{for} \quad x \in [x_{N-1}, x_N] \\
\end{cases}
\]

\[
s(x) = s_k(x) = f_k + d_k (x-x_k) \quad \text{for} \quad x_k \leq x \leq x_{k+1}
\]

- first derivative discontinuous (slope)
- second derivative discontinuous (curvature)
For Piecewise Quadratic Polynomial, 2nd derivative may be discontinuous (i.e. Curvature discontinuous, abrupt changes, undesired bend or distortion at the nodes)

To have a 2nd derivative continuity, Piecewise Cubic Polynomial or Spline should be assumed.

**Piecewise Cubic Spline:**

\[ s(x) = s_k(x) \]

\[ s_k(x) = c_{0,k} + c_{1,k} (x-x_k) + c_{2,k} (x-x_k)^2 + c_{3,k} (x-x_k)^3 \]

for \( x \in [x_k, x_{k+1}] \) for each \( k = 0,1, \ldots, N-1 \)

1. \( s(x_k) = f_k \) for \( k=0,1,\ldots,N \)

2. \( s_k(x_{k+1}) = s_{k+1}(x_{k+1}) \) for \( k=0,1,\ldots,N-2 \) (continuity in the spline curve)

3. \( s_k'(x_{k+1}) = s_{k+1}'(x_{k+1}) \) for \( k=0,1,\ldots,N-2 \) (smooth curve, first derivative continuity (slope))

4. \( s_k''(x_{k+1}) = s_{k+1}''(x_{k+1}) \) for \( k=0,1,\ldots,N-2 \) (curvature continuity)

\( N+1 \rightarrow \) Data points (pairs)

Each \( s_k(x) \) has 4 unknowns (\( c_0, c_1, c_2, c_3 \)) and there are \( N \) \( s_k(x) \), then total 4N unknowns.

\[ s(x_k) = f_k \quad \rightarrow \quad N+1 \text{ equations} \]

\[ s_k(x_{k+1}) = s_{k+1}(x_{k+1}) \quad \rightarrow \quad N-1 \text{ equations} \]

\[ s_k'(x_{k+1}) = s_{k+1}'(x_{k+1}) \quad \rightarrow \quad N-1 \text{ equations} \]

\[ s_k''(x_{k+1}) = s_{k+1}''(x_{k+1}) \quad \rightarrow \quad N-1 \text{ equations} \]

4N-2 equations

Two additional equations are needed, which are called end conditions (or end constraints).

**Finding coefficients of cubic splines:**

\[ s_k(x) = c_{0,k} + c_{1,k} (x-x_k) + c_{2,k} (x-x_k)^2 + c_{3,k} (x-x_k)^3 \]

\[ s(x = x_k) = f_k \rightarrow c_{0,k} = f_k \] (1)

\[ s(x = x_{k+1}) = f_{k+1} \rightarrow c_{0,k} + c_{1,k} h_k + c_{2,k} h_k^2 + c_{3,k} h_k^3 = f_{k+1} \] (2)

\[ s_k''(x) = 2c_{2,k} + 6c_{3,k} (x-x_k) \]

At \( x=x_k \) and \( x=x_{k+1} \)

\[ s_k''(x = x_k) = 2c_{2,k} \] (3)

\[ s_k''(x = x_{k+1}) = 2c_{2,k} + 6c_{3,k} h_k \] (4)
$$c_{2,k} = \frac{s''(x_k)}{2} \quad \text{and} \quad c_{3,k} = \frac{s''(x_{k+1}) - s''(x_k)}{6h_k}$$

From Eqn(2)

$$c_{1,k} = \frac{f_{k+1} - f_k - s''(x_{k+1}) + 2s''(x_k)}{6h_k}$$

$$s_k(x) = f_k + \left[ c_{1,k} + \frac{f_{k+1} - f_k - s''(x_{k+1}) + 2s''(x_k)}{6h_k} \right] s + \frac{s''(x_k) - s''(x_{k+1})}{6h_k}$$  (5)

Where \( s = x - x_k \)

The first derivative of eqn (5) for \( s_k(x_k) \) and \( s_{k-1}(x_k) \) should be equal at \( x = x_k \), then

$$h_{k-1}s''_{k-1}(x_{k-1}) + 2(h_{k-1} + h_k)s''(x_k) + h_k s''(x_{k+1}) = 6 \left[ \frac{1}{h_{k-1}} f_{k-1} - \left( \frac{1}{h_{k-1}} + \frac{1}{h_k} \right) f_k + \frac{1}{h_k} f_{k+1} \right]$$

Where \( h_k = x_{k+1} - x_k \)

Call \( m_{k-1} = s''_{k-1}(x_{k-1}) \), \( m_k = s''(x_k) \), \( m_{k+1} = s''(x_{k+1}) \)

then

$$h_{k-1} m_{k-1} + 2(h_{k-1} + h_k)m_k + h_k m_{k+1} = u_k \quad \text{k=1, ..., N-1}$$

where \( u_k = 6(d_k - d_{k-1}) \), \( d_k = \frac{f_{k+1} - f_k}{h_k} \), \( d_{k-1} = \frac{f_k - f_{k-1}}{h_{k-1}} \)

**End Conditions**

i) "clamped cubic spline" \( s'(x_0) \) and \( s'(x_N) \) \( \rightarrow \) given then

$$m_0 = \frac{3}{h_0} \left[ d_0 - s'(x_0) \right] - \frac{m_1}{2}$$

$$m_N = \frac{3}{h_{N-1}} \left[ s'(x_N) - d_{N-1} \right] - \frac{m_{N-1}}{2}$$

ii) "Natural Cubic Spline" or a "relaxed curve" \( s''(x_0) = 0 \) and \( s''(x_N) = 0 \) \( \quad m_0 = 0 \), \( m_N = 0 \)

iii) Extrapolate \( s'(x_0) \) to the end points

$$m_0 = m_1 - \frac{h_0}{h_1} (m_1 - m)$$

$$m_N = m_{N-1} - \frac{h_{N-1}}{h_{N-2}} (m_{N-1} - m_{N-2})$$

iv) \( s'(x) \) is constant

near the end points \( m_0 = m_1 \), \( m_N = m_{N-1} \)
v) Specify \( s'(x) \) at each end point

\[
m_0 = s'(x_0), \quad m_N = s'(x_N)
\]

**Example:**

End constraints are given as \( m_0 \) and \( m_N \)

\[
h_{k-1} m_{k-1} + 2(h_{k-1} + h_k) m_k + h_k m_{k+1} = u_k
\]

for \( k = 1 \)

\[
2(h_0 + h_1) m_1 + h_1 m_2 - u_0 = 0
\]

for \( k = N-1 \)

\[
h_{N-2} m_{N-2} + 2(h_{N-2} + h_{N-1}) m_{N-1} = u_{N-1} - h_{N-1} m_N
\]

where \( u_k = 6(d_k - d_{k-1}), \quad d_k = \frac{f_{k+1} - f_k}{h_k}, \quad d_{k-1} = \frac{f_k - f_{k-1}}{h_{k-1}}, \quad k = 1, \ldots, N-1 \)

then \( H \mathbf{m} = \mathbf{v} \)

\[
\begin{bmatrix}
  b_1 & c_1 \\
  a_2 & b_2 & c_2 \\
  a_{N-2} & b_{N-2} & c_{N-2} \\
  a_{N-1} & b_{N-1}
\end{bmatrix}
\begin{bmatrix}
  m_1 \\
  m_2 \\
  \vdots \\
  m_{N-2} \n\end{bmatrix}
= \begin{bmatrix}
  v_1 \\
  v_2 \\
  \vdots \\
  v_{N-2} \n\end{bmatrix}
\]

After determination of \( m_k \)’s

\[
c_{0,k} = f_k
\]

\[
c_{1,k} = d_k - \frac{h_k(2m_k + m_{k+1})}{6}
\]

\[
c_{2,k} = \frac{m_k}{2}
\]

\[
c_{3,k} = \frac{m_{k+1} - m_k}{6h_k}
\]

\[
s_k(x) = c_{0,k} + c_{1,k} (x - x_k) + c_{2,k} (x - x_k)^2 + c_{3,k} (x - x_k)^3
\]

**Clamped Spline:** \( s''(a) = s'(x_0) \) and \( s''(b) = s'(x_N) \)

\[
\left( \frac{3}{2} h_0 + 2h \right) m_1 + h m_2 = u_1 - 3[d_0 - s'(x_0)]
\]

\[
h_{k-1} m_{k-1} + 2(h_{k-1} + h_k) m_k + h_k m_{k+1} = u_k \quad \text{for } k = 2, 3, \ldots, N-2
\]

\[
h_{N-2} m_{N-2} + \left( 2h_{N-2} + \frac{3}{2} h_{N-1} \right) m_{N-1} = u_{N-1} - 3[s'(x_N) - d_{N-1}]
\]
Natural Spline: $s''(a) = 0$ and $s''(b) = 0$

\[ 2(h_0 + h_1)m_i + h_im_{i+1} = u_i \]
\[ h_km_{k-1} + 2(h_{k-1} + h_k)m_k + h_km_{k+1} = u_k \quad \text{for } k = 2, 3, \ldots, N - 2 \]
\[ h_{N-2}m_N + 2(h_{N-1} + h_{N-2})m_{N-1} = u_{N-1} \]

Extrapolated Spline: $s''(a)$ and $s''(b)$ extrapolated values by using nodes $x_1, x_2$ and $x_{N-1}, x_{N-2}$ respectively

\[
\begin{align*}
& \left(h_0 + 2h_1 + h_2 \right)m_1 + \left(h_1 - h_2 \right)m_2 = u_1 \\
& h_km_{k-1} + 2(h_{k-1} + h_k)m_k + h_km_{k+1} = u_k \quad \text{for } k = 2, 3, \ldots, N - 2 \\
& \left(h_{N-2} - \frac{h_{N-1}^2}{h_{N-2}} \right)m_{N-2} + \left(2h_{N-1} + 3h_{N-2} + \frac{h_{N-1}^2}{h_{N-2}} \right)m_{N-1} = u_{N-1}
\end{align*}
\]

Parabolically Terminated Spline:

$s''(x) = 0$ on the interval $[x_0, x_i]$ and $[x_{N-1}, x_N]$

\[
\begin{align*}
(3h_0 + 2h_1)m_i + h_im_{i+1} = u_i \\
h_km_{k-1} + 2(h_{k-1} + h_k)m_k + h_km_{k+1} = u_k \quad \text{for } k = 2, 3, \ldots, N - 2 \\
h_{N-2}m_{N-2} + (2h_{N-1} + 3h_{N-2})m_{N-1} = u_{N-1}
\end{align*}
\]

Endpoint Curvature - Adjusted Spline:

$s''(a)$ and $s''(b)$ are specified.

\[
\begin{align*}
& 2(h_0 + h_1)m_i + h_im_{i+1} = u_i - h_is''(x_i) \\
& h_km_{k-1} + 2(h_{k-1} + h_k)m_k + h_km_{k+1} = u_k \quad \text{for } k = 2, 3, \ldots, N - 2 \\
& h_{N-2}m_N + 2(h_{N-1} + h_{N-2})m_{N-1} = u_{N-1} - h_{N-1}s''(x_N)
\end{align*}
\]

Example:

Find clamped cubic spline that passes through $(0,0)$, $(1,0.5)$, $(2,2)$, $(3,1.5)$ with the boundary conditions $s'(0) = 0.2$ and $s'(3) = -1$

<table>
<thead>
<tr>
<th></th>
<th>(0,0)</th>
<th>(1,0.5)</th>
<th>(2,2)</th>
<th>(3,1.5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_0$</td>
<td>1</td>
<td>1</td>
<td>0.5</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Solution:

\[
h_{k-1}m_{k-1} + 2(h_{k-1} + h_k)m_k + h_km_{k+1} = u_k \quad \text{for } k = 1, \ldots, N-1
\]

where $u_k = 6(d_k - d_{k-1})$, $d_k = \frac{f_{k+1} - f_k}{h_k}$, $d_{k-1} = \frac{f_k - f_{k-1}}{h_{k-1}}$

\[
h_0=h_1=h_2=1, d_0 = \frac{f_1 - f_0}{h_0} = \frac{0.5 - 0.0}{1} = 0.5
\]
\[ d_1 = \frac{f_2 - f_1}{h_1} = \frac{2.0 - 0.5}{1} = 1.5 \]
\[ d_2 = \frac{f_3 - f_2}{h_2} = \frac{1.5 - 2.0}{1} = -0.5 \]
\[ u_1 = 6(d_1 - d_o) = 6.0(1.5 - 0.5) = 6.0 \]
\[ u_2 = 6(d_2 - d_1) = 6.0(-0.5 - 1.5) = -12.0 \]
\[ m_0 = \frac{3}{h_0} [d_0 - s'(x_0)] - \frac{m_1}{2} \]
\[ m_N = \frac{3}{h_{N-1}} [s'(x_N) - d_{N-1}] - \frac{m_{N-1}}{2} \]
\[ \left( \frac{3}{2} + 2 \right) m_1 + m_2 = 6.0 - 3.0[0.5 - 0.2] = 5.1 \]
\[ m_1 + \left( 2 + \frac{3}{2} \right) m_2 = -12.0 - 3.0[-1.0 - (-0.5)] = -10.5 \]

\[
\begin{bmatrix}
3.5 \\
1.0
\end{bmatrix}
\begin{bmatrix}
m_1 \\
m_2
\end{bmatrix}
= 
\begin{bmatrix}
5.1 \\
-10.5
\end{bmatrix} \Rightarrow 
\begin{align*}
m_1 &= 2.52 \\
m_2 &= -3.72
\end{align*}
\]
then \( m_0 \) and \( m_3 \) from table (equations for the clamped case)
\[ m_0 = 3(0.5 - 0.2) - \frac{2.52}{2} = -0.36 \]
\[ m_3 = 3(-1.0 + 0.5) - \frac{3.72}{2} = 0.36 \]
\[ m_0, m_1, m_2, m_3 \Rightarrow c_{0,k}, c_{1,k}, c_{2,k}, c_{3,k} \]

\[ c_{0,k} = f_k \]
\[ c_{1,k} = \frac{h_k (2m_k + m_{k+1})}{6} \]
\[ c_{2,k} = \frac{m_k}{2} \]
\[ c_{3,k} = \frac{m_{k+1} - m_k}{6h_k} \]
\[ s_k(x) = c_{0,k} + c_{1,k} (x - x_k) + c_{2,k} (x - x_k)^2 + c_{3,k} (x - x_k)^3 \]
\( s_0(x) = 0.48x^3 - 0.18x^2 + 0.2x \quad 0 \leq x \leq 1 \)
\( s_1(x) = -1.04(x-1)^3 + 1.26(x-1)^2 \\
+ 1.28(x-1) + 0.5 \quad 1 \leq x \leq 2 \)
\( s_2(x) = 0.68(x-2)^3 - 1.86(x-2)^2 \\
+ 0.68(x-2) + 2.0 \quad 2 \leq x \leq 3 \)