**Interpolation and Approximation**[1]

***The problem:***

Given a set of discrete data $\mathbf{S} = \{(\mathbf{x_i}, \mathbf{y_i}) \,|\, \mathbf{i} = 0,1,2,\ldots,\mathbf{n}\}$ determine a function **y = P(x)** such that, $\mathbf{P(x_i) = y_i}$ for i = 0, 1, 2, …, n.
*(Note there are n +1 points in set S.)*

- If we can find such a function we say **P(x) interpolates the data set S**.
- **P(x) is called an interpolant** to the data set S.
- We say the data set **S** consists of **distinct points** if the x-coordinates are all different from one another. That $\mathbf{x_i} \neq \mathbf{x_j}$, whenever $\mathbf{i} \neq \mathbf{j}$.
- The function **y = P(x)** is said to be an approximation to the function from which the discrete sample in set **S** was obtained.

Types of interpolants include,

- Polynomials
- Rational functions
- Piecewise polynomials
- Splines (special piecewise polynomials)

**Case of polynomial interpolants.**

*Hunting license.*

Weierstrass Approximation Theorem

For function **y = f(x)** in C[a, b], and any $\varepsilon$>0, there exists a polynomial **p(x)** such that $\max_{x \varepsilon [a,b]} |f(x) - p(x)| < \varepsilon$.

Notes:

1. This is an existence theorem, meaning such a polynomial p(x) exists and is "close" to f(x).
2. The proof is not truly constructive, in the sense that it doesn't tell us how to build p(x) and it doesn't tell us the degree of p(x).

*Taylor Polynomials*

We have seen that Taylor polynomials can provide an approximation to a sufficiently differentiable function around the center of expansion. However, often as we move away from then center of expansion the Taylor polynomial's accuracy deteriorates. In a sense a Taylor polynomial does interpolate. It matches the value of the function and its derivatives at the center of expansion.

[1] Interpolation_and_approximation.doc 10/8/2005

**General Polynomial Interpolation Theorem**

For a distinct set $\mathbf{S} = \{(\mathbf{x_i}, \mathbf{y_i}) \,|\, \mathbf{i} = 0,1,2,\ldots,\mathbf{n}\}$ there **exists** a **unique** polynomial $\mathbf{P_n(x)}$ of degree n or less that interpolates the data in **S**; that is, $\mathbf{P_n(x_i)} = \mathbf{y_i}$ for $\mathbf{i} = 0,1,2,\ldots,\mathbf{n}$.

*Discussion:*

The set of all polynomials of degree n or less, denote $\mathscr{P}$**n**, is a vector space. It has various bases that can be used to express the interpolating polynomial. We describe three bases here. Each will play a role in our development.

- The **standard basis** is $\mathbf{B} = \{1, \mathbf{x}, \mathbf{x}^2, \ldots, \mathbf{x}^{\mathbf{n}}\}$. Note that the basis members are each of a different degree and do not depend on the data set **S**.

- The **Lagrange basis** is the set $\mathbf{L_i(x)},\ \mathbf{i} = 0,1,2,\ldots\mathbf{n}$ given by the following expressions which show dependence on the data **S**;
$$\mathbf{L_i(x)} = \frac{(\mathbf{x}-\mathbf{x}_0)(\mathbf{x}-\mathbf{x}_1)\cdots(\mathbf{x}-\mathbf{x}_{\mathbf{i}-1})(\mathbf{x}-\mathbf{x}_{\mathbf{i}+1})\cdots(\mathbf{x}-\mathbf{x_n})}{(\mathbf{x_i}-\mathbf{x}_0)(\mathbf{x_i}-\mathbf{x}_1)\cdots(\mathbf{x_i}-\mathbf{x}_{\mathbf{i}-1})(\mathbf{x_i}-\mathbf{x}_{\mathbf{i}+1})\cdots(\mathbf{x_i}-\mathbf{x_n})}$$
Note that each of the Lagrange basis members are of degree n and satisfy the property $\mathbf{L_i}(\mathbf{x_j}) = \begin{cases} 1, & \text{when } i = j \\ 0, & \text{when } i \neq j \end{cases}$.

- The **divided difference basis** is the set
$$\begin{aligned}
\mathbf{D_0(x)} &= 1 \\
\mathbf{D_1(x)} &= (\mathbf{x}-\mathbf{x}_0) \\
\mathbf{D_2(x)} &= (\mathbf{x}-\mathbf{x}_0)(\mathbf{x}-\mathbf{x}_1) \\
&\vdots \\
\mathbf{D_n(x)} &= (\mathbf{x}-\mathbf{x}_0)(\mathbf{x}-\mathbf{x}_1)\cdots(\mathbf{x}-\mathbf{x}_{\mathbf{n}-1})
\end{aligned}$$

  Note that the members of the divided difference basis depend on the data set and they are of different degrees.

**Proof of the General Polynomial Interpolation Theorem**

Using standard basis, let $\mathbf{P_n(x)} = \mathbf{a}_0 + \mathbf{a}_1\mathbf{x} + \mathbf{a}_2\mathbf{x}^2 + \cdots + \mathbf{a_n x^n}$. Successively set $\mathbf{x} = \mathbf{x_i}$ and then we require $\mathbf{P_n(x_i)} = \mathbf{a}_0 + \mathbf{a}_1\mathbf{x_i} + \mathbf{a}_2\mathbf{x_i}^2 + \cdots + \mathbf{a_n x_i^n} = \mathbf{y_i}$. This leads to a system of n + 1 equations in n + 1 unknowns and this process is called the method of undetermined coefficients. The linear system is

$$\begin{bmatrix} 1 & \mathbf{x}_0 & \mathbf{x}_0^2 & \cdots & \mathbf{x}_0^{\mathbf{n}} \\ 1 & \mathbf{x}_1 & \mathbf{x}_1^2 & \cdots & \mathbf{x}_1^{\mathbf{n}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \mathbf{x_n} & \mathbf{x_n}^2 & \cdots & \mathbf{x_n}^{\mathbf{n}} \end{bmatrix} \begin{bmatrix} \mathbf{a}_0 \\ \mathbf{a}_1 \\ \vdots \\ \vdots \\ \mathbf{a_n} \end{bmatrix} = \begin{bmatrix} \mathbf{y}_0 \\ \mathbf{y}_1 \\ \vdots \\ \vdots \\ \mathbf{y_n} \end{bmatrix}$$

The coefficient matrix is called a Vandermonde matrix and can be shown to be nonsingular for distinct data sets $\mathbf{S} = \{(\mathbf{x_i}, \mathbf{y_i}) \,|\, \mathbf{i} = 0, 1, 2, \ldots, \mathbf{n}\}$. Hence there exists a unique interpolant to the data set **S**.

Note: The method of undetermined coefficients is not restricted to use with interpolating polynomials. For instance, it can be used to construct derivative approximations and approximations to definite integrals.

## ***Constructing Interpolation Polynomials***

The method of undetermined coefficients is generally not used to construct interpolation polynomial. This is because it requires the solution of a linear system whose coefficient matrix, a Vandermonde matrix, tends to be unstable in the Gaussian elimination process especially for large data sets. In addition, if we change the data by adjoining new points or deleting points the entire process must be repeated. Note that the coefficient matrix depends on the x-coordinates of the data set.

*Using the Lagrange basis*

An alternative to the method of undetermined coefficients employs the Lagrange basis. In this case we avoid the solution of a linear system. The construction of the members of the Lagrange basis is easy for small data sets, but becomes cumbersome for large data sets. Once we have the Lagrange basis $\mathbf{L}_0(\mathbf{x}), \mathbf{L}_1(\mathbf{x}), \ldots, \mathbf{L_n(x)}$ for a data set $\mathbf{S} = \{(\mathbf{x_i}, \mathbf{y_i}) \,|\, \mathbf{i} = 0, 1, 2, \ldots, \mathbf{n}\}$ the interpolating polynomial is constructed as a linear combination of the basis members using the y-coordinates of the data set as coefficients.

$$\mathbf{P_n(x)} = \sum_{\mathbf{k}=0}^{\mathbf{n}} \mathbf{y_k} * \mathbf{L_k(x)} = \mathbf{y}_0 * \mathbf{L}_0(\mathbf{x}) + \mathbf{y}_1 * \mathbf{L}_1(\mathbf{x}) + \cdots + \mathbf{y_n} * \mathbf{L_n(x)}.$$

If the data set is expressed in terms of points on a function f(x) as $\mathbf{S}=\{(\mathbf{x_i},\mathbf{f}(\mathbf{x_i}))\,|\,\mathbf{i}=0,1,2,\ldots,\mathbf{n}\}$ then the interpolating polynomial using the Lagrange basis is given by

$$\mathbf{P_n}(\mathbf{x})=\sum_{\mathbf{k}=0}^{\mathbf{n}}\mathbf{f}(\mathbf{x_k})*\mathbf{L_k}(\mathbf{x})=\mathbf{f}(\mathbf{x}_0)*\mathbf{L}_0(\mathbf{x})+\mathbf{f}(\mathbf{x}_1)*\mathbf{L}_1(\mathbf{x})+\cdots+\mathbf{f}(\mathbf{x_n})*\mathbf{L_n}(\mathbf{x}).$$

**Example** Construct an interpolating polynomial to $\mathbf{f}(\mathbf{x})=\sqrt{\mathbf{x}}$ at $\mathbf{x}_0=0,\ \mathbf{x}_1=1,\ \mathbf{x}_2=4.$

Solution process:

- Determine the corresponding function values,
  $\mathbf{f}(\mathbf{x}_0)=\sqrt{0}=0,\ \mathbf{f}(\mathbf{x}_1)=\sqrt{1}=1,\ \mathbf{f}(\mathbf{x}_2)=\sqrt{4}=2.$
- Construct the Lagrange basis for the data set $\mathbf{S}=\{(0,0),(1,1),(4,2)\}$.

$$\mathbf{L}_0(\mathbf{x})=\frac{(\mathbf{x}-\mathbf{x}_1)(\mathbf{x}-\mathbf{x}_2)}{(\mathbf{x}_0-\mathbf{x}_1)(\mathbf{x}_0-\mathbf{x}_2)}=\frac{(\mathbf{x}-1)(\mathbf{x}-4)}{(0-1)(0-4)}=\frac{1}{4}\left(\mathbf{x}^2-5\mathbf{x}+4\right)$$

$$\mathbf{L}_1(\mathbf{x})=\frac{(\mathbf{x}-\mathbf{x}_0)(\mathbf{x}-\mathbf{x}_2)}{(\mathbf{x}_1-\mathbf{x}_0)(\mathbf{x}_1-\mathbf{x}_2)}=\frac{(\mathbf{x}-0)(\mathbf{x}-4)}{(1-0)(1-4)}=\frac{-1}{3}\left(\mathbf{x}^2-4\mathbf{x}\right)$$

$$\mathbf{L}_2(\mathbf{x})=\frac{(\mathbf{x}-\mathbf{x}_0)(\mathbf{x}-\mathbf{x}_1)}{(\mathbf{x}_2-\mathbf{x}_0)(\mathbf{x}_2-\mathbf{x}_1)}=\frac{(\mathbf{x}-0)(\mathbf{x}-1)}{(4-0)(4-1)}=\frac{1}{12}\left(\mathbf{x}^2-\mathbf{x}\right)$$

- Construct the interpolating polynomial as

$$\mathbf{P}_2(\mathbf{x})=\sum_{\mathbf{k}=0}^{2}\mathbf{f}(\mathbf{x_k})*\mathbf{L_k}(\mathbf{x})=\mathbf{f}(\mathbf{x}_0)*\mathbf{L}_0(\mathbf{x})+\mathbf{f}(\mathbf{x}_1)*\mathbf{L}_1(\mathbf{x})+\mathbf{f}(\mathbf{x}_2)*\mathbf{L}_2(\mathbf{x})$$

$$=0*\mathbf{L}_0(\mathbf{x})+1*\mathbf{L}_1(\mathbf{x})+2*\mathbf{L}_2(\mathbf{x})=\frac{1}{6}\left(-\mathbf{x}^2+7\mathbf{x}\right)$$

Let's use the interpolating polynomial to estimate values of **f(x)** at x = 2. 3, and 5. We get

$$\mathbf{P}_2(2)=\frac{10}{6}\approx 1.6667,\qquad \mathbf{f}(2)=\sqrt{2}\approx 1.4142$$

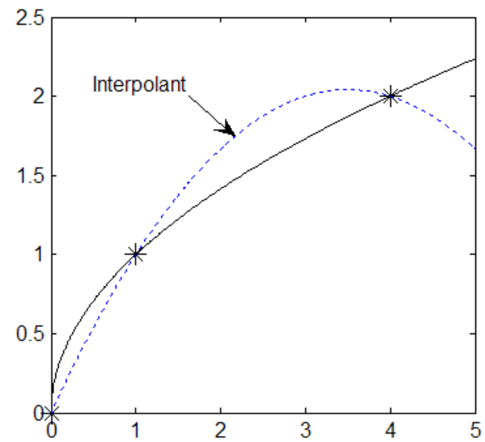
$$\mathbf{P}_2(3)=2,\qquad \mathbf{f}(3)=\sqrt{3}\approx 1.7321$$

$$\mathbf{P}_2(5)=\frac{10}{6}\approx 1.6667,\qquad \mathbf{f}(5)=\sqrt{5}\approx 2.2361$$

This interpolating polynomial doesn't provide a highly accurate approximation to $\mathbf{f}(\mathbf{x})=\sqrt{\mathbf{x}}$ . This example points out a property of interpolating polynomials.

Interpolating polynomials are constructed to go through a data set S, but they need not be close at other points to the function from which the set S is derived.

Finally let's look at the graph of $\mathbf{P_2(x)}$ compared to $\mathbf{f(x)} = \sqrt{\mathbf{x}}$ .




++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

*Remark:* While using the Lagrange basis avoids the solution of a linear system, altering the data set S by deleting or adjoining points requires the re-computation of the basis members.

<u>MATLAB Notes:</u>

1.Routine **sqrtdemo** provides a graphical look at the preceding example and displays a graph of the $\sqrt{\mathbf{x}} - \mathbf{P_2(x)}$.

2. As we remarked above, the Lagrange basis members have the property $\mathbf{L_i}\left(\mathbf{x_j}\right) = \begin{cases} 1, \text{ when } i = j \\ 0, \text{ when } i \neq j \end{cases}$. This means that the graph of $\mathbf{y} = \mathbf{L_i(x)}$ has height 1 at $\mathbf{x = x_i}$, but has an x-intercept at all the other points in S. So that $\mathbf{y} = \mathbf{L_i(x)}$ is a polynomial of degree n that interpolates the data set

$$\mathbf{T_i} = \left\{\left(\mathbf{x_i},1\right), \left(\mathbf{x_j},0\right) \text{for } \mathbf{j} = 0,1,2,\mathbf{...,n}, j \neq i\right\}$$

To see a graphical demonstration of this and the construction of the interpolant as a linear combination of the Lagrange basis use routine **lagrang2**.

3. Routine **lagrang5** lets you enter a function and an interval. It construct the Lagrange basis for 5 equispaced points, forms their linear combination to get the interpolant, and also compares the interpolant's accuracy with that of the Taylor polynomial of degree 4 with center of expansion the midpoint of the interval.

4. Routine **interpit** lets you choose a data set using the mouse, construct the interpolant, and see its graph.

*Application of the Lagrange Basis*

Approximate $\int_{-1}^{1} \mathbf{f(x)\ dx}$ by constructing the Lagrange interpolant $\mathbf{P_2(x)}$ to the data set $\mathbf{S} = \{(-1, \mathbf{f(-1)})(0, \mathbf{f(0)}), (1, \mathbf{f(1)})\}$ and integrating $\mathbf{P_2(x)}$; that is,

$$\int_{-1}^{1} \mathbf{f(x)\ dx} \approx \int_{-1}^{1} \mathbf{P_2(x)\ dx}.$$

Solution process:

- Construct the interpolant.

$$\mathbf{P_2(x)} = \mathbf{f(-1)}\frac{\mathbf{(x-0)(x-1)}}{\mathbf{(-1)(-2)}} + \mathbf{f(0)}\frac{\mathbf{(x+1)(x-1)}}{\mathbf{(1)(-1)}} + \mathbf{f(1)}\frac{\mathbf{(x+1)(x-0)}}{\mathbf{(2)(1)}}$$
$$= \frac{1}{2}\mathbf{f(-1)(x^2-x)} - \mathbf{f(0)(x^2-1)} + \frac{1}{2}\mathbf{f(1)(x^2+x)}$$

- Integrate the interpolant.

$$\int_{-1}^{1} \mathbf{P_2(x)\ dx} = \frac{1}{2}\mathbf{f(-1)}\int_{-1}^{1}\mathbf{(x^2-x)\ dx} - \mathbf{f(0)}\int_{-1}^{1}\mathbf{(x^2-1)\ dx} + \frac{1}{2}\mathbf{f(1)}\int_{-1}^{1}\mathbf{(x^2+x)\ dx}$$
$$= \frac{1}{2}\mathbf{f(-1)}\left[\frac{\mathbf{x}^3}{3} - \frac{\mathbf{x}^2}{2}\right]_{-1}^{1} - \mathbf{f(0)}\left[\frac{\mathbf{x}^3}{3} - \mathbf{x}\right]_{-1}^{1} + \frac{1}{2}\mathbf{f(1)}\left[\frac{\mathbf{x}^3}{3} + \frac{\mathbf{x}^2}{2}\right]_{-1}^{1}$$
$$= \frac{1}{3}\mathbf{f(-1)} + \frac{4}{3}\mathbf{f(0)} + \frac{1}{3}\mathbf{f(1)} = \frac{1}{3}\left(\mathbf{f(-1)} + 4\mathbf{f(0)} + \mathbf{f(1)}\right)$$

The integral of the interpolant provides a numerical integration formula, also called a quadrature formula, that can be used to approximate $\int_{-1}^{1} \mathbf{f(x)\ dx}$. The formula just developed is called the (basic) **Simpson's Rule**. Simpson's Rule is not restricted to the interval [-1, 1]. It can be extended to approximate $\int_{\mathbf{a}}^{\mathbf{b}} \mathbf{f(x)\ dx}$ where $\mathbf{x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b}$. In addition it can be used as a composite rule where interval [a, b] is divided into an even number of subintervals.

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

*Using the Divided Difference Basis*

The Lagrange form of the interpolating polynomial is advantageous for certain applications, but has a major difficulty.

> If a point is adjoined to the data set or if a point is deleted from the data set the entire set of basis members must be recomputed.

Here we describe the **'divided difference basis'**, which is designed to have an **add-on feature**. That is, if a new point is adjoined to the data set the new interpolation polynomial is obtained by adding a new term to the old interpolation polynomial. It also has the property that if the last point adjoined is deleted, a term can be deleted to obtain the interpolant for the smaller data set. These properties make the divided difference basis attractive for a variety of interpolation situations.

The interpolating polynomial to a set $\mathbf{S} = \{(\mathbf{x_i}, \mathbf{y_i}) \,|\, \mathbf{i} = 0, 1, 2, \ldots, \mathbf{n}\}$ of distinct points is constructed starting with a polynomial of degree zero that is just the y-coordinate of the first data point of S. Next we add a term so that we get a polynomial of degree one through the first two data. We continue in this fashion adjoining a point and adding on a term so that the interpolant goes through the previous points in addition to the adjoined new point. Conceptually this is an easy idea, the only computational expense is to get the coefficients of the terms that are added on. To get the coefficients it is helpful to use a device called a divided difference table, a portion is displayed next.

$$
\begin{array}{llllll}
 & \text{0-thDD} & \text{1-stDD} & \text{2-ndDD} & \text{3-rdDD} \\
x_0 & f(x_0) & & & \\
 & & f[x_1,x_0]=\dfrac{f(x_1)-f(x_0)}{x_1-x_0} & & \\
x_1 & f(x_1) & & f[x_2,x_1,x_0]=\dfrac{f[x_2,x_1]-f[x_1,x_0]}{x_2-x_0} & \\
 & & f[x_2,x_1]=\dfrac{f(x_2)-f(x_1)}{x_2-x_1} & & f[x_3,x_2,x_1,x_0]=\dfrac{f[x_3,x_2,x_1]-f[x_2,x_1,x_0]}{x_3-x_0} \\
x_2 & f(x_2) & & f[x_3,x_2,x_1]=\dfrac{f[x_3,x_2]-f[x_2,x_1]}{x_3-x_1} & \\
 & & f[x_3,x_2]=\dfrac{f(x_3)-f(x_2)}{x_3-x_2} & & \\
x_3 & f(x_3) & & &
\end{array}
$$

Computationally we initialize the table with the first two columns from our data set. The first column contains the x-values and the second the y-values. The y-values are labeled 0-thDD, for $0^{th}$ order divided difference. Then we compute the column of $1^{st}$ order divided differences. Note that $1^{st}$ order DDs are just slopes between successive data points. We then proceed to compute the higher order DDs in a recursive format. For points labeled 0, 1, ..., n, there will be $0^{th}$ DD through $n^{th}$ DD. The last column of the table will contain the single entry, $\mathbf{f[x_n, x_{n-1}, \ldots, x_1, x_0]}$, an $n^{th}$ order DD.

The interpolating polynomial for the divided difference table above is

$$P_3(x) = f[x_0] + f[x_1, x_0](x - x_0)$$
$$+ f[x_2, x_1, x_0](x - x_0)(x - x_1) + f[x_3, x_2, x_1, x_0](x - x_0)(x - x_1)(x - x_2)$$

| |
|---|
| **Note that the coefficients for the divided difference basis are obtained from the top diagonal of the table.** |

We have coefficients

$$f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$
$$f[x_0, x_1, x_2] = \frac{f[x_2, x_1] - f[x_1, x_0]}{x_2 - x_0}$$
$$\vdots$$
$$f[x_0, x_1, x_2, \ldots, x_n] = \frac{f[x_n, \ldots x_2, x_1] - f[x_{n-1}, \ldots x_1, x_0]}{x_n - x_0}$$

It more difficult to shown general expressions for items in the divided difference table than it is do actually compute them from numerical data. We illustrate this in the next example.

**<u>Example</u>** Construct the DD table for data S = {(3,-2), (5,4), (1,0), (2,3)}. Fill in the missing positions in the table.

$$\begin{array}{cccccc}
 & \text{0 - th DD} & \text{1 - st DD} & 2 - \text{nd DD} & \text{3 - rd DD} \\
3 & -2 & & & \\
 & & \dfrac{4-(-2)}{5-3} = 3 & & \\
5 & 4 & & _____ & \\
 & & ______ & & _____ \\
1 & 0 & & _____ & \\
 & & ______ & & \\
2 & 3 & & &
\end{array}$$

On the line below write out the expression for the interpolating polynomial[2] to set S using the DD basis.

$P_3(x)$=_____________________________________________________

To check that your computations are correct compute $P_3(3)$, $P_3(5)$, $P_3(1)$, $P_3(2)$.
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

[2] You should find $P_3(x) = -2 + 3\ (x - 3) + 1\ \ (x - 3)\ (x - 5) + (5/3)\ \ (x - 3)\ (x - 5)\ (x - 1)$

We have a routine in MATLAB that will compute a divided difference table for data. Use **help divdiff** to get directions. The display generated for the preceding example is

| | | | | |
|---|---|---|---|---|
| 3.0000 | -2.0000 | 3.0000 | 1.0000 | 1.6667 |
| 5.0000 | 4.0000 | 1.0000 | -0.6667 | 0 |
| 1.0000 | 0 | 3.0000 | 0 | 0 |
| 2.0000 | 3.0000 | 0 | 0 | 0 |

or in rational form

| | | | | |
|---|---|---|---|---|
| 3 | -2 | 3 | 1 | 5/3 |
| 5 | 4 | 1 | -2/3 | 0 |
| 1 | 0 | 3 | 0 | 0 |
| 2 | 3 | 0 | 0 | 0 |

<u>The coefficients for the interpolating polynomial are in the top row starting with the second column.</u> Rather than a triangular table display as used in the development above, the MATLAB display is rectangular for ease in programming.

The next example shows how to adjoin a point to the data set and use the add-on feature to get the new interpolant.

**Example** Suppose we have constructed the divided difference table for data set S = {(3,-2), (5,4), (1,0), (2,3)}. We get the table shown below.

| **x** | 0-th DD | 1-st DD | 2 – nd DD | 3-rd DD |
|---|---|---|---|---|
| 3 | $-2$ | | | |
| | | 3 | | |
| 5 | 4 | | 1 | |
| | | 1 | | $\frac{5}{3}$ |
| 1 | 0 | | $\frac{-2}{3}$ | |
| | | 3 | | |
| 2 | 3 | | | |

If we want to adjoin point (6,2) to the data set, we extend the table as follows. Put 6 in the x-column and 2 in the $0^{th}$ DD column, then compute terms at the bottom of each of the $1^{st}$, $2^{nd}$, and $3^{rd}$ DD columns. There will be a new column, label $4^{th}$ DD for which we compute the coefficient of the new term that is to be adjoin to the cubic polynomial that interpolates the first four points.

| **x** | $0-$**th DD** | $1-$**st DD** | $2-$**nd DD** | $3-$**rd DD** | $4-$**th DD** |
|---|---|---|---|---|---|
| 3 | $-2$ | | | | |
| | | 3 | | | |
| 5 | 4 | | 1 | | |
| | | 1 | | $\frac{5}{3}$ | |
| 1 | 0 | | $\frac{-2}{3}$ | | $\boxed{\frac{1/60-5/3}{6-3}=\frac{-11}{20}}$ |
| | | 3 | | $\boxed{\frac{-13/60-(-2/3)}{6-5}=\frac{1}{60}}$ | |
| 2 | 3 | | $\boxed{\frac{-1/4-3}{6-1}=\frac{-13}{20}}$ | | |
| | | $\boxed{\frac{2-3}{6-2}=\frac{-1}{4}}$ | | | |
| $\boxed{6}$ | $\boxed{2}$ | | | | |

The interpolant for the five points is

$$\mathbf{P_4(x)} = \mathbf{P_3(x)} + \frac{-11}{20}(\mathbf{x}-3)(\mathbf{x}-5)(\mathbf{x}-1)(\mathbf{x}-2)$$

$$= -2 + 3(\mathbf{x}-3) + 1(\mathbf{x}-3)(\mathbf{x}-5) + \frac{5}{3}(\mathbf{x}-3)(\mathbf{x}-5)(\mathbf{x}-1) + \frac{-11}{20}(\mathbf{x}-3)(\mathbf{x}-5)(\mathbf{x}-1)(\mathbf{x}-2)$$

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

<u>MATLAB Notes:</u>

1. Routine **divdiff** will compute a divided difference table. Use **help divdiff** for information on using this routine.

2. Routine **divpoly** will determine an expression for a divided difference interpolation polynomial. Use **help divpoly** for information on using this routine.

3. Routine **pinterp** compute an interpolation polynomial. It includes graphical options. Use **help pinterp** for information on using this routine.