Interpolation & Polynomial Approximation

Data Approximation & Neville's Method

Numerical Analysis (9th Edition)

R L Burden & J D Faires

Beamer Presentation Slides
prepared by
John Carroll
Dublin City University

Outline

1. Interpolation with access to function values only

Outline

1. Interpolation with access to function values only
2. Introduction to Neville's Method

Outline

1. Interpolation with access to function values only
2. Introduction to Neville's Method
3. Applying Neville's Method to Tabulated Data

Outline

Outline

Outline

1. Interpolation with access to function values only
2. Introduction to Neville's Method
3. Applying Neville's Method to Tabulated Data
4. Applying Neville's Method to Tabulated Data with 4-digit Rounding Arithmetic
5. Neville's Iterated Interpolation Algorithm

Interpolating Accuracy without underlying $f(x)$

Context

Interpolating Accuracy without underlying $f(x)$

Context

- A frequent use of interpolating polynomials involves the interpolation of tabulated data.

Interpolating Accuracy without underlying $f(x)$

Context

- A frequent use of interpolating polynomials involves the interpolation of tabulated data.
- In this case, an explicit representation of the polynomial might not be needed, only the values of the polynomial at specified points.

Interpolating Accuracy without underlying $f(x)$

Context

- A frequent use of interpolating polynomials involves the interpolation of tabulated data.
- In this case, an explicit representation of the polynomial might not be needed, only the values of the polynomial at specified points.
- In this situation the function underlying the data might not be known so the explicit form of the error cannot be used.

Interpolating Accuracy without underlying $f(x)$

Context

- A frequent use of interpolating polynomials involves the interpolation of tabulated data.
- In this case, an explicit representation of the polynomial might not be needed, only the values of the polynomial at specified points.
- In this situation the function underlying the data might not be known so the explicit form of the error cannot be used.
- The following example illustrates a practical application of interpolation in such a situation.

Interpolating Accuracy without underlying $f(x)$

Example: Tabulated Data

The following table

x	1.0	1.3	1.6	1.9	2.2
$f(x)$	0.7651977	0.6200860	0.4554022	0.2818186	0.1103623

lists values of a function f at various points. The approximations to $f(1.5)$ obtained by various Lagrange polynomials that use this data will be compared to try and determine the accuracy of the approximation.

Interpolating Accuracy without underlying $f(x)$

x	1.0	1.3	1.6	1.9	2.2
$f(x)$	0.7651977	0.6200860	0.4554022	0.2818186	0.1103623

Solution (1/7)

Interpolating Accuracy without underlying $f(x)$

x	1.0	1.3	1.6	1.9	2.2
$f(x)$	0.7651977	0.6200860	0.4554022	0.2818186	0.1103623

Solution (1/7)

The most appropriate linear polynomial uses $x_0 = 1.3$ and $x_1 = 1.6$ because 1.5 is between 1.3 and 1.6.

Interpolating Accuracy without underlying $f(x)$

x	1.0	1.3	1.6	1.9	2.2
$f(x)$	0.7651977	0.6200860	0.4554022	0.2818186	0.1103623

Solution (1/7)

The most appropriate linear polynomial uses $x_0 = 1.3$ and $x_1 = 1.6$ because 1.5 is between 1.3 and 1.6. The value of the interpolating polynomial at 1.5 is

$$P_1(1.5) = \frac{(1.5-1.6)}{(1.3-1.6)} f(1.3) + \frac{(1.5-1.3)}{(1.6-1.3)} f(1.6)$$

Interpolating Accuracy without underlying $f(x)$

x	1.0	1.3	1.6	1.9	2.2
$f(x)$	0.7651977	0.6200860	0.4554022	0.2818186	0.1103623

Solution (1/7)

The most appropriate linear polynomial uses $x_0 = 1.3$ and $x_1 = 1.6$ because 1.5 is between 1.3 and 1.6. The value of the interpolating polynomial at 1.5 is

$$\begin{aligned} P_1(1.5) &= \frac{(1.5-1.6)}{(1.3-1.6)} f(1.3) + \frac{(1.5-1.3)}{(1.6-1.3)} f(1.6) \\ &= \frac{(1.5-1.6)}{(1.3-1.6)} (0.6200860) + \frac{(1.5-1.3)}{(1.6-1.3)} (0.4554022) \end{aligned}$$

Interpolating Accuracy without underlying $f(x)$

x	1.0	1.3	1.6	1.9	2.2
$f(x)$	0.7651977	0.6200860	0.4554022	0.2818186	0.1103623

Solution (1/7)

The most appropriate linear polynomial uses $x_0 = 1.3$ and $x_1 = 1.6$ because 1.5 is between 1.3 and 1.6. The value of the interpolating polynomial at 1.5 is

$$\begin{aligned} P_1(1.5) &= \frac{(1.5-1.6)}{(1.3-1.6)} f(1.3) + \frac{(1.5-1.3)}{(1.6-1.3)} f(1.6) \\ &= \frac{(1.5-1.6)}{(1.3-1.6)}(0.6200860) + \frac{(1.5-1.3)}{(1.6-1.3)}(0.4554022) \\ &= 0.5102968 \end{aligned}$$

Interpolating Accuracy without underlying $f(x)$

Solution (2/7)

Interpolating Accuracy without underlying $f(x)$

Solution (2/7)

Two polynomials of degree 2 can reasonably be used, one with $x_0 = 1.3$, $x_1 = 1.6$ and $x_2 = 1.9$, which gives

$$P_2(1.5) = \frac{(1.5-1.6)(1.5-1.9)}{(1.3-1.6)(1.3-1.9)}(0.6200860)$$

Interpolating Accuracy without underlying $f(x)$

Solution (2/7)

Two polynomials of degree 2 can reasonably be used, one with $x_0 = 1.3$, $x_1 = 1.6$ and $x_2 = 1.9$, which gives

$$
\begin{aligned}
P_2(1.5) &= \frac{(1.5-1.6)(1.5-1.9)}{(1.3-1.6)(1.3-1.9)}(0.6200860) \\
&\quad + \frac{(1.5-1.3)(1.5-1.9)}{(1.6-1.3)(1.6-1.9)}(0.4554022)
\end{aligned}
$$

Interpolating Accuracy without underlying $f(x)$

Solution (2/7)

Two polynomials of degree 2 can reasonably be used, one with $x_0 = 1.3$, $x_1 = 1.6$ and $x_2 = 1.9$, which gives

$$\begin{aligned} P_2(1.5) &= \frac{(1.5-1.6)(1.5-1.9)}{(1.3-1.6)(1.3-1.9)}(0.6200860) \\ &\quad + \frac{(1.5-1.3)(1.5-1.9)}{(1.6-1.3)(1.6-1.9)}(0.4554022) \\ &\quad + \frac{(1.5-1.3)(1.5-1.6)}{(1.9-1.3)(1.9-1.6)}(0.2818186) = 0.5112857 \end{aligned}$$

and one with $x_0 = 1.0$, $x_1 = 1.3$, and $x_2 = 1.6$, which gives $\hat{P}_2(1.5) = 0.5124715$.

Interpolating Accuracy without underlying $f(x)$

x	1.0	1.3	1.6	1.9	2.2
$f(x)$	0.7651977	0.6200860	0.4554022	0.2818186	0.1103623

Solution (3/7)

Interpolating Accuracy without underlying $f(x)$

x	1.0	1.3	1.6	1.9	2.2
$f(x)$	0.7651977	0.6200860	0.4554022	0.2818186	0.1103623

Solution (3/7)

- In the third-degree case, there are also two reasonable choices for the polynomial.

Interpolating Accuracy without underlying $f(x)$

x	1.0	1.3	1.6	1.9	2.2
$f(x)$	0.7651977	0.6200860	0.4554022	0.2818186	0.1103623

Solution (3/7)

- In the third-degree case, there are also two reasonable choices for the polynomial. One with $x_0 = 1.3$, $x_1 = 1.6$, $x_2 = 1.9$ and $x_3 = 2.2$, which gives $P_3(1.5) = 0.5118302$.

Interpolating Accuracy without underlying $f(x)$

x	1.0	1.3	1.6	1.9	2.2
$f(x)$	0.7651977	0.6200860	0.4554022	0.2818186	0.1103623

Solution (3/7)

- In the third-degree case, there are also two reasonable choices for the polynomial. One with $x_0 = 1.3$, $x_1 = 1.6$, $x_2 = 1.9$ and $x_3 = 2.2$, which gives $P_3(1.5) = 0.5118302$.
- The second third-degree approximation is obtained with $x_0 = 1.0$, $x_1 = 1.3$, $x_2 = 1.6$ and $x_3 = 1.9$, which gives $\hat{P}_3(1.5) = 0.5118127$.

Interpolating Accuracy without underlying $f(x)$

x	1.0	1.3	1.6	1.9	2.2
$f(x)$	0.7651977	0.6200860	0.4554022	0.2818186	0.1103623

Solution (4/7)

- The fourth-degree Lagrange polynomial uses all the entries in the table.

Interpolating Accuracy without underlying $f(x)$

x	1.0	1.3	1.6	1.9	2.2
$f(x)$	0.7651977	0.6200860	0.4554022	0.2818186	0.1103623

Solution (4/7)

- The fourth-degree Lagrange polynomial uses all the entries in the table.
- With $x_0 = 1.0$, $x_1 = 1.3$, $x_2 = 1.6$, $x_3 = 1.9$, and $x_4 = 2.2$, the approximation is $P_4(1.5) = 0.5118200$.

Interpolating Accuracy without underlying $f(x)$

$P_3(1.5)$	$\hat{P}_3(1.5)$	$P_4(1.5)$
0.5118302	0.5118127	0.5118200

Solution (5/7)

Interpolating Accuracy without underlying $f(x)$

$P_3(1.5)$	$\hat{P}_3(1.5)$	$P_4(1.5)$
0.5118302	0.5118127	0.5118200

Solution (5/7)

- Because $P_3(1.5)$, $\hat{P}_3(1.5)$, and $P_4(1.5)$ all agree to within 2×10^{-5} units, we expect this degree of accuracy for these approximations.

Interpolating Accuracy without underlying $f(x)$

$P_3(1.5)$	$\hat{P}_3(1.5)$	$P_4(1.5)$
0.5118302	0.5118127	0.5118200

Solution (5/7)

- Because $P_3(1.5)$, $\hat{P}_3(1.5)$, and $P_4(1.5)$ all agree to within 2×10^{-5} units, we expect this degree of accuracy for these approximations.
- We also expect $P_4(1.5)$ to be the most accurate approximation, since it uses more of the given data.

Interpolating Accuracy without underlying $f(x)$

$P_3(1.5)$	$\hat{P}_3(1.5)$	$P_4(1.5)$
0.5118302	0.5118127	0.5118200

Solution (5/7)

- Because $P_3(1.5)$, $\hat{P}_3(1.5)$, and $P_4(1.5)$ all agree to within 2×10^{-5} units, we expect this degree of accuracy for these approximations.
- We also expect $P_4(1.5)$ to be the most accurate approximation, since it uses more of the given data.
- The function we are approximating is actually the Bessel function of the first kind of order zero, whose value at 1.5 is known to be 0.5118277.

Interpolating Accuracy without underlying $f(x)$

Solution (6/7)

Therefore, the true accuracies of the approximations are as follows:

$$
\begin{aligned}
|P_1(1.5) - f(1.5)| &\approx 1.53 \times 10^{-3} \\
|P_2(1.5) - f(1.5)| &\approx 5.42 \times 10^{-4} \\
|\hat{P}_2(1.5) - f(1.5)| &\approx 6.44 \times 10^{-4} \\
|P_3(1.5) - f(1.5)| &\approx 2.5 \times 10^{-6} \\
|\hat{P}_3(1.5) - f(1.5)| &\approx 1.50 \times 10^{-5} \\
|P_4(1.5) - f(1.5)| &\approx 7.7 \times 10^{-6}
\end{aligned}
$$

Concluding Remarks (7/7)

Interpolating Accuracy without underlying $f(x)$

Concluding Remarks (7/7)

- Although $P_3(1.5)$ is the most accurate approximation, if we had no knowledge of the actual value of $f(1.5)$, we would accept $P_4(1.5)$ as the best approximation since it includes the most data about the function.

Concluding Remarks (7/7)

- Although $P_3(1.5)$ is the most accurate approximation, if we had no knowledge of the actual value of $f(1.5)$, we would accept $P_4(1.5)$ as the best approximation since it includes the most data about the function.
- The theoretical Lagrange error term Theorem cannot be applied here because we have no knowledge of the fourth derivative of f.

Concluding Remarks (7/7)

- Although $P_3(1.5)$ is the most accurate approximation, if we had no knowledge of the actual value of $f(1.5)$, we would accept $P_4(1.5)$ as the best approximation since it includes the most data about the function.
- The theoretical Lagrange error term ▸ Theorem cannot be applied here because we have no knowledge of the fourth derivative of f.
- Unfortunately, this is generally the case.

Outline

Introduction to Neville's Method

Motivation

Introduction to Neville's Method

Motivation

- A practical difficulty with Lagrange interpolation is that the error term is difficult to apply, so the degree of the polynomial needed for the desired accuracy is generally not known until computations have been performed.

Introduction to Neville's Method

Motivation

- A practical difficulty with Lagrange interpolation is that the error term is difficult to apply, so the degree of the polynomial needed for the desired accuracy is generally not known until computations have been performed.
- A common practice is to compute the results given from various polynomials until appropriate agreement is obtained.

Introduction to Neville's Method

Motivation

- A practical difficulty with Lagrange interpolation is that the error term is difficult to apply, so the degree of the polynomial needed for the desired accuracy is generally not known until computations have been performed.
- A common practice is to compute the results given from various polynomials until appropriate agreement is obtained.
- However, the work done in calculating the approximation by the second polynomial does not lessen the work needed to calculate the third approximation; nor is the fourth approximation easier to obtain once the third approximation is known, and so on.

Introduction to Neville's Method

Motivation

- A practical difficulty with Lagrange interpolation is that the error term is difficult to apply, so the degree of the polynomial needed for the desired accuracy is generally not known until computations have been performed.
- A common practice is to compute the results given from various polynomials until appropriate agreement is obtained.
- However, the work done in calculating the approximation by the second polynomial does not lessen the work needed to calculate the third approximation; nor is the fourth approximation easier to obtain once the third approximation is known, and so on.
- We will now derive these approximating polynomials in a manner that uses the previous calculations to greater advantage.

Introduction to Neville's Method

Definition: Lagrange Polynomial $P_{m_1,m_2,\ldots,m_k}(x)$

- Let f be a function defined at $x_0, x_1, x_2, \ldots, x_n$, and suppose that m_1, m_2, ..., m_k are k distinct integers, with $0 \leq m_i \leq n$ for each i.

Introduction to Neville's Method

Definition: Lagrange Polynomial $P_{m_1,m_2,\ldots,m_k}(x)$

- Let f be a function defined at $x_0, x_1, x_2, \ldots, x_n$, and suppose that m_1, m_2, . . ., m_k are k distinct integers, with $0 \leq m_i \leq n$ for each i.
- The Lagrange polynomial that agrees with $f(x)$ at the k points $x_{m_1}, x_{m_2}, \ldots, x_{m_k}$ is denoted by

$$P_{m_1,m_2,\ldots,m_k}(x)$$

Introduction to Neville's Method

Example: $P_{1,2,4}(x)$

- Suppose that $x_0 = 1$, $x_1 = 2$, $x_2 = 3$, $x_3 = 4$, $x_4 = 6$, and $f(x) = e^x$.
- Determine the interpolating polynomial denoted $P_{1,2,4}(x)$, and use this polynomial to approximate $f(5)$.

Introduction to Neville's Method

$P_{1,2,4}(x)$ Solution (1/2)

This is the Lagrange polynomial that agrees with $f(x)$ at $x_1 = 2$, $x_2 = 3$, and $x_4 = 6$.

Introduction to Neville's Method

$P_{1,2,4}(x)$ Solution (1/2)

This is the Lagrange polynomial that agrees with $f(x)$ at $x_1 = 2$, $x_2 = 3$, and $x_4 = 6$. Hence

$$P_{1,2,4}(x) = \frac{(x-3)(x-6)}{(2-3)(2-6)}e^2 + \frac{(x-2)(x-6)}{(3-2)(3-6)}e^3 + \frac{(x-2)(x-3)}{(6-2)(6-3)}e^6.$$

Introduction to Neville's Method

$P_{1,2,4}(x)$ Solution (2/2)

So

$$f(5) \approx P(5) = \frac{(5-3)(5-6)}{(2-3)(2-6)}e^2 + \frac{(5-2)(5-6)}{(3-2)(3-6)}e^3 + \frac{(5-2)(5-3)}{(6-2)(6-3)}e^6$$

$$= -\frac{1}{2}e^2 + e^3 + \frac{1}{2}e^6 \approx 218.105.$$

Recursive Lagrange Polynomial Approximations

Theorem

Let f be defined at $x_0, x_1, \ldots, x_k$, and let x_j and x_i be two distinct numbers in this set.

Recursive Lagrange Polynomial Approximations

Theorem

Let f be defined at $x_0, x_1, \ldots, x_k$, and let x_j and x_i be two distinct numbers in this set. Then

$$P(x) = \frac{(x - x_j)P_{0,1,\ldots,j-1,j+1,\ldots,k}(x) - (x - x_i)P_{0,1,\ldots,i-1,i+1,\ldots,k}(x)}{(x_i - x_j)}$$

is the kth Lagrange polynomial that interpolates f at the $k + 1$ points $x_0, x_1, \ldots, x_k$.

Recursive Lagrange Polynomial Approximations

For ease of notation, let

$$Q \equiv P_{0,1,\ldots,i-1,i+1,\ldots,k} \quad \text{and} \quad \hat{Q} \equiv P_{0,1,\ldots,j-1,j+1,\ldots,k}$$

Recursive Lagrange Polynomial Approximations

For ease of notation, let

$$Q \equiv P_{0,1,\ldots,i-1,i+1,\ldots,k} \qquad \text{and} \qquad \hat{Q} \equiv P_{0,1,\ldots,j-1,j+1,\ldots,k}$$

Since $Q(x)$ and $\hat{Q}(x)$ are polynomials of degree $k-1$ or less, $P(x)$ is of degree at most k.

Recursive Lagrange Polynomial Approximations

For ease of notation, let

$$Q \equiv P_{0,1,\ldots,i-1,i+1,\ldots,k} \qquad \text{and} \qquad \hat{Q} \equiv P_{0,1,\ldots,j-1,j+1,\ldots,k}$$

Since $Q(x)$ and $\hat{Q}(x)$ are polynomials of degree $k-1$ or less, $P(x)$ is of degree at most k.

Proof (1/2)

First note that $\hat{Q}(x_i) = f(x_i)$, implies that

$$P(x_i) = \frac{(x_i - x_j)\hat{Q}(x_i) - (x_i - x_i)Q(x_i)}{x_i - x_j} = \frac{(x_i - x_j)}{(x_i - x_j)} f(x_i) = f(x_i).$$

Recursive Lagrange Polynomial Approximations

For ease of notation, let

$$Q \equiv P_{0,1,\ldots,i-1,i+1,\ldots,k} \qquad \text{and} \qquad \hat{Q} \equiv P_{0,1,\ldots,j-1,j+1,\ldots,k}$$

Since $Q(x)$ and $\hat{Q}(x)$ are polynomials of degree $k-1$ or less, $P(x)$ is of degree at most k.

Proof (1/2)

First note that $\hat{Q}(x_i) = f(x_i)$, implies that

$$P(x_i) = \frac{(x_i - x_j)\hat{Q}(x_i) - (x_i - x_i)Q(x_i)}{x_i - x_j} = \frac{(x_i - x_j)}{(x_i - x_j)} f(x_i) = f(x_i).$$

Similarly, since $Q(x_j) = f(x_j)$, we have $P(x_j) = f(x_j)$.

Recursive Lagrange Polynomial Approximations

Proof (2/2)

In addition, if $0 \leq r \leq k$ and r is neither i nor j, then $Q(x_r) = \hat{Q}(x_r) = f(x_r)$.

Recursive Lagrange Polynomial Approximations

Proof (2/2)

In addition, if $0 \leq r \leq k$ and r is neither i nor j, then $Q(x_r) = \hat{Q}(x_r) = f(x_r)$. So

$$P(x_r) = \frac{(x_r - x_j)\hat{Q}(x_r) - (x_r - x_i)Q(x_r)}{x_i - x_j} = \frac{(x_i - x_j)}{(x_i - x_j)} f(x_r) = f(x_r)$$

Recursive Lagrange Polynomial Approximations

Proof (2/2)

In addition, if $0 \leq r \leq k$ and r is neither i nor j, then $Q(x_r) = \hat{Q}(x_r) = f(x_r)$. So

$$P(x_r) = \frac{(x_r - x_j)\hat{Q}(x_r) - (x_r - x_i)Q(x_r)}{x_i - x_j} = \frac{(x_i - x_j)}{(x_i - x_j)} f(x_r) = f(x_r)$$

But, by definition, $P_{0,1,\ldots,k}(x)$ is the unique polynomial of degree at most k that agrees with f at $x_0, x_1, \ldots, x_k$. Thus, $P \equiv P_{0,1,\ldots,k}$.

Recursive Lagrange Polynomial Approximations

Comment on the Theorem

- This result implies that the interpolating polynomials can be generated recursively.

Recursive Lagrange Polynomial Approximations

Comment on the Theorem

- This result implies that the interpolating polynomials can be generated recursively.
- For example, we have

$$
\begin{aligned}
P_{0,1} &= \frac{1}{x_1 - x_0}[(x - x_0)P_1 + (x - x_1)P_0] \\
P_{1,2} &= \frac{1}{x_2 - x_1}[(x - x_1)P_2 + (x - x_2)P_1] \\
P_{0,1,2} &= \frac{1}{x_2 - x_0}[(x - x_0)P_{1,2} + (x - x_2)P_{0,1}]
\end{aligned}
$$

and so on.

Neville's Method: Recursive Generation

The following table illustrates how the interpolating polynomials can be generated recursively, where each row is completed before the succeeding rows are begun.

x_0	P_0				
x_1	P_1	$P_{0,1}$			
x_2	P_2	$P_{1,2}$	$P_{0,1,2}$		
x_3	P_3	$P_{2,3}$	$P_{1,2,3}$	$P_{0,1,2,3}$	
x_4	P_4	$P_{3,4}$	$P_{2,3,4}$	$P_{1,2,3,4}$	$P_{0,1,2,3,4}$

The procedure that uses the result of the theorem ▸ Theorem to recursively generate interpolating polynomial approximations is called Neville's method.

Neville's Method: Recursive Generation

Avoiding Cumbersome Notation

Neville's Method: Recursive Generation

Avoiding Cumbersome Notation

- The P notation used in the table is cumbersome because of the number of subscripts used to represent the entries.

Neville's Method: Recursive Generation

Avoiding Cumbersome Notation

- The P notation used in the table is cumbersome because of the number of subscripts used to represent the entries.
- Note, however, that as an array is being constructed, only two subscripts are needed.

Neville's Method: Recursive Generation

Avoiding Cumbersome Notation

- The P notation used in the table is cumbersome because of the number of subscripts used to represent the entries.
- Note, however, that as an array is being constructed, only two subscripts are needed.
- Proceeding down the table corresponds to using consecutive points x_i with larger i, and proceeding to the right corresponds to increasing the degree of the interpolating polynomial.

Neville's Method: Recursive Generation

Avoiding Cumbersome Notation

- The P notation used in the table is cumbersome because of the number of subscripts used to represent the entries.
- Note, however, that as an array is being constructed, only two subscripts are needed.
- Proceeding down the table corresponds to using consecutive points x_i with larger i, and proceeding to the right corresponds to increasing the degree of the interpolating polynomial.
- Since the points appear consecutively in each entry, we need to describe only a starting point and the number of additional points used in constructing the approximation.

Neville's Method: Recursive Generation

To avoid the multiple subscripts, we let $Q_{i,j}(x)$, for $0 \leq j \leq i$, denote the interpolating polynomial of degree j on the $(j+1)$ numbers $x_{i-j}, x_{i-j+1}, \ldots, x_{i-1}, x_i$; that is,

$$Q_{i,j} = P_{i-j,i-j+1,\ldots,i-1,i}.$$

Neville's Method: Recursive Generation

To avoid the multiple subscripts, we let $Q_{i,j}(x)$, for $0 \leq j \leq i$, denote the interpolating polynomial of degree j on the $(j+1)$ numbers $x_{i-j}, x_{i-j+1}, \ldots, x_{i-1}, x_i$; that is,

$$Q_{i,j} = P_{i-j,i-j+1,\ldots,i-1,i}.$$

Using this notation provides the following Q notation array.

x_0	$P_0 = Q_{0,0}$				
x_1	$P_1 = Q_{1,0}$	$P_{0,1} = Q_{1,1}$			
x_2	$P_2 = Q_{2,0}$	$P_{1,2} = Q_{2,1}$	$P_{0,1,2} = Q_{2,2}$		
x_3	$P_3 = Q_{3,0}$	$P_{2,3} = Q_{3,1}$	$P_{1,2,3} = Q_{3,2}$	$P_{0,1,2,3} = Q_{3,3}$	
x_4	$P_4 = Q_{4,0}$	$P_{3,4} = Q_{4,1}$	$P_{2,3,4} = Q_{4,2}$	$P_{1,2,3,4} = Q_{4,3}$	$P_{0,1,2,3,4} = Q_{4,4}$

Outline

Neville's Method: Recursive Generation Example

Example: Using the 'Q' Notation

Values of various interpolating polynomials at $x = 1.5$ were obtained in an earlier example using the following data:

x	1.0	1.3	1.6	1.9	2.2
$f(x)$	0.7651977	0.6200860	0.4554022	0.2818186	0.1103623

Apply Neville's method to the data by constructing a recursive table in the Q-notation array format.

Neville's Method: Recursive Generation Example

Solution (1/6)

- Let $x_0 = 1.0$, $x_1 = 1.3$, $x_2 = 1.6$, $x_3 = 1.9$, and $x_4 = 2.2$, then $Q_{0,0} = f(1.0)$, $Q_{1,0} = f(1.3)$, $Q_{2,0} = f(1.6)$, $Q_{3,0} = f(1.9)$, and $Q_{4,0} = f(2.2)$.

Neville's Method: Recursive Generation Example

Solution (1/6)

- Let $x_0 = 1.0$, $x_1 = 1.3$, $x_2 = 1.6$, $x_3 = 1.9$, and $x_4 = 2.2$, then $Q_{0,0} = f(1.0)$, $Q_{1,0} = f(1.3)$, $Q_{2,0} = f(1.6)$, $Q_{3,0} = f(1.9)$, and $Q_{4,0} = f(2.2)$.
- These are the five polynomials of degree zero (constants) that approximate $f(1.5)$, and are the same as data given in the example table.

Neville's Method: Recursive Generation Example

Solution (2/6)

Calculating the first-degree approximation $Q_{1,1}(1.5)$ gives

$$
\begin{aligned}
Q_{1,1}(1.5) &= \frac{(x - x_0)Q_{1,0} - (x - x_1)Q_{0,0}}{x_1 - x_0} \\
&= \frac{(1.5 - 1.0)Q_{1,0} - (1.5 - 1.3)Q_{0,0}}{1.3 - 1.0} \\
&= \frac{0.5(0.6200860) - 0.2(0.7651977)}{0.3} \\
&= 0.5233449
\end{aligned}
$$

Neville's Method: Recursive Generation Example

Solution (3/6)

Similarly,

$$Q_{2,1}(1.5) = \frac{(1.5-1.3)(0.4554022)-(1.5-1.6)(0.6200860)}{1.6-1.3}$$
$$= 0.5102968$$

$$Q_{3,1}(1.5) = 0.5132634 \quad \text{and} \quad Q_{4,1}(1.5) = 0.5104270$$

The best linear approximation is expected to be $Q_{2,1}$ because 1.5 is between $x_1 = 1.3$ and $x_2 = 1.6$.

Neville's Method: Recursive Generation Example

Solution (3/6)

Similarly,

$$Q_{2,1}(1.5) = \frac{(1.5 - 1.3)(0.4554022) - (1.5 - 1.6)(0.6200860)}{1.6 - 1.3}$$
$$= 0.5102968$$

$$Q_{3,1}(1.5) = 0.5132634 \quad \text{and} \quad Q_{4,1}(1.5) = 0.5104270$$

The best linear approximation is expected to be $Q_{2,1}$ because 1.5 is between $x_1 = 1.3$ and $x_2 = 1.6$.

Neville's Method: Recursive Generation Example

Solution (4/6)

In a similar manner, approximations using higher-degree polynomials are given by

$$Q_{2,2}(1.5) = \frac{(1.5 - 1.0)(0.5102968) - (1.5 - 1.6)(0.5233449)}{1.6 - 1.0}$$
$$= 0.5124715$$

$$Q_{3,2}(1.5) = 0.5112857$$

$$Q_{4,2}(1.5) = 0.5137361$$

Neville's Method: Recursive Generation Example

The higher-degree approximations are generated in a similar manner and are shown in the following table.

Solution (5/6)

1.0	0.7651977				
1.3	0.6200860	0.5233449			
1.6	0.4554022	0.5102968	0.5124715		
1.9	0.2818186	0.5132634	0.5112857	0.5118127	
2.2	0.1103623	0.5104270	0.5137361	0.5118302	0.5118200

Neville's Method: Recursive Generation Example

Solution (6/6)

- If the latest approximation, $Q_{4,4}$, was not sufficiently accurate, another node, x_5, could be selected, and another row added:

$$x_5 \quad Q_{5,0} \quad Q_{5,1} \quad Q_{5,2} \quad Q_{5,3} \quad Q_{5,4} \quad Q_{5,5}.$$

Neville's Method: Recursive Generation Example

Solution (6/6)

- If the latest approximation, $Q_{4,4}$, was not sufficiently accurate, another node, x_5, could be selected, and another row added:

$$x_5 \quad Q_{5,0} \quad Q_{5,1} \quad Q_{5,2} \quad Q_{5,3} \quad Q_{5,4} \quad Q_{5,5}.$$

Then $Q_{4,4}$, $Q_{5,4}$, and $Q_{5,5}$ could be compared to determine further accuracy.

Neville's Method: Recursive Generation Example

Solution (6/6)

- If the latest approximation, $Q_{4,4}$, was not sufficiently accurate, another node, x_5, could be selected, and another row added:

$$x_5 \quad Q_{5,0} \quad Q_{5,1} \quad Q_{5,2} \quad Q_{5,3} \quad Q_{5,4} \quad Q_{5,5}.$$

  Then $Q_{4,4}$, $Q_{5,4}$, and $Q_{5,5}$ could be compared to determine further accuracy.

- The function in this example is the Bessel function of the first kind of order zero, whose value at 2.5 is -0.0483838, and the next row of approximations to $f(1.5)$ is

$$2.5 \quad -0.0483838 \quad 0.4807699 \quad 0.5301984 \quad 0.5119070 \quad 0.5118430 \cdots$$

Neville's Method: Recursive Generation Example

Solution (6/6)

- If the latest approximation, $Q_{4,4}$, was not sufficiently accurate, another node, x_5, could be selected, and another row added:

$$x_5 \quad Q_{5,0} \quad Q_{5,1} \quad Q_{5,2} \quad Q_{5,3} \quad Q_{5,4} \quad Q_{5,5}.$$

  Then $Q_{4,4}$, $Q_{5,4}$, and $Q_{5,5}$ could be compared to determine further accuracy.

- The function in this example is the Bessel function of the first kind of order zero, whose value at 2.5 is -0.0483838, and the next row of approximations to $f(1.5)$ is

$$2.5 \quad -0.0483838 \quad 0.4807699 \quad 0.5301984 \quad 0.5119070 \quad 0.5118430 \cdots$$

The final new entry, 0.5118277, is correct to all 7 decimal places.

Outline

Neville's Method: 4-Digit Rounding Arithmetic

Example: 4-Digit Values of $\ln x$

The following table lists the values of $f(x) = \ln x$ accurate to the places given.

i	x_i	$\ln x_i$
0	2.0	0.6931
1	2.2	0.7885
2	2.3	0.8329

Use Neville's method and 4-digit rounding arithmetic to approximate $f(2.1) = \ln 2.1$ by completing the Neville table.

Neville's Method: 4-Digit Rounding Arithmetic

Solution (1/2)

Because $x - x_0 = 0.1$, $x - x_1 = -0.1$, $x - x_2 = -0.2$, and we are given $Q_{0,0} = 0.6931$, $Q_{1,0} = 0.7885$, and $Q_{2,0} = 0.8329$, we have

$$Q_{1,1} = \frac{1}{0.2}\left[(0.1)0.7885 - (-0.1)0.6931\right] = \frac{0.1482}{0.2} = 0.7410$$

and

$$Q_{2,1} = \frac{1}{0.1}\left[(-0.1)0.8329 - (-0.2)0.7885\right] = \frac{0.07441}{0.1} = 0.7441.$$

Neville's Method: 4-Digit Rounding Arithmetic

Solution (1/2)

Because $x - x_0 = 0.1$, $x - x_1 = -0.1$, $x - x_2 = -0.2$, and we are given $Q_{0,0} = 0.6931$, $Q_{1,0} = 0.7885$, and $Q_{2,0} = 0.8329$, we have

$$Q_{1,1} = \frac{1}{0.2}\left[(0.1)0.7885 - (-0.1)0.6931\right] = \frac{0.1482}{0.2} = 0.7410$$

and

$$Q_{2,1} = \frac{1}{0.1}\left[(-0.1)0.8329 - (-0.2)0.7885\right] = \frac{0.07441}{0.1} = 0.7441.$$

The final approximation we can obtain from this data is

$$Q_{2,1} = \frac{1}{0.3}\left[(0.1)0.7441 - (-0.2)0.7410\right] = \frac{0.2276}{0.3} = 0.7420.$$

Neville's Method: 4-Digit Rounding Arithmetic

Solution (2/2)

The calculations are summarized in the following table:

i	x_i	$x - x_i$	Q_{i0}	Q_{i1}	Q_{i2}
0	2.0	0.1	0.6931		
1	2.2	-0.1	0.7885	0.7410	
2	2.3	-0.2	0.8329	0.7441	0.7420

Accuracy of 4-Digit Approximations

Absolute Error *versus* Error Bound (1/2)

In the preceding example, we have $f(2.1) = \ln 2.1 = 0.7419$ to four decimal places, so the absolute error is

$$|f(2.1) - P_2(2.1)| = |0.7419 - 0.7420| = 10^{-4}$$

Accuracy of 4-Digit Approximations

Absolute Error *versus* Error Bound (1/2)

In the preceding example, we have $f(2.1) = \ln 2.1 = 0.7419$ to four decimal places, so the absolute error is

$$|f(2.1) - P_2(2.1)| = |0.7419 - 0.7420| = 10^{-4}$$

However, $f'(x) = 1/x$, $f''(x) = -1/x^2$, and $f'''(x) = 2/x^3$, so the Lagrange error formula ▸ Theorem gives the error bound

$$|f(2.1) - P_2(2.1)| = \left| \frac{f'''(\xi(2.1))}{3!}(x - x_0)(x - x_1)(x - x_2) \right|$$

Accuracy of 4-Digit Approximations

Absolute Error *versus* Error Bound (1/2)

In the preceding example, we have $f(2.1) = \ln 2.1 = 0.7419$ to four decimal places, so the absolute error is

$$|f(2.1) - P_2(2.1)| = |0.7419 - 0.7420| = 10^{-4}$$

However, $f'(x) = 1/x$, $f''(x) = -1/x^2$, and $f'''(x) = 2/x^3$, so the Lagrange error formula ▸ Theorem gives the error bound

$$\begin{aligned} |f(2.1) - P_2(2.1)| &= \left| \frac{f'''(\xi(2.1))}{3!}(x - x_0)(x - x_1)(x - x_2) \right| \\ &= \left| \frac{1}{3\,(\xi(2.1))^3}(0.1)(-0.1)(-0.2) \right| \end{aligned}$$

Accuracy of 4-Digit Approximations

Absolute Error *versus* Error Bound (1/2)

In the preceding example, we have $f(2.1) = \ln 2.1 = 0.7419$ to four decimal places, so the absolute error is

$$|f(2.1) - P_2(2.1)| = |0.7419 - 0.7420| = 10^{-4}$$

However, $f'(x) = 1/x$, $f''(x) = -1/x^2$, and $f'''(x) = 2/x^3$, so the Lagrange error formula ▸ Theorem gives the error bound

$$\begin{aligned} |f(2.1) - P_2(2.1)| &= \left| \frac{f'''(\xi(2.1))}{3!}(x - x_0)(x - x_1)(x - x_2) \right| \\ &= \left| \frac{1}{3\,(\xi(2.1))^3}(0.1)(-0.1)(-0.2) \right| \\ &\leq \frac{0.002}{3(2)^3} = 8.\overline{3} \times 10^{-5} \end{aligned}$$

Accuracy of 4-Digit Approximations

Absolute Error *versus* Error Bound

- Notice that the actual error, 10^{-4}, exceeds the error bound, $8.\overline{3} \times 10^{-5}$.

Accuracy of 4-Digit Approximations

Absolute Error *versus* Error Bound

- Notice that the actual error, 10^{-4}, exceeds the error bound, $8.\overline{3} \times 10^{-5}$.
- This apparent contradiction is a consequence of finite-digit computations.

Accuracy of 4-Digit Approximations

Absolute Error *versus* Error Bound

- Notice that the actual error, 10^{-4}, exceeds the error bound, $8.\overline{3} \times 10^{-5}$.
- This apparent contradiction is a consequence of finite-digit computations.
- We used four-digit rounding arithmetic, whereas the Lagrange error formula assumes infinite-digit arithmetic.

Accuracy of 4-Digit Approximations

Absolute Error *versus* Error Bound

- Notice that the actual error, 10^{-4}, exceeds the error bound, $8.\overline{3} \times 10^{-5}$.
- This apparent contradiction is a consequence of finite-digit computations.
- We used four-digit rounding arithmetic, whereas the Lagrange error formula assumes infinite-digit arithmetic.
- This caused our actual errors to exceed the theoretical error estimate.

Outline

Neville's Iterated Interpolation Algorithm

To evaluate the interpolating polynomial P on the $n+1$ distinct numbers $x_0, \ldots, x_n$ at the number x for the function f:

Neville's Iterated Interpolation Algorithm

To evaluate the interpolating polynomial P on the $n+1$ distinct numbers $x_0, \ldots, x_n$ at the number x for the function f:

INPUT numbers $x, x_0, x_1, \ldots, x_n$; values $f(x_0), f(x_1), \ldots, f(x_n)$ as the first column $Q_{0,0}, Q_{1,0}, \ldots, Q_{n,0}$ of Q

Neville's Iterated Interpolation Algorithm

To evaluate the interpolating polynomial P on the $n+1$ distinct numbers $x_0, \ldots, x_n$ at the number x for the function f:

INPUT	numbers $x, x_0, x_1, \ldots, x_n$; values $f(x_0), f(x_1), \ldots, f(x_n)$ as the first column $Q_{0,0}, Q_{1,0}, \ldots, Q_{n,0}$ of Q
OUTPUT	the table Q with $P(x) = Q_{n,n}$

Neville's Iterated Interpolation Algorithm

To evaluate the interpolating polynomial P on the $n+1$ distinct numbers $x_0, \ldots, x_n$ at the number x for the function f:

INPUT numbers $x, x_0, x_1, \ldots, x_n$; values $f(x_0), f(x_1), \ldots, f(x_n)$ as the first column $Q_{0,0}, Q_{1,0}, \ldots, Q_{n,0}$ of Q

OUTPUT the table Q with $P(x) = Q_{n,n}$

Step 1 For $i = 1, 2, \ldots, n$

for $j = 1, 2, \ldots, i$

set $$Q_{i,j} = \frac{(x - x_{i-j})Q_{i,j-1} - (x - x_i)Q_{i-1,j-1}}{x_i - x_{i-j}}$$

Neville's Iterated Interpolation Algorithm

To evaluate the interpolating polynomial P on the $n+1$ distinct numbers $x_0, \ldots, x_n$ at the number x for the function f:

INPUT numbers $x, x_0, x_1, \ldots, x_n$; values $f(x_0), f(x_1), \ldots, f(x_n)$ as the first column $Q_{0,0}, Q_{1,0}, \ldots, Q_{n,0}$ of Q

OUTPUT the table Q with $P(x) = Q_{n,n}$

Step 1 For $i = 1, 2, \ldots, n$

for $j = 1, 2, \ldots, i$

set $$Q_{i,j} = \frac{(x - x_{i-j})Q_{i,j-1} - (x - x_i)Q_{i-1,j-1}}{x_i - x_{i-j}}$$

Step 2 OUTPUT (Q)

STOP

Neville's Iterated Interpolation Algorithm

Additional Nodes & Stopping Criteria

- The algorithm can be modified to allow for the addition of new interpolating nodes. For example, the inequality

$$\left|Q_{i,i} - Q_{i-1,i-1}\right| < \varepsilon$$

can be used as a stopping criterion, where ε is a prescribed error tolerance.

Neville's Iterated Interpolation Algorithm

Additional Nodes & Stopping Criteria

- The algorithm can be modified to allow for the addition of new interpolating nodes. For example, the inequality

$$\left| Q_{i,i} - Q_{i-1,i-1} \right| < \varepsilon$$

  can be used as a stopping criterion, where ε is a prescribed error tolerance.

- If the inequality is true, $Q_{i,i}$ is a reasonable approximation to $f(x)$.

Neville's Iterated Interpolation Algorithm

Additional Nodes & Stopping Criteria

- The algorithm can be modified to allow for the addition of new interpolating nodes. For example, the inequality

$$\left| Q_{i,i} - Q_{i-1,i-1} \right| < \varepsilon$$

can be used as a stopping criterion, where ε is a prescribed error tolerance.

- If the inequality is true, $Q_{i,i}$ is a reasonable approximation to $f(x)$.

- If the inequality is false, a new interpolation point, x_{i+1}, is added.

Questions?

Reference Material

The Lagrange Polynomial: Theoretical Error Bound

Suppose $x_0, x_1, \ldots, x_n$ are distinct numbers in the interval $[a, b]$ and $f \in C^{n+1}[a, b]$. Then, for each x in $[a, b]$, a number $\xi(x)$ (generally unknown) between $x_0, x_1, \ldots, x_n$, and hence in (a, b), exists with

$$f(x) = P(x) + \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x - x_0)(x - x_1)\cdots(x - x_n)$$

where $P(x)$ is the interpolating polynomial given by

$$P(x) = f(x_0)L_{n,0}(x) + \cdots + f(x_n)L_{n,n}(x) = \sum_{k=0}^{n} f(x_k)L_{n,k}(x)$$

Return to Data Approximation Example

Return to Error Calculations for Tabulated Data Example

Recursive Lagrange Polynomial Approximations

Theorem

Let f be defined at $x_0, x_1, \ldots, x_k$, and let x_j and x_i be two distinct numbers in this set. Then

$$P(x) = \frac{(x - x_j)P_{0,1,\ldots,j-1,j+1,\ldots,k}(x) - (x - x_i)P_{0,1,\ldots,i-1,i+1,\ldots,k}(x)}{(x_i - x_j)}$$

is the kth Lagrange polynomial that interpolates f at the $k + 1$ points $x_0, x_1, \ldots, x_k$.

Return to Neville's Method