Interpolation, Approximation and Their Applications

We introduce the basic concepts of interpolation and its applications in the first part and that of least-squares in the second part.

Part I. Interpolation

Basic concepts of interpolation

1. Given a set of data points \((x_i, y_i) \in X \times Y, \ i = 1 : n\). We illustrate with the simple case that \(X, Y \subset \mathbb{R}\). The data represent a function \(f(x)\), which is either a discrete function or a sampling of a continuous function. We may assume for the moment that variable \(x\) is independent and we refer to the points \(x_i\) as the interpolation nodes. We assume further that the nodes are distinct.

2. Provided with a set of \(m\) modeling functions, for example, \(m\) polynomials of degree \(< m\), that are defined over a domain \(D\) containing \(X\). Let \(V\) be the vector space spanned by the modeling functions.

3. Find an interpolating function (interpolant) \(f \in V\) that satisfies the basic interpolating condition

\[
g(x_i) = y_i, \quad i = 1 : n.
\]

Certain interpolation problems require additional interpolation conditions.

Application in fast on-line function evaluation

The evaluation of many important functions is a complicated computation task. The values of a function \(f(x)\) at certain points, \((x_i, f(x_i)), \ i = 1 : n\), may be available from theoretical analysis or from off-line computation. These function values are tabulated in a handbook in the traditional way or in a

\[\text{Acknowledgment: this note is proofread and corrected by Jeremy Steinberg.}\]
database in electronic device for easy retrieval of the values at the nodes and fast evaluation at non-nodal points. By interpolation, the function values at non-nodal points are approximated by an interpolant $g(x)$ that is easy and fast to evaluate.

EXAMPLE. The normal distribution error function

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt.$$

It is required that the approximation error can be made arbitrarily small as required by different users. It may be desired or required that the smooth property of a function is preserved to certain degree. That is, the interpolation satisfies the osculating condition

$$f^{(k)}(x_i) = g^{(k)}(x_i), \quad k = 0 : m_i, \quad i = 1 : n.$$

In the special case $m_i = 1$, for all $i = 1 : N$, the first derivative (tangent lines) of the interpolating function agrees with the original function at the interpolation nodes. When the interpolation functions are polynomials, they are called Hermite interpolating polynomials.

A related problems is the inverse interpolation.

APPLICATION IN DATA MODELING AND REPRESENTATION

In some situations the data are available in discrete form $(x_i, y_i)$ only and we want to characterize more about the relation they represent or a trend they may indicate, such as the change in a stock value or in local temperature with time in the unit of hours or days. In certain rather ideal situations, a continuous function can be determined exactly by a finite number of sampling points and a set of modeling functions. One may find such study in signal processing, for instance. There are also situations we want to avoid storing a huge amount of data by representing the data on a selected subset and representing the rest in a compressed form, such as in the design and employment of a typeface setting. See the homework problems.

Interpolation with Polynomials

We introduce a few different approaches for interpolation with polynomials and their extensions to other functions.
Vandermonde System for Coefficient Determination

In the coefficient determination approach, we select a set of basis functions first and then determine the coefficients for the interpolant. To illustrate the approach, we choose the natural basis functions in \( P_n \), the set of monic monomials of degree \( \leq n \),

\[
1, x, x^2, x^3, \ldots, x^n.
\]

That is, the polynomial interpolant \( p(x) \) takes the form

\[
p(x) = \alpha_n x^n + \alpha_{n-1} x^{n-1} + \cdots + \alpha_1 x + \alpha_0,
\]

and we determine the coefficients \( \alpha_i \) to meet the interpolation condition

\[
p(x_j) = y_j, \quad j = 0 : n.
\]

In matrix form with \( n = 4 \) we have the following system of linear equations for the coefficients

\[
\begin{bmatrix}
1 & x_0 & x_0^2 & x_0^3 & x_0^4 \\
1 & x_1 & x_1^2 & x_1^3 & x_1^4 \\
1 & x_2 & x_2^2 & x_2^3 & x_2^4 \\
1 & x_3 & x_3^2 & x_3^3 & x_3^4 \\
1 & x_4 & x_4^2 & x_4^3 & x_4^4
\end{bmatrix}
\begin{bmatrix}
\alpha_0 \\
\alpha_1 \\
\alpha_2 \\
\alpha_3 \\
\alpha_4
\end{bmatrix}
=
\begin{bmatrix}
 y_0 \\
y_1 \\
y_2 \\
y_3 \\
y_4
\end{bmatrix},
\]

where the matrix is referred to as the Vandermonde matrix at the nodes \( x_i \) and hence the system is called a Vandermonde system. When the interpolation nodes are distinct, the matrix is nonsingular, and the coefficients exist uniquely.

Once the coefficients are determined, the evaluation of the interpolant at any point \( x \) can be done in \( O(n) \) arithmetic operations. We are concerned with the cost in solving the Vandermonde system when the interpolation data are changed frequently.

\[ \diamond \text{Find out the complexity of existing algorithms for solving a Vandermonde system.} \]
\[ \diamond \text{Find out reported problems with the existing algorithms.} \]

In a more general setting, we let let \( \{ b_j, j = 0 : n \} \) be a fixed basis of \( P_n \). Then the interpolating polynomial \( p \) can be represented as

\[
p(x) = \sum_{j=0}^{n} \alpha_j b_j(x).
\]
The interpolation condition gives the interpolating equations for the combination coefficients

\[
\begin{pmatrix}
  b_0(x_0) & b_1(x_0) & \cdots & b_n(x_0) \\
  b_0(x_1) & b_1(x_1) & \cdots & b_n(x_1) \\
  \vdots & \vdots & \ddots & \vdots \\
  b_0(x_n) & b_1(x_n) & \cdots & b_n(x_n)
\end{pmatrix}
\begin{pmatrix}
  \alpha_0 \\
  \alpha_2 \\
  \vdots \\
  \alpha_n
\end{pmatrix}
= \begin{pmatrix}
  y_0 \\
  y_1 \\
  \vdots \\
  y_n
\end{pmatrix}.
\]

The matrix depends on the basis functions and the interpolation nodes only. The Vandermonde system is a special case. If the matrix is diagonal, the coefficients can be determined easily and quickly. By scaling, we may consider the special case that the matrix is the identity matrix. In other words, the basis functions satisfy the basic interpolation conditions \( b_i(x_j) = \delta_{ij} \) at the interpolation nodes. The solution can then be read out, i.e., \( y_i \) are the coefficients. Such a matrix can be constructed in two ways: given a set of basis functions, find a set of interpolation nodes; or given a set of interpolation nodes, construct a set of functions. We introduced the latter.

**Lagrange Interpolation**

In Lagrange interpolation we are given a set of distinct interpolation nodes \( x_i, \ i = 0 : n \) and we construct a set of basis polynomials \( L_i \) such that

\[ L_i(x_j) = \delta_{ij}, \quad j = 0 : n, \]

herein, \( \delta_{ij} \) is the Kronecker delta function. Then the interpolant on any given set of values \( y_i \) can be simply represented as a linear combination of \( L_i(x) \)

\[ p(x) = \sum_{i=0}^{n} y_i L_i(x). \]

The remaining question is on the existence of \( L_i \). The answer is yes when all nodes \( x_i \) are distinct and it is proved by construction,

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

which are referred to as Lagrange polynomials.

Larange’s approach is straightforward and useful in analysis because the interpolant is represented in a clear closed form in terms of the interpolation nodes and values. And it also leads to close-form formula for the derivatives.
and antiderivatives of the interpolant \( p(x) \). However, Lagrange’s representation of the interpolant is hardly used directly for function evaluation. Consider the following issues.

- All Lagrange’s polynomials on \( n + 1 \) nodes are of degree \( n \). What is the arithmetic complexity for evaluating the interpolant \( p(x) \) based on its Lagrange representation?

- Consider the case that the interpolant \( p(x) \) is of degree much lower than \( n \), for example, \( p(x) \) is linear. What is the complexity for evaluating a linear interpolant represented in terms of Lagrange’s polynomials? Is there any cancellation in arithmetic operations?

- Iterative construction. Denote by \( L_{m,i}(x) \) the Lagrange polynomials on \( m + 1 \) nodes. Let \( n > m \). Represent \( L_{n,i} \) in terms of \( L_{m,i} \). Does the new representation lead to a more efficient evaluation approach?

The idea in the last item leads to Neville’s method for efficient evaluation of an polynomial interpolant.

**Interpolation Accuracy**

Following construction and evaluation methods, we consider the accuracy of interpolation at a non-nodal point. The following accuracy estimation is based on a smoothness assumption.

**Theorem**

(Accuracy estimation of polynomial interpolation)

Let \( g \in C^{n+1}[a, b] \). Let \( p_n \) be the interpolating polynomial at \( n \) distinct nodes \( x_i, i = 1 : n \), in \([a, b]\). Then

\[
g(x) = p_n(x) + e(x),
\]

with

\[
e(x) = \frac{f^{(n+1)}(\xi)}{(n + 1)!} \prod_{j}(x - x_j), \quad x \in [a, b].
\]

- The Taylor theorem may be used for the proof of the above theorem.

- Find some sufficient conditions so that the interpolation error at any non-nodal point decreases as the number of interpolation nodes increases.
diamond Find some conditions so that the interpolation error does not decrease as the number of interpolation nodes increases.

diamond In comparison to approximation by Taylor polynomials, the interpolating polynomials do not require the evaluation of derivatives at the nodal points.

**Runge’s example : a test case**

Consider polynomial interpolation of the *Runge’s function*

\[
\text{runge}(x) = (1 + x^2)^{-1}
\]

at equa-spaced nodes in \([-5, 5]\).

- The interpolating polynomial oscillates more as the number of nodes increases. The errors at the non-nodal points do not decrease with the increase of nodes.

- Is this a contradiction to the above theorem on polynomial interpolation? Try to explain the reasons for the phenomenon.

- Is this a contradiction to Weierstrass theorem on function approximation with polynomials? Try to elaborate your answer.

- Make some comments about polynomial interpolations based on your investigation with this test function.

**Extension of polynomial interpolation**

Polynomial interpolation may be extended in many different ways, which can be used in combination. A straightforward approach is interpolation with rational functions by making a simple change in the dependent variable \(y\). For example, one can use rational functions to interpolate a set of given data.

- When \(y_i \neq 0\), one may consider interpolating \((x_i, y_i^{-1})\) instead with polynomial \(p(x)\). That is, the data is interpolated with a rational function \(1/p(x)\).

  This variable change solves the curve fitting problem with sampled data from Runge’s function.
○ One may first map \( y \) to \( q(x)y \) where \( q(x) \) is a chosen polynomial with \( q(x_i) \neq 0 \). Then interpolate \( (x_i, q(x_i)y_i) \) with polynomial \( p(x) \). This amounts to interpolation with a rational function \( p(x)/q(x) \).

We introduce next two important extension approaches.

**Interpolation with piecewise polynomials and splines**

In spline interpolation, the interpolation domain is partitioned into \( n \) smaller subdomains. For example, an interpolation interval \( [a, b] \) is partitioned into subintervals \( [x_{i-1}, x_i] \) by the interpolation nodes \( x_i, i = 0 : n \), they are often called knots in this case. The interpolating function on \( [a, b] \) is a *piecewise polynomial* in the sense that it is a polynomial on each and every subinterval \( [x_{i-1}, x_i] \). It is not necessarily a polynomial on the whole domain \( [a, b] \). The degree of a piecewise polynomial is the highest degree among the polynomial pieces over the subintervals. An interpolating piecewise polynomial is continuous on \( [a, b] \) by the interpolation condition; its derivatives, however, may not be continuous at the nodes. A *spline* function is a piecewise polynomial with additional conditions on the interior nodes \( x_i, i = 1 : n - 1 \). Specifically, a spline function \( S \) of degree \( k \) on knots \( x_i \) satisfies the following conditions:

1. **A PIECEWISE POLYNOMIAL**
   
   On each \( [x_{i-1}, x_i] \), \( S(x) \) is a polynomial of degree \( \leq d \),
   
   \[
   S(x) = s_i(x) \in P_k, \quad x \in [x_{i-1}, x_i], \quad i = 0 : n - 1.
   \]

2. **CONTINUOUS DERIVATIVES AT THE INTERIOR NODES**
   
   \[
   s^{(k)}_{i-1}(x_i) = s^{(k)}_i(x_i). \quad k = 0 : d - 1
   \]

   Note that \( S^{(d)} \) is piecewise constant and therefore is not necessarily continuous (is \( S(x) \) a polynomial on \( [a, b] \) when \( S^{(d)} \) is continuous?) An interpolating spline at the knots satisfies the interpolation condition

   \[
   s_i(x_{i-1}) = y_{i-1}, \quad s_i(x_i) = y_i, \quad i = 1 : n
   \]

   A spline interpolant satisfies some additional boundary condition. For example, a *cubic spline* may be required to satisfy either the natural boundary condition

   \[
   s''(x_0) = s''(x_n) = 0,
   \]
or the clamped boundary condition
\[ s'(x_0) = g'(x_0), \quad s'(x_n) = g'(x_n). \]

In the latter case the slopes at the end points are given. The interpolant is respectively referred to as the cubic natural spline or the cubic clamped spline.

To determine a cubic spline interpolant, it is convenient to represent the piecewise polynomial in the translated form
\[ s_j(x) = \alpha_j + \beta_j(x - x_j) + \gamma_j(x - x_j)^2 + \delta_j(x - x_j)^3. \]

There are \((d + 1)n\) coefficients in total to be determined. We leave it as an exercise problem to find out how many equations we have for the coefficients based on the interpolation condition and spline properties.

Remarks

- Find at least two approaches to determining a cubic spline interpolant and discuss the advantages and disadvantages of each approach.
- With a fixed set of \(n + 1\) knots the set of natural cubic splines forms a subspace of \(C^2[a, b]\).
- Except the linear splines \((d = 1)\) and quadratic splines \((d = 2)\), a spline may change globally when the data are locally changed.
- Estimate interpolation accuracy.
- Make comparison to interpolation with piecewise Taylor polynomials.

Parametric Interpolation

In parametric interpolation, we treat \(x_i\) and \(y_i\) equally and take them as functions of parameter \(t\) at nodes \(t_i\). We introduce the parameter variable \(t\) and look for a pair of interpolating functions of \(t\), \([x(t), y(t)]\), so that
\[ x(t_i) = x_i, \quad y(t_i) = y_i, \quad i = 0 : n. \]

The parameterization includes the special case \(t = x\). There is flexibility in choosing the range and interpolation nodes of the parameter. Except when \(t = x\), the following setup is convenient:
\[ t \in [0, 1], \quad t_i = i/n, \quad i = 0 : n. \]
Parametric interpolation can be viewed as a special case of interpolating points of multiple coordinates \((t_i, x_i, y_i)\) where \(t_i\) is taken as the independent variable.

The parameterization does not make interpolation more complicated. In the simplest case, we interpolate \((t_i, x_i)\) and \((t_i, y_i)\) separately as in the earlier discussion. But we have more advantages with parametric interpolation.

- It is permissible that the data \((x_i, y_i)\) have multiple values of \(y\) associated with a single value of \(x\) or vice versa.

- When both \(x(t)\) and \(y(t)\) are polynomials, the function \(y(x)\), or \(x(y)\), is not necessarily a polynomial.

  For example, when \(x(t) = t^3\) and \(y(t) = t\), \(y(x) = x^{1/3}\). Note that this is not included in interpolation by rational functions.

- The subspace for interpolating function \(x(t)\) is not necessarily the same as that for \(y(t)\).

- The change in variable can be applied to both \(x\) and \(y\).

In computer graphics we often use interpolation with parametric piece-wise cubic Hermite polynomials in \textit{Bézier representation}. See the homework assignment.

**Review of interpolation with polynomials**

1. basis functions: monic monomials
   - Vandermonde systems of linear equations
   - Fast solution of Vandermonde systems

2. basis functions: polynomials with local supports
   - Taylor polynomials for smooth functions,
   - Splines, especially cubic splines,
   - Fast solution of tridiagonal systems,
   - Piecewise Hermite Polynomials

3. basis functions: orthogonal polynomials:
   - Chebyshev polynomials,
   - Hermite polynomials,
- Legendre polynomials.

They differ in weight distribution and in interval domains.

4. basis functions in vector form
   - parameterization
   - multi-dimensional data

**Interpolations with trigonometric polynomials**

We leave this subsection for self-study.

- Find at least two approaches for interpolation with trigonometric polynomials. (Cf. the notes on trigonometric polynomials.)
  Discuss on the advantages and disadvantages of each approach.

- Discuss on some special cases that make the computation more efficient.

- Estimate interpolation accuracy.

- Find a test function with which the interpolation is not effective.

- Find an extension approach that at least improves the test case.