Interpolation, Approximation and Their Applications

PART I : Interpolation

We consider the following two basic cases for interpolation in a subspace of finite dimension.

1. Given a set of data points $(x_i, g(x_i)) = (x_i, y_i) \in X \times Y$, $i = 1 : N$, X and $Y \subset \mathcal{R}$ (or $\mathcal{C}$) are the domains x_i and y_i reside, respectively. The variable x is assumed to be independent, and the points x_i are called the interpolation nodes and assumed distinct. Provided with a specific linear subspace V of functions in $C(X)$. Find an interpolating function f in V satisfying the *interpolating condition*

   $$f(x_i) = y_i, \quad i = 1 : N.$$

   An interpolation function is also called interpolant. In this case of interpolation,

   - ◇ The interpolation models a set of tabulated function values or discrete data into a continuous function. We call such a process *data fitting* or *curve fitting*.
   - ◇ The continuous function (curve) may characterize the relation between variables x and y more than their correspondence at the discrete points. It can be used to estimate variable y corresponding to a non-nodal point $x \in [a, b] - \{x_i\}$ (interpolation) or to a point outside of $[a, b]$ (extrapolation).

- ◇ For the same set of data, the interpolation changes with the selection of subspaces. The following are commonly used for interpolations.
  - (a) polynomials
  - (b) splines
  - (c) trigonometric polynomials

2. Given a function $g \in C^m(X)$, $m \geq 1$, and its derivatives at a set of distinct nodes

$$\left(x_i, g^k(x_i), k = 0 : m_i\right), \quad m_i \leq r \leq m, \quad i = 1 : N$$

Provided with a specific linear subspace in $C^r(X)$. Find an interpolating function f in the subspace satisfying the *osculating condition*

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

This is the case of function approximation via interpolation.

- ◇ The interpolating function f is used to replace or simplify the original function g with certain smooth property preserved at the discrete interpolation nodes and their neighborhood. For example, to evaluate a complicated function one may pre-compute the function at certain reference or nodal points and evaluate the function at the other points by the interpolating function. We may call such a process *curve simplification*.
- ◇ 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.

Interpolation with Polynomials

We introduce different approaches for interpolation with polynomials.

Lagrange interpolation

Define the interpolating polynomials $L_{n,j}$ on the interpolation nodes

$$L_{N,j}(x_i) = \delta_{ij}, \quad i, j = 1 : N.$$

In other words, $L_{N,j}$ interpolates the special data (x_i, y_i) with y_j equal to 1 and $y_i = 0$, $i \neq j$. Then the interpolating polynomial is simply represented as

$$p(x) = \sum_{j=1}^{N} y_j L_j(x_i).$$

It is easy to see that the special polynomials $L_{N.j}$ are linearly independent if x_i are distinct, and hence form a basis for P_n. Note that each and every of the basis polynomials is of degree $N-1$.

The basis polynomials $L_{N,j}$ can be easily constructed as follows.
For $N = 2$,

$$L_{2,1}(x) = \frac{x - x_2}{x_1 - x_2}, \quad L_{2,2}(x) = \frac{x - x_1}{x_2 - x_1}.$$

For $N = 3$,

$$\begin{aligned} L_{3,1}(x) &= \frac{x-x_2}{x_1-x_2} \cdot \frac{x-x_3}{x_1-x_3}, \\ L_{3,2}(x) &= \frac{x-x_1}{x_2-x_1} \cdot \frac{x-x_3}{x_2-x_3}, \\ L_{3,3}(x) &= \frac{x-x_1}{x_3-x_1} \cdot \frac{x-x_2}{x_3-x_2}, \end{aligned}$$

In general,

$$L_{N,j}(x) = \prod_{i \neq j} \frac{x-x_i}{x_j-x_i}, \quad j = 1:N$$

These polynomials are known as Lagrange polynomials at the interpolation nodes x_i.

The Lagrange approach is useful in analysis. For example, we have shown the *existence* of a polynomial interpolating the data at distinct nodes.

We have some comments on the evaluation.

◇ Polynomial evaluation with the Lagrange representation is of high complexity when N the size of data is large. Even the Neville evaluation method takes $O(N^2)$ arithmetic operations.

◇ Since each of the Lagrange polynomials is of degree $N-1$, there are cancellations in degree when the data are from a polynomial of a lower degree. The extreme case is $y_i = c$, $i = 1:N$.

◇ The evaluation complexity can be reduced for the case that the interpolation points are equally spaced.

Coefficient determination with a fixed basis

Instead of building the basis functions on every and each set of n interpolation nodes, we let $\{b_j, j = 1 : N\}$ be a fixed basis of P_n. Then the interpolating polynomial p can be represented as

$$p(x) = \sum_{j}^{N} \alpha_j b_j(x).$$

The interpolation condition gives the interpolating equations for the combination coefficients

$$\begin{bmatrix} b_1(x_1) & b_2(x_1) & \cdots & b_N(x_1) \\ b_1(x_2) & b_2(x_2) & \cdots & b_N(x_2) \\ & & \cdots & \\ b_1(x_N) & b_2(x_N) & \cdots & b_N(x_N) \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}.$$

Note that if $b_j(x_i) = \delta_{ij}$, then b_j are the Lagrange polynomials at the nodes x_i, and the matrix is the identity. When the set of interpolation nodes is changed, the matrix is changed as well. We find the coefficients by numerically solving the system of linear equations.

When the basis is $\{1, x, \cdots, x^{N-1}\}$, the matrix is known as Vandermonde matrix. It is nonsingular as long as the nodes are distinct. In other words, the interpolating coefficients can be determined for any set of data (x_i, y_i) as long as x_i are distinct. Once the coefficients are obtained, the evaluation at any point x can be done with $O(N)$ operations. The question left is on the complexity of solving the equations.

◇ Find out the complexity of existing algorithms for solving Vandermonde system.

◇ Find out reported problems with the existing algorithms.

- ◇ Find an orthogonal basis for P_n and discuss the advantages and disadvantages.

Interpolation accuracy

We want to estimate the accuracy of interpolation at a non-nodal point in X. The following accuracy estimation is based on a smoothness assumption and is in the sense of L_∞ norm.

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.
- ◇ The approximation error at a nodal point is zero.
- ◇ Find some sufficient conditions so that the interpolation error at any non-nodal point decreases as the number of interpolation nodes increases.

  (Consider, e.g. , the trigonometric functions $\cos(mx)$ on $[-\pi, \pi]$.)

- ◇ Find some conditions so that that the interpolation error does not decrease as the number of interpolation nodes increases.
- ◇ In comparison to approximation by Taylor polynomials, the interpolating polynomial do not require the evaluation of derivatives.
- ◇ When the data is locally changed, the interpolating function may be changed globally.

Test cases

Consider polynomial interpolation of the **Runge's function**

$$g(x) = (1 + x^2)^{-1}$$

at equa-spaced nodes in $[-5, 5]$.

- ○ The interpolating polynomial seems oscillating more as the number of nodes increases. The errors at the non-nodal points do not seem decreasing 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.
- ○ Find some other test functions with which polynomial interpolations are not effective computationally and numerically.

Extension/modification of polynomial interpolation

Polynomial interpolation may be extended in many different ways. One approach is to make a change in variable y.

- When $y_i \geq 0$, one may consider interpolating $(x_i, \sqrt{y_i})$ instead.
- When $y_i \neq 0$, one may consider interpolating (x_i, y_i^{-1}) instead. This is equivalent to interpolation with rational function of the form $1/p(x)$ where p is a polynomial.

  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 is a polynomial with $q(x_i) \neq 0$. Then interpolate $(x_i, q(x_i)y_i)$ with polynomials. This amounts to interpolation with a rational function $p(x)/q(x)$.
- Find another extension method.

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.

Interpolation with splines

We consider the case $X = [a, b]$. In spline interpolation, the interval $[a, b]$ is partitioned into n smaller subintervals $[x_{i-1}, x_i]$ by $n+1$ interpolation nodes x_i, $i = 0 : n$. Here we let the index start with 0, for convenience. A *spline* $s(x)$ of degree d is a piece-wise polynomial in C^{d-1}, namely,

1. PIECEWISE POLYNOMIAL
   On each $[x_{i-1}, x_i]$, $S(x)$ is a polynomial of degree $\le d$,
   $$s(x) = p_{i-1}(x), \quad x \in [x_{i-1}, x_i], \quad i = 0 : n-1.$$

2. SMOOTHNESS AT THE INTERIOR NODES

$$p_{i-1}^{(k)}(x_i) = p_i^{(k)}(x_i). \quad k = 0 : d-1$$

Since $s^{(d)}$ is piecewise constant on $[a, b]$, not necessarily continuous, $s(x)$ is not necessarily a polynomial over $[a, b]$.

A spline interpolant satisfies the interpolating condition

$$s(x_i) = g(x_i) = y_i, \quad i = 0 : n,$$

and some additional boundary condition.

A *cubic spline interpolant* satisfies either the *natural boundary condition*

$$s'(x_0) = s(x_n) = 0,$$

or the *clamped boundary condition*

$$s'(x_0) = g'(x_0), \quad s'(x_n) = g'(x_n).$$

And the interpolant is correspondingly called the cubic natural spline or the cubic clamped spline. To determine cubic spline interpolants, it is convenient to represent the piecewise polynomial in the translated form

$$p_j(x) = \alpha_j + \beta_j(x - x_j) + \gamma_j(x - x_j)^2 + \delta_j(x - x_j)^3.$$

REMARKS

◇ With a fixed set of $n+1$ partition nodes the set of natural cubic splines is a subspace of $C^2[a, b]$.

◇ Find out the relationship between the set of natural cubic splines and the set of clamped cubic splines With the same set of partition nodes.

- ◇ Find at least two approaches for interpolation with cubic splines. And verify that the natural (or clamped) cubic spline interpolant exists and is unique with distinct interpolation nodes.
- ◇ Discuss on the advantages and disadvantages of each approach.
- ◇ Discuss on some special cases that make the computation more efficient.
- ◇ Except the linear splines ($d = 1$) and quadratic splines ($d = 2$), a spline may change globally when the data is locally changed.
- ◇ Estimate interpolation accuracy.
- ◇ Make comparison to interpolation with piecewise Taylor polynomials.

Parametric Interpolation

In parametric interpolation (curve fitting), 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.

$$t \in [0, 1], \quad t_i = i/n, \quad i = 0 : n.$$

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.

Applications in computer graphics use interpolation with parametric piece-wise cubic Hermite polynomials in Bézier representation. See the homework assignment.