Chapter 3: Polynomial Interpolation

1 Polynomial Interpolation

A typical polynomial interpolation problem can be formulated as: given a set of data points $[t_1, y_1]$, $[t_2, y_2]$, ..., $[t_n, y_n]$, assuming $t_1 < t_2 < ... < t_n$, find a polynomial of degree at most $n-1$,

$$p_{n-1}(t) = x_1 + x_2 t + x_3 t^2 + \ldots + x_n t^{n-1},$$

where $x_1, x_2, ..., x_n$ represent the coefficients of this polynomial, such that

$$p_{n-1}(t_i) = x_1 + x_2 t_i + x_3 t_i^2 + \ldots + x_n t_i^{n-1} = y_i, \quad i = 1, 2, \ldots, n.$$

The above n equations can be written as the following system for the polynomial coefficients x_i, $i = 1, 2, \ldots, n$,

$$A\mathbf{x} = \begin{bmatrix} 1 & t_1 & t_1^2 & & t_1^{n-1} \\ 1 & t_2 & t_2^2 & \ldots & t_2^{n-1} \\ 1 & t_3 & t_3^2 & & t_3^{n-1} \\ & \vdots & & \ddots & \vdots \\ 1 & t_n & t_n^2 & \ldots & t_n^{n-1} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix}.$$

The matrix A is called the *Vandermonde* matrix. In the following section, we will show that this matrix is nonsingular and there is a unique solution to the above system. It takes $O(n^3)$ flops to solve the above linear system by using Gauss elimination. Also, the Vandermonde matrix may become very ill conditioned when n is large.

2 Interpolation

A more general interpolation problem can be formulated as: given a set of data points $[t_1, y_1]$, $[t_2, y_2]$, ..., $[t_m, y_m]$, assuming $t_1 < t_2 < ... < t_m$, determine a function $f(t) : \mathbb{R} \to \mathbb{R}$, such that

$$f(t_i) = y_i, \quad i = 1, 2, ..., m. \tag{1}$$

Here $f(t)$ can be any type of functions, e.g., polynomials, or trigonometric functions.

Let us consider to determine the function $f(t)$ among a n-dimensional function space whose basis functions are $\phi_1(t)$, $\phi_2(t)$, ..., $\phi_n(t)$. Denote the function $f(t)$ by

$$f(t) = \sum_{j=1}^{n} x_j \phi_j(t),$$

where x_j, $j = 1, ..., n$, are the coefficient. Problem (1) can then be formulated as: find the coefficient vector $\mathbf{x} = [x_1, x_2, ..., x_n]$ such that

$$f(t_i) = \sum_{j=1}^{n} x_j \phi_j(t_i) = y_i, \quad i = 1, 2, ..., m.$$

In matrix form, it can be written as: determine $\mathbf{x}$ such that

$$A\mathbf{x} = \begin{bmatrix} \phi_1(t_1) & \phi_2(t_1) & \dots & \phi_n(t_1) \\ \phi_1(t_2) & \phi_2(t_2) & & \phi_n(t_2) \\ \vdots & & \ddots & \vdots \\ \phi_1(t_m) & \phi_2(t_m) & \dots & \phi_n(t_m) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = y. \tag{2}$$

where each entry of the matrix A is determined by $A_{ij} = \phi_j(t_i)$.

There are m equations in the system (2) to determine n unknowns. If $m > n$, the system (2) may not have a solution, i.e., there is not such a function in the n-dimensional space to interpolate m data points. In this case, a *least squares solution* will be considered.

Here we only consider the case $m = n$, i.e., when the dimension of the functional space equals to the number of interpolation points. In this case, we call the function $f(t)$ satisfying (1) the interpolation function.

3 Other bases in Polynomial Interpolation

Different types of function $f(t)$ can be considered in the interpolation problems. Among them, polynomials are the simplest and the most commonly used. Let us denote P_{n-1} the set of all polynomials of degree at most $n-1$. We know that the dimension of the space P_{n-1} is n.

The most popular polynomial basis is probably the *monomial basis*, $1, t, t^2, \dots, t^{n-1}$. We have just discussed the polynomial interpolation in monomial basis at the beginning of this chapter, where to determine the interpolation polynomial, it takes $O(n^3)$ flops to solve the Vandermonde system of equations.

Some other bases can also be used in the polynomial interpolation.

3.1 Lagrange basis

The following defined Lagrange polynomial functions also form a basis of the polynomial space. Given a set of n interpolation points $(t_i, y_i), i = 1, 2, ..., n$, where $t_1 < t_2 < ... < t_n$, define the following Lagrange polynomial functions

$$\begin{aligned} L_1(t) &= \frac{(t-t_2)(t-t_3)...(t-t_n)}{(t_1-t_2)(t_1-t_3)...(t_1-t_n)}, \\ L_2(t) &= \frac{(t-t_1)(t-t_3)...(t-t_n)}{(t_2-t_1)(t_2-t_3)...(t_2-t_n)}, \\ &\dots \\ L_n(t) &= \frac{(t-t_1)(t-t_2)...(t-t_{n-1})}{(t_n-t_1)(t_n-t_2)...(t_n-t_{n-1})}, \end{aligned}$$

where in general

$$L_k(t) = \frac{(t-t_1)...(t-t_{k-1})(t-t_{k+1})...(t-t_n)}{(t_k-t_1)...(t_k-t_{k-1})(t_k-t_{k+1})...(t_k-t_n)}.$$

We can see that, for each $k = 1, 2, ..., n$, $L_k(t) \in P_{n-1}$, and satisfies, for $i = 1, 2, \ldots, n$

$$L_k(t_i) = \begin{cases} 0, & i \neq k, \\ 1, & i = k. \end{cases}$$

Graphically, each function $L_k(t)$ equals one at the node t_k, and equals zero at the other nodes t_i, for $i \neq k$.

Given any set of interpolation points, $(t_i, y_i), i = 1, 2, ..., n$, where $t_1 < t_2 < ... < t_n$, the polynomial

$$p_{n-1}(t) = y_1 L_1(t) + y_2 L_2(t) + y_3 L_3(t) + \ldots + y_n L_n(t).$$

belongs to P_{n-1} and satisfies

$$p_{n-1}(t_i) = y_i, \quad i = 1, 2, 3, ..., n.$$

This shows the existence of the interpolation polynomial for any given set of interpolation points (t_i, y_i), as long as $t_1 < t_2 < ... < t_n$. In fact the interpolation polynomial can be represented explicitly, as shown above, as a linear combination of the Lagrange polynomial functions. Now the question is: is such an interpolation polynomial unique? let us assume that both $p(t)$ and $q(t)$ belong to P_{n-1}, and they both satisfy the interpolation conditions, i.e., for $i = 1, 2, 3, ..., n$,

$$p(t_i) \;=\; q(t_i) \;=\; y_i, \quad i = 1, 2, 3, ..., n.$$

Denote $r(t) = p(t) - q(t)$. Then we can see that $r(t) \in P_{n-1}$, i.e., $r(t)$ is a polynomial of degree at most $n-1$, and $r(t_i) = p(t_i) - q(t_i) = 0,\ i = 1, 2, 3, ..., n$. Therefore $r(t)$ has at least n distinct zeros and has to be exactly zero since it is a polynomial of degree at most $n-1$. This proves that the two interpolation polynomials $p(t)$ and $q(t)$ must be the same polynomial, i.e., the uniqueness of the interpolation polynomial.

Therefore the Lagrange polynomial functions $L_k(t),\ k = 1, 2, ..., n$, form a basis of P_{n-1}, since any polynomial in P_{n-1} can be represented uniquely as a linear combination of those Lagrange polynomial functions.

The existence and uniqueness of the interpolation polynomial also shows that the Vandermonde matrix formed in the previous section for the case of using monomial basis is non-singular.

The Lagrange formula provides an explicit formula of the interpolation polynomials. However, it is difficult to apply this formula in many applications. For example, in order to evaluate the interpolation polynomial $p_{n-1}(t)$ in the Lagrange formula at a point t, n

terms need to be evaluated and to evaluate each single term requires $O(n)$ flops, which is in total $O(n^2)$ flops. Also it is difficult to compute the derivatives or integrals of the interpolation polynomial in the Lagrange formula.

Compared with the Lagrange basis functions, one advantage of using the monomial basis is that it is very efficient to evaluate the function value

$$p_{n-1}(t) = x_1 + x_2 t + x_3 t^2 + ... + x_n t^{n-1}$$

at any point t, which can be proceeded in the following manner

$$p_{n-1}(t) = x_1 + t * (x_2 + t * (x_3 + t * (\ldots + t * (x_{n-1} + t * x_n) \ldots))).$$

and only requires n additions and n multiplications in total. Also the differentiation and integration of $p_{n-1}(t)$ in the monomial basis is simple to compute.

3.2 Another basis

Given any set of n distinct interpolation points $(t_i, y_i), i = 1, 2, \ldots, n$, $t_1 < t_2 < \ldots < t_n$, define the following functions in P_{n-1},

$$\begin{aligned}
N_1(t) &= 1, \\
N_2(t) &= (t - t_1), \\
N_3(t) &= (t - t_1)(t - t_2), \\
N_4(t) &= (t - t_1)(t - t_2)(t - t_3), \\
&\ldots \\
N_n(t) &= (t - t_1)(t - t_2) \ldots (t - t_{n-1}),
\end{aligned}$$

and in general, for any $k = 1, 2, \ldots, n$,

$$N_k(t) = (t - t_1)(t - t_2) \ldots (t - t_{k-1}).$$

We note that, for any $k = 1, 2, \ldots, n$,

$$N_k(t_i) \begin{cases} = 0, & i = 1, 2, \ldots, k-1, \\ \neq 0, & i = k, k+1, \ldots, n. \end{cases}$$

Let us determine an interpolation function $p_{n-1}(t) \in P_{n-1}$ of the form

$$p_{n-1}(t) = x_1 N_1(t) + x_2 N_2(t) + x_3 N_3(t) + \ldots + x_n N_n(t) = \sum_{k=1}^{n} x_k N_k(t),$$

such that

$$p_{n-1}(t_i) = \sum_{k=1}^{n} x_k N_k(t_i) = y_i, \quad i = 1, 2, \ldots, n,$$

which leads to the following system of linear equation

$$B\mathbf{x} = \begin{bmatrix} 1 & N_2(t_1) & N_3(t_1) & & N_n(t_1) \\ 1 & N_2(t_2) & N_3(t_2) & \dots & N_n(t_2) \\ 1 & N_2(t_3) & N_3(t_3) & & N_n(t_3) \\ & \vdots & & \ddots & \\ 1 & N_2(t_n) & N_3(t_n) & \dots & N_n(t_n) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix}.$$

Then from the property that for any $k = 1, 2, \dots, n$,

$$N_k(t_i) \begin{cases} = 0, & i = 1, 2, \dots, k-1, \\ \neq 0, & i = k, k+1, \dots, n, \end{cases}$$

we know that the coefficient matrix B is lower triangular and invertible. The solution of the above system can be done by the forward substitution and costs $O(n^2)$ flops.

The evaluation of $p_{n-1}(t) = x_1N_1(t) + x_2N_2(t) + x_3N_3(t) + \ldots + x_nN_n(t)$ at any point t, can be proceeded in the following manner

$$p_{n-1}(t) = x_1 + (t-t_1) * \{x_2 + (t-t_2) * \{x_3 + (t-t_3) * \{\ldots + (t-t_{n-2}) * \{x_{n-1} + (t-t_{n-1}) * x_n\} \ldots\}\}\}$$

and costs n additions, n substraction, and n multiplications in total.

4 Accuracy of polynomial interpolation

Given n interpolation points $(t_i, y_i), i = 1, 2, ..., n$, e.g., $y_i = g(t_i)$, we can determine a unique polynomial $p_{n-1}(t) \in P_{n-1}$, such that $p_{n-1}(t_i) = g(t_i)$, $i = 1, 2, ..., n$. The interpolation polynomial represents a certain approximation of $f(x)$. They have the same values on those interpolation points. The question is how much is the difference between $p_{n-1}(t)$ and $g(t)$ elsewhere.

The following theorem tell us that theoretically we can find an arbitrarily accurate polynomial approximation to any given continuous function.

Theorem 1 *(Weierstrass Approximation Theorem) Let $g(t)$ be continuous on a finite closed interval $[a, b]$. For any positive value ϵ, there always exists a polynomial $p(t)$ of a certain degree, such that*

$$\max_{t \in [a,b]} |g(t) - p(t)| \leq \epsilon.$$

This theorem tells us that any continuous function on a closed interval $[a, b]$ can be approximated arbitrarily well by a polynomial. But when ϵ is small, the degree of the polynomial $p(t)$ can be very high. Also this theorem says nothing about how to find such a polynomial from a given set of finite fitting points.

For the interpolation polynomial, we have the following theorem on its accuracy to interpolate a function.

Theorem 2 *Let $g(t)$ be a function on an interval $[a,\ b]$ and $g^{(n)}(t)$ be continuous on $[a,\ b]$. Let $p_{n-1}(t) \in P_{n-1}$ be a polynomial of degree at most $n-1$ which interpolates a set of n points $(t_i,\ g(t_i))$, for $i = 1, ..., n$, where $a = t_1 < ... < t_n = b$. Then for any $t \in [a,\ b]$, there exists a $\xi \in [a,\ b]$, such that*

$$g(t) - p_{n-1}(t) \ = \ \frac{g^{(n)}(\xi)}{n!}\ (t-t_1)\ (t-t_2)\ ...\ (t-t_n).$$

Proof: It is trivial for the case $t = t_i$, for which $g(t_i) - p_{n-1}(t_i) = 0$. In the following we assume that $t \neq t_i$, for $i = 1, 2, ..., n$.

Let us define a function $F(z)$ on the interval $[a,\ b]$ by

$$F(z) = g(z) - p_{n-1}(z) - K(z-t_1)(z-t_2)...(z-t_n)$$

where the constant K is chosen as

$$K = \frac{g(t) - p_{n-1}(t)}{(t-t_1)\ (t-t_2)\ ...\ (t-t_n)}.$$

Then we can see that $F(t) = 0$, as well as $F(t_i) = 0$, for $i = 1, 2, ..., n$.

Since the function $F(z)$ has at least $n+1$ distinct zeros on the interval $[a,\ b]$ and $F^{(n)}(z)$ is continuous on $[a,\ b]$, we know that there exists a certain $\xi \in [a,\ b]$, such that

$$F^{(n)}(\xi) = 0,$$

i.e.,

$$g^{(n)}(\xi) - Kn! = 0.$$

Together with the definition of K, we have

$$g(t) - p_{n-1}(t) = \frac{g^{(n)}(\xi)}{n!}\ (t-t_1)\ (t-t_2)\ ...\ (t-t_n).$$

□

The difference between the interpolation polynomial and the original function depends on the derivative $g^{(n)}(t)$. If $g^{(n)}(t)$ is large, the difference can in fact be large. This is what happened when interpolating the Runge's function

$$g(t) = (1 + 25t^2)^{-1},$$

by using equally spaced points. In this case

$$|g^{(n)}(t)| \approx n! \left(\frac{50t}{1+25t^2}\right)^n.$$

If $t = 1/5$, we can see that $|R^{(n)}(1/5)| \approx 5^n\ n! \ \to \infty, \quad \text{as } n \to \infty.$

Another issue of applying high order polynomial interpolation is due to the ill-conditioning of the problem. For example, to determine the coefficient of the interpolation polynomial

on the monomial basis, we need to solve a system of linear equation with the Vandermonde matrix. However the Vandermonde matrix can be very ill-conditioned, which make the solution less accurate. For example, when we interpolate the sine function using high order polynomials on equally spaced points, the difference between the interpolation polynomial and the original sine function can still be large for large n even though the high order derivatives of sine are all bounded.

5 Orthogonal Polynomials

Orthogonal polynomials play an important role in the interpolation theory. For eaxmple, there is one remedy in the polynomial interpolation of Runge's function, where we can choose the fitting points as the roots of Chebyshev polynomials, which are more dense close to the two end of the interval and less crowded in the middle.

5.1 Legendre polynomials

Consider a set of polynomials $p_i(t)$, for $i = 0, 1, 2, ...$, on an interval $[a, b]$, where each $p_i(t)$ is a polynomial of degree i. We say two polynomials $p_i(t)$ and $p_j(t)$ are orthogonal, if

$$(p_i, p_j) = \int_a^b p_i(t)p_j(t)w(t) = 0,$$

where $w(t)$ is a certain given weight function in the inner product.

For example, taking $a = -1$, $b = 1$, and the weight function $w(t) = 1$, we have the following set of orthogonal polynomials

$$P_0(t) = 1, \quad P_1(t) = t, \quad P_2(x) = \frac{3}{2}t^2 - \frac{1}{2}, \quad P_3(t) = \frac{5}{2}t^3 - \frac{3}{2}t, \quad ...$$

which satisfy

$$\int_{-1}^{1} P_i(t)P_j(t) = 0, \quad i \neq j.$$

They are called the *Legendre polynomials*. The first n Legendre polynomials also form a basis of the polynomial space P_{n-1}.

5.2 Chebyshev polynomial

Another set of orthogonal polynomials are the Chebyshev polynomials. The kth Chebyshev polynomial, for $k = 0, 1, 2, \ldots$, is defined by

$$T_k(t) = \cos(k \arccos(t)), \quad t \in [-1,\ 1].$$

When $k = 0$ and 1, we have respectively

$$T_0(t) = 1, \quad T_1(t) = t.$$

Then recursively, we have

$$T_{k+1}(t) = 2tT_k(t) - T_{k-1}(t),$$

which can be derived from

$$\begin{aligned}
& T_{k+1}(t) + T_{k-1}(t) \\
= \; & \cos\left((k+1)\arccos(t)\right) + \cos\left((k-1)\arccos(t)\right) \\
= \; & 2\cos\left(\frac{(k+1)\arccos(t) + (k-1)\arccos(t)}{2}\right)\cos\left(\frac{(k+1)\arccos(t) - (k-1)\arccos(t)}{2}\right) \\
= \; & 2\cos(\arccos(t))\cos(k\arccos(t)) = 2tT_k(t).
\end{aligned}$$

From the three-term recurrence of the Chebyshev polynomials, we know that $T_k(t)$ is a polynomial of degree k, and the Chebyshev polynomials also provide a basis of the polynomial space.

The Chebyshev polynomials satisfy the following orthogonality

$$\int_{-1}^{1} \frac{T_i(t)T_j(t)}{\sqrt{1-t^2}}\, dt = 0, \quad i \neq j.$$

The zeros of a Chebyshev polynomial $T_k(t)$ are given by

$$k\arccos(t) = \frac{\pi}{2},\ \frac{3\pi}{2},\ \frac{5\pi}{2},\ \ldots,\ \frac{(2k-1)\pi}{2}.$$

i.e.,

$$t_i = \cos\frac{(2i-1)\pi}{2k}, \qquad i = 1,\ 2,\ \ldots,\ k.$$

If we choose the interpolation points at the zeros of a Chebyshev points, which are more clustered at the ends of the interval $[-1, 1]$, then the oscillation with the Runge's function interpolation disappear. We can understand this by saying that by placing the interpolation points at the zeros of Chebyshev polynomial, the n-th derivative of the Runge's function, $R^{(n)}(\xi)$ is relatively small, due to the position of ξ. Mathematically, we can think that the Runge's function is a complex function, which has a pole (singularity) at the point $\pm i/5$, which is just off the interval $[-1, 1]$. This singularity make the equally spaced interpolation worse when there are more points close to the pole, where you can think the derivatives close to the pole are infinite. The Chebyshev points instead cluster at the ends of the interval, away from the singularity.

6 Piecewise polynomial interpolation

A more practical way to avoid the instability of the high order polynomial interpolation is to apply the piecewise polynomial interpolation.

6.1 Piecewise linear interpolation

Piecewise linear interpolation is simple, where we use a straight line to connect the neighboring points $[t_i, y_i]$ and $[t_{i+1}, y_{i+1}]$. Or in words, for a given set of points $[t_i, y_i]$, $i = 1, 2, 3, ..., n$, we find a local linear interpolation on each small interval $[t_i, t_{i+1}]$.

We can write out the interpolation function by

$$L(t) = y_i \frac{t - t_{i+1}}{t_i - t_{i+1}} + y_{i+1} \frac{t - t_i}{t_{i+1} - t_i}, \quad \text{for} \quad t_i \le t \le t_{i+1}, \quad i = 1, 2, ..., n-1.$$

We can see that at each point t_i, $L(t_i) = y_i$, and the interpolation error at any other point essentially depends on the length of the intervals $[t_i, t_{i+1}]$. We also see that the interpolation function $L(t)$ is continuous on the interval $[t_1, t_n]$. But its derivative may not be continuous.

6.2 Piecewise cubic interpolation

A smoother piecewise polynomial interpolation is the piecewise cubic interpolation. Given a set of points $[t_i, y_i]$, $i = 1, 2, 3, ..., n$, we find on each interval $[t_i, t_{i+1}]$ a cubic polynomial interpolation with some continuity conditions of them at the interpolation points.

Let us denote the cubic interpolation polynomial on each interval $[t_i, t_{i+1}]$ by

$$p_i(t) = a_i + b_i t + c_i t^2 + d_i t^3.$$

In total we have $n - 1$ intervals, and therefore $4(n - 1)$ unknowns to determine for the whole interpolation on the $n - 1$ intervals $[t_i, t_{i+1}]$, for $i = 1,\ 2,\ ...,\ n - 1$.

On each interval $[t_i, t_{i+1}]$, we require that

$$p_i(t_i) = y_i, \quad p_i(t_{i+1}) = y_{i+1}, \quad i = 1, 2, ..., n-1$$

which give $2(n - 1)$ equations on the $n - 1$ intervals.

We also require the piecewise interpolation functions have continuous first and second order derivatives at the interior interpolation points, i.e.,

$$p'_{i-1}(t_i) = p'_i(t_i), \quad p''_{i-1}(t_i) = p''_i(t_i), \quad i = 2,\ ...,\ n - 1,$$

which give another $2(n - 2)$ equations to determined the $4n - 4$ unknowns.

The last 2 equations are given, e.g., by requiring

$$p'_1(t_1) = p'_{n-1}(t_n) = 0.$$

For example, given three points (t_i, y_i) by $(-1, 0)$, $(0, 1)$, $(1, 0)$, which contains two intervals $[-1, 0]$ and $[0, 1]$, we can determine a pair of cubic polynomials $s_1(t) = a_1 + a_2 t + a_3 t^2 + a_4 t^3$ for $t \in [-1, 0]$, and $s_2(t) = a_5 + a_6 t + a_7 t^2 + a_8 t^3$ for $t \in [0, 1]$, such that the above conditions are satisfied. This lead to a system of linear equations for the unknowns a_i, $i = 1, 2, \ldots, 8$, which gives the solution $s_1(t) = 1 - 3t^2 - 2t^3$ for $t \in [-1, 0]$, and $s_2(t) = 1 - 3t^2 + 2t^3$ for $t \in [0, 1]$.

Some MATLAB functions related to the polynomial interpolation and piecewise cubic interpolations are *polyfit* and *spline*. For example, using the MATLAB *spline* function as the following draws the curve corresponding to the piecewise cubic interpolation for the above example.

```
>>  t = [-1 0 1];

>>  y = [ 0 1 0];

>>  pp = spline(t,y);

>>  grid = linspace(-1, 1, 50);

>>  plot(grid, ppval(pp, grid), '-*', 'LineWidth', 2);

>>  hold on;

>>  plot(t,y,'LineWidth', 2);
```