CP315-Portfolio2-Yiqian-Kang

Yiqian Kang

March 1 2024

7 Interpolation, Curve Fitting

7.1 Vandermonde-based collocation

The Maple code for the Vandermonde-based collocation is:

```
with(LinearAlgebra);
Vander := proc(f, llist)
    local y, a, B, R, L, v;
    y := <map(f, [seq(v, v in llist)])>;
    a := 0;
    B := VandermondeMatrix(llist);
    R := ReducedRowEchelonForm(<B | y>);
    L := convert(R, list)[-numelems(llist) .. -1];
    for v to numelems(llist) do
        a := a + L[v]*x^(v - 1);
    end do;
end proc;
```

This is the testing code for this Vandermonde-based collocation is:

```
f := x -> 30/(1 + 3*x^2);
llist := [-3, -2, -1, 0, 1, 2, 3];
poly:= Vander(f, llist);
print(poly);
map(f, [seq(i, i in llist)]);
```

For the function$f(x) := \frac{30}{1+3x^2}$:
By using the given 7 data point, The interpolation polynomial of degree 6 is:

$$30 - \frac{5445}{182}x^2 + \frac{5805}{728}x^4 - \frac{405}{728}x^6$$

Finally, by using the map function we can get the point

$$\left[\frac{15}{14}, \frac{30}{13}, \frac{15}{2}, 30, \frac{15}{2}, \frac{30}{13}, \frac{15}{14}\right]$$

7.2 Lagrange interpolation

The Maple code for the Lagrange interpolation is:

```
with(LinearAlgebra);
laginter := proc(f, llist)
    local L, a, n, i, j;
    a := 0;
    n := numelems(llist);
    for i to n do
        L := 1;
        for j to n do
            if i <> j then
                L := L*(x - llist[j])/(llist[i] - llist[j]);
            end if;
        end do;
        a := a + f(llist[i])*L;
    end do;
    a := simplify(a);
    return a;
end proc;
```

This is the testing code for this Lagrange interpolation is:

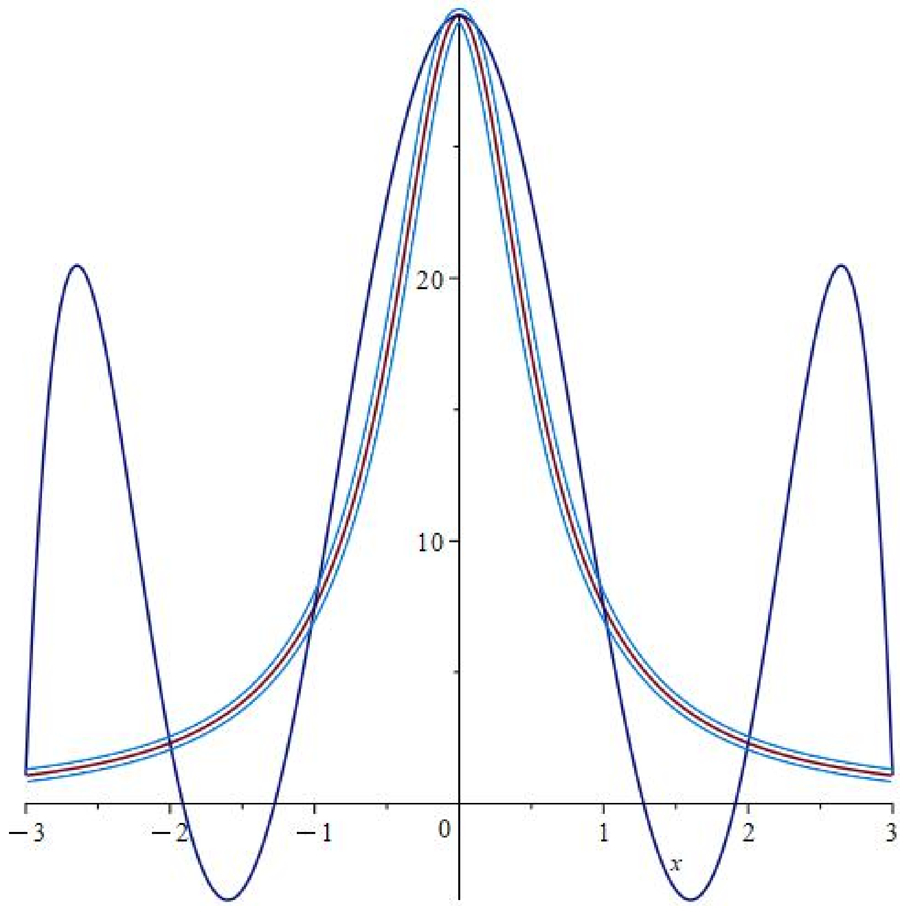
```
f := x -> 30/(1 + 3*x^2);
llist := [-3, -2, -1, 0, 1, 2, 3];
poly:= laginter(f, llist);
print(poly);
map(f, [seq(i, i in llist)]);
```

For the function $f(x) := \frac{30}{1+3x^2}$:
By using the given 7 data point, The interpolation polynomial of degree 6 is:

$$30 - \frac{5445}{182}x^2 + \frac{5805}{728}x^4 - \frac{405}{728}x^6$$

Finally, by using the map function we can get the point

$$\left[\frac{15}{14}, \frac{30}{13}, \frac{15}{2}, 30, \frac{15}{2}, \frac{30}{13}, \frac{15}{14}\right]$$



7.3 Comparison of the function and the interpolation polynomial(s)

We have showed that the 2 functions we derived from different methods are same in 7.1 and 7.2.

By using

$$f(x) := \frac{30}{1+3x^2}$$

$$g(x) := 30 - \frac{5445}{182}x^2 + \frac{5805}{728}x^4 - \frac{405}{728}x^6$$

plot([f(x), g(x)], x = -3 .. 3)
We can see the plot for f(x) and g(x)

7.4 2D Lagrange interpolation

The Maple code for the 2D Lagrange interpolation is:

```
lagrangeinter := proc(mmtrix, Xlist, Ylist)
    local m, n, i, j, k, l, Li, Lj, a;
    m := numelems(Xlist) - 1;
    n := numelems(Ylist) - 1;
    a := 0;
    for i from 0 to m do
        for j from 0 to n do
            Li := 1;
            for k from 0 to m do
                if k <> i then
                    Li := Li*(x - Xlist[k + 1])/(Xlist[i + 1] - Xlist[k + 1]);
                end if;
            end do;
            Lj := 1;
            for l from 0 to n do
                if l <> j then
                    Lj := Lj*(y - Ylist[l + 1])/(Ylist[j + 1] - Ylist[l + 1]);
                end if;
            end do;
            a := res + Li*Lj*mmtrix[i + 1, j + 1];
        end do;
    end do;
    return simplify(a);
end proc;
```

This is the testing code for this 2D Lagrange interpolation is:

```
mmtrix := Matrix([[1, 2, 7], [7, 20, 54], [54, 148, 403]]);
Xlist := [0, 1, 2];
Ylist := [0, 2, 4];
poly := lagrangeinter(mmtrix, Xlist, Ylist);
eval(poly, {x = 1.5, y = 3});
```

The interpolation polynomial L(x, y) for 2D Lagrange is

$$\frac{\left(123y^2+30y+328\right)x^2}{16}+\frac{\left(-89y^2-2y-232\right)x}{16}+\frac{y^2}{2}-\frac{y}{2}+1$$

the estimate for the function value f (1.5, 3) is: 121.0468750.

8 Non-Linear Regression

We aim to approximate a complex trigonometric function using non-linear regression. The function in question is defined as:

$$f(x) = \cos(x^2\pi) + \cos(2x^2) + \sin(x^2\pi) + \sin(2x^2) \tag{1}$$

over the interval $[0, 4]$. The approximation process involves the following steps:

1. Define the number of data points ($n = 50$) and frequency terms (fre $=$ 100).
2. Generate x values evenly distributed over the interval and compute corresponding y values for the function.
3. Construct a model using a series of sine and cosine functions with coefficients to be determined.
4. Compute the sum of squared differences between the model predictions and actual y values.
5. Differentiate this sum with respect to each coefficient and solve the resulting system of equations to find the optimal coefficients.
6. Plot the original function and the regression model for visual comparison.

The Maple code for these steps is presented below:

```
restart;
n := 50;
fre := 100;
x := [seq(Pi*i/n, i = 1 .. n)];
y := [seq(evalf(cos((Pi*i/n)^2*Pi) + sin((Pi*i/n)^2*Pi) + cos(2*(Pi*i/n)^2) + sin(2*(Pi*i/n)
omega := Pi/10;
y := cos(x^2*Pi) + cos(2*x^2) + sin(x^2*Pi) + sin(2*x^2)

for i to n do
    sums := 0;
    k := 1;
    for j to fre do
        if j mod 2 <> 0 then sums := sums + eval(cat(a, j))*sin(k*omega*x[i]);
        else sums := sums + eval(cat(a, j))*cos(k*omega*x[i]); k := k + 1; end if;
    end do;
    assign(cat(e, i), y[i] - sums);
end do
//computes the error for each data point

S := 0;
for i to n do
```

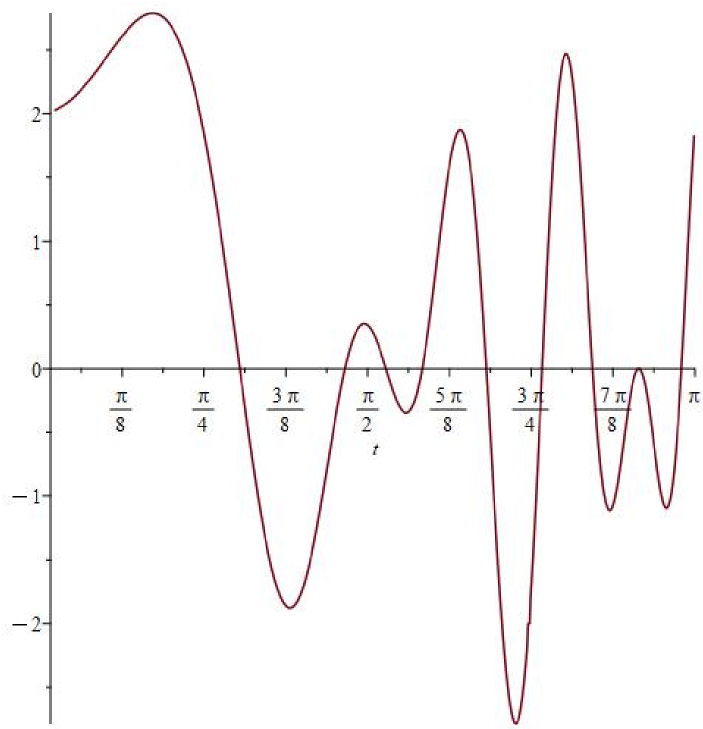
```
    S := e[i]^2 + S;
end do;
S := evalf(S);
//Calculates the sum of squared errors

for i to fre do
    Sa[i] := diff(S, eval(cat('a', i)));
end do
//Differentiates the sum of squared errors with respect to each coefficient a[i]

eqs := 0;
k := 1;
for j to fre do
    if j mod 2 <> 0 then eqs := eqs + eval(cat(a, j))*sin(k*omega*z); else eqs := eqs + eval
end do;
eqs := evalf(eqs);
//Solves the equations

dataPoints := plot([seq([x[i], y[i]], i = 1 .. n)], style = point);
lb := min(op(x));
ub := max(op(x));
quadr := plot(eqs, z = lb .. ub, color = blue);
plots[display]({dataPoints, quadr});
// Plots the regression model
plot(cos(t^2*Pi) + cos(2*t^2) + sin(t^2*Pi) + sin(2*t^2), t = Pi/50 .. Pi)
```

This approach allows for the approximation of trigonometric functions with arbitrary complexity, adjusting the number of frequency terms as necessary to achieve a desired level of accuracy.


2
1
0
−1
−2
$\frac{\pi}{8}$
$\frac{\pi}{4}$
$\frac{3\pi}{8}$
$\frac{\pi}{2}$
$\frac{5\pi}{8}$
$\frac{3\pi}{4}$
$\frac{7\pi}{8}$
π
t