Spline Interpolation

Yi-Chen Zhang

September 27, 2018

Class of Polynomial Functions

$$p(x) = a_0 + a_1 x + \cdots + a_m x^m$$

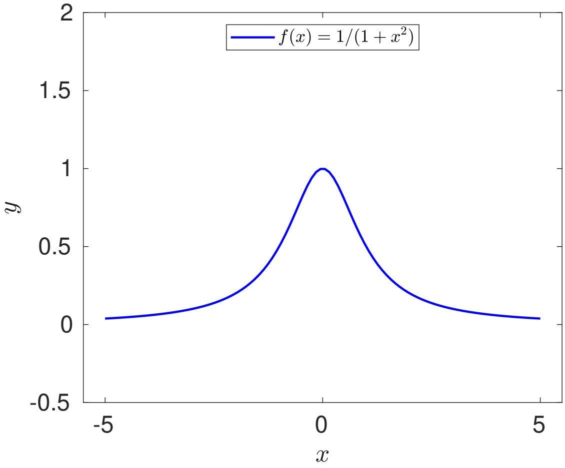
Pros:

- The most common class of functions
- Have a simple form
- Well known and understood properties
- Moderate flexibility of shapes
- Computationally easy to use

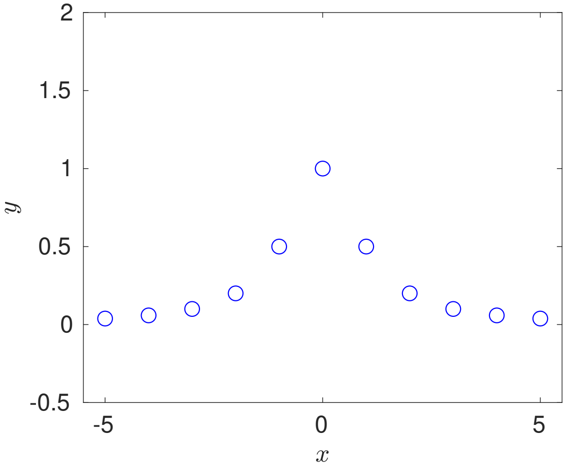
Cons:

- Poor interpolatory properties
- Poor extrapolatory properties
- Poor asymptotic properties
- Have a shape/degree trade off

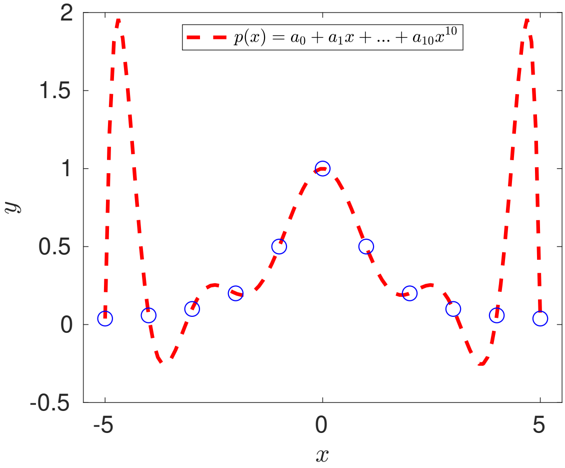
Runge's Example



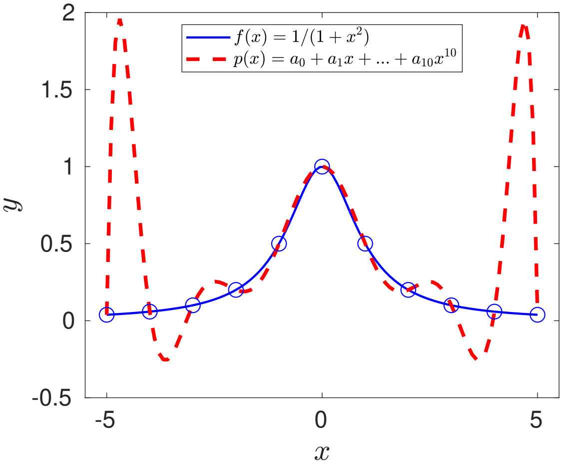
Runge's Example



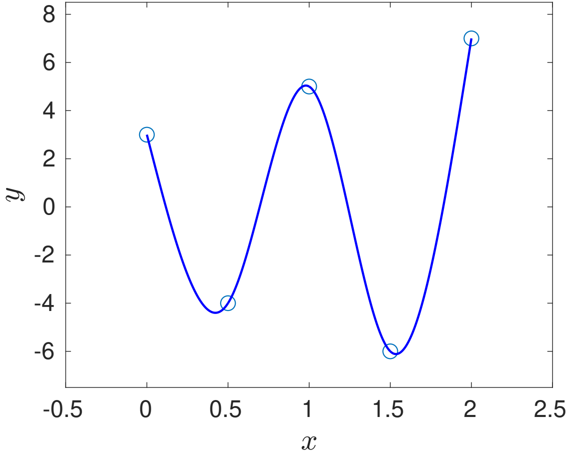
Runge's Example



Runge's Example



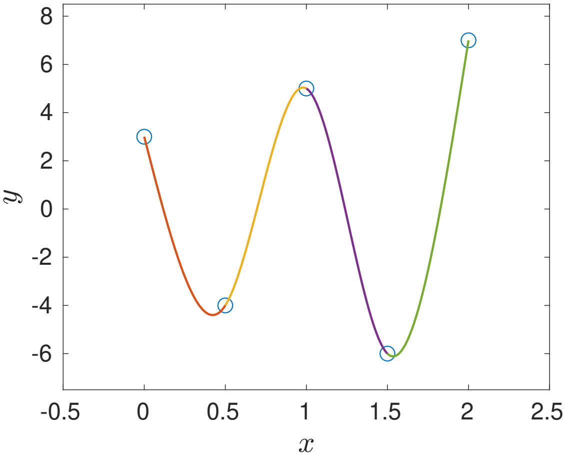
Another Example



- In general, it is not wise to use a high-degree interpolating polynomial to approximate a function on an interval $[a, b]$ unless this interval is sufficiently small

Idea:

- Partition the domain into small numbers of subdomains
- Fit a polynomial on each subdomain


8
6
4
2
0
-2
-4
-6
y
-0.5
0
0.5
1
1.5
2
2.5
x

Piecewise Polynomial

Let $a = x_1 < \cdots < x_n = b$.
A **piecewise polynomial** is a function $p(x)$ defined on $[a, b]$ by

$$p(x) = p_i(x), \quad x_i \leq x < x_{i+1}, \quad i = 1, \ldots, n-1$$

where each function $p_i(x)$ is a polynomial defined on $[x_i, x_{i+1})$.

- Typically, piecewise polynomials are not smooth on the breakpoints.
- In order to fit a smooth function we have to impose a certain number of continuous derivatives on the breakpoints.

Spline Interpolation

- A spline is a piecewise polynomial of degree q between knots (or breakpoints) that has $q-1$ continuous derivatives at the knots
- The most commonly used spline is a *cubic spline* (that is $q=3$)
- Let $a = x_1 < \cdots < x_n = b$ be the knot locations
- Let y_i be the corresponding output value at x_i

A **cubic spline** is a piecewise polynomial $s(x)$ that satisfies the following properties:

1. The spline $s(x) = s_i(x)$ if $x \in [x_i, x_{i+1})$, for $i = 1, \ldots, n-1$, where $s_i(x)$ is a cubic polynomial of the form:

$$s_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

2. $s(x_i) = y_i$ for $i = 1, \ldots, n$
3. $s(x)$ is continuous on (a, b)
4. $s(x)$ is twice continuously differentiable on (a, b).
   $\Rightarrow s(x)$ is twice differentiability at the knots.

Constructing Cubic Spline

By the first and second properties, $s(x)$ interpolate the data.

- That is, $s(x_i) = y_i$ for $i = 1, \ldots, n$
- Since $s(x) = s_i(x)$ for $x \in [x_i, x_{i+1})$

$$\begin{aligned} s_i(x_i) &= a_i + b_i(x_i - x_i) + c_i(x_i - x_i)^2 + d_i(x_i - x_i)^3 \\ \Rightarrow y_i &= a_i \end{aligned} \tag{1}$$

By the third property, $s(x)$ is continuous on $[a, b]$

- That is $s_i(x_{i+1}) = s_{i+1}(x_{i+1})$ for $i = 1, \ldots, n-1$
- For each x_{i+1}, $i = 1, \ldots, n-1$, we have

$$s_i(x_{i+1}) = a_i + b_i(x_{i+1} - x_i) + c_i(x_{i+1} - x_i)^2 + d_i(x_{i+1} - x_i)^3$$

- Note that $s_{i+1}(x_{i+1}) = a_{i+1}$
- $h_{i+1} = x_{i+1} - x_i$, then we have

$$a_{i+1} = a_i + b_i h_{i+1} + c_i h_{i+1}^2 + d_i h_{i+1}^3 \tag{2}$$

By the forth property, $s^{'}(x)$ is continuous on (a, b)

- That is $s_i^{'}(x_{i+1}) = s_{i+1}^{'}(x_{i+1})$, for $i = 1, \ldots, n-1$
- For each x_{i+1}, $i = 1, \ldots, n-1$, we have

$$s_i^{'}(x_{i+1}) = b_i + 2c_i(x_{i+1} - x_i) + 3d_i(x_{i+1} - x_i)^2$$

- Note that $s_{i+1}^{'}(x_{i+1}) = b_{i+1}$
- Thus, we have

$$b_{i+1} = b_i + 2c_ih_{i+1} + 3d_ih_{i+1}^2 \tag{3}$$

By the forth property again, $s^{''}(x)$ is continuous on (a, b)

- That is $s_i^{''}(x_{i+1}) = s_{i+1}^{''}(x_{i+1})$, for $i = 1, \ldots, n-1$
- For each x_{i+1}, $i = 1, \ldots, n-1$, we have

$$s_i^{''}(x_{i+1}) = 2c_i + 6d_i(x_{i+1} - x_i)$$

- Note that $s_{i+1}^{''}(x_{i+1}) = 2c_{i+1}$
- Thus, we have

$$c_{i+1} = c_i + 3d_i h_{i+1} \tag{4}$$

Alternative formulation

- Let $M_i = s_i^{''}(x_i)$, conditions (1) to (4) are

$$
\begin{aligned}
a_i &= y_i \\
b_i &= \frac{y_{i+1} - y_i}{h_{i+1}} - \left(\frac{M_{i+1} + 2M_i}{6}\right) h_{i+1} \\
c_i &= \frac{M_i}{2} \\
d_i &= \frac{M_{i+1} - M_i}{6h_{i+1}}
\end{aligned}
$$

- If we know M_i, we can get back a_i, b_i, c_i, and d_i since y_i and h_{i+1} are given from the data.

- Plugging b_i, c_i, and d_i into equation (3) $(b_{i+1} = b_i + 2c_i h_{i+1} + 3d_i h_{i+1}^2)$, we have

$$h_{i+1}M_i + 2(h_{i+1} + h_{i+2})M_{i+1} + h_{i+2}M_{i+2} = 6\left(\frac{y_{i+2} - y_{i+1}}{h_{i+2}} - \frac{y_{i+1} - y_i}{h_{i+1}}\right),$$

for $i = 1, 2, \ldots, n-2$
- Note that these are $n-2$ linear equations for n unknowns $\boldsymbol{M} = (M_1, \ldots, M_n)^T$, where $M_i = s_i^{''}(x_i)$.

$$
\begin{pmatrix}
h_2 & 2(h_2+h_3) & h_3 & 0 & \cdots & 0 & 0 \\
0 & h_3 & 2(h_3+h_4) & h_4 & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \cdots & h_{n-1} & 0 \\
0 & 0 & 0 & 0 & \cdots & 2(h_{n-1}+h_n) & h_n
\end{pmatrix}
\begin{pmatrix}
M_1 \\ M_2 \\ \vdots \\ M_{n-1} \\ M_n
\end{pmatrix}
$$

$$
= 6
\begin{pmatrix}
\frac{1}{h_2} & -\frac{1}{h_2} - \frac{1}{h_3} & \frac{1}{h_3} & 0 & \cdots & 0 & 0 \\
0 & \frac{1}{h_3} & -\frac{1}{h_3} - \frac{1}{h_4} & \frac{1}{h_4} & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \cdots & \frac{1}{h_{n-1}} & 0 \\
0 & 0 & 0 & 0 & \cdots & -\frac{1}{h_{n-1}} - \frac{1}{h_n} & \frac{1}{h_n}
\end{pmatrix}
\begin{pmatrix}
y_1 \\ y_2 \\ \vdots \\ y_{n-1} \\ y_n
\end{pmatrix}
$$

- Tridiagonal linear system with $n-2$ rows and n columns
- That is, $n-2$ linear equations for n unknowns. There is no unique solution
- To generate a unique cubic spline function given the data, we need to impose two additional conditions
- Three common choices:
  1. Natural cubic spline
     - Force the second derivatives at the endpoints to be zero, i.e., linear beyond the endpoints: $M_1 = M_n = 0$
  2. Clamped spline
     - Force the first derivatives at the end points such as $s^{'}(x_1) = A$ and $s^{'}(x_n) = B$.
  3. Cubic runout spline
     - Assign $M_1 = 2M_2 - M_3$ and $M_n = 2M_{n-1} - M_{n-2}$
- We will focus on Natural cubic spline

Natural Cubic Spline

$$\begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ h_2 & 2(h_2+h_3) & h_3 & 0 & \cdots & 0 & 0 \\ 0 & h_3 & 2(h_3+h_4) & h_4 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & h_{n-1} & 0 \\ 0 & 0 & 0 & 0 & \cdots & 2(h_{n-1}+h_n) & h_n \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} M_1 \\ M_2 \\ \vdots \\ M_{n-1} \\ M_n \end{pmatrix}$$

$$= 6 \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ \frac{1}{h_2} & -\frac{1}{h_2} - \frac{1}{h_3} & \frac{1}{h_3} & 0 & \cdots & 0 & 0 \\ 0 & \frac{1}{h_3} & -\frac{1}{h_3} - \frac{1}{h_4} & \frac{1}{h_4} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \frac{1}{h_{n-1}} & 0 \\ 0 & 0 & 0 & 0 & \cdots & -\frac{1}{h_{n-1}} - \frac{1}{h_n} & \frac{1}{h_n} \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n-1} \\ y_n \end{pmatrix}$$

Since $c_i = \frac{M_i}{2}$, we can further simply above equation:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ h_2 & 2(h_2+h_3) & h_3 & 0 & \cdots & 0 & 0 \\ 0 & h_3 & 2(h_3+h_4) & h_4 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & h_{n-1} & 0 \\ 0 & 0 & 0 & 0 & \cdots & 2(h_{n-1}+h_n) & h_n \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n-1} \\ c_n \end{pmatrix}$$

$$= 3 \begin{pmatrix} 0 \\ \frac{1}{h_3}(y_3 - y_2) - \frac{1}{h_2}(y_2 - y_1) \\ \vdots \\ \frac{1}{h_n}(y_n - y_{n-1}) - \frac{1}{h_{n-1}}(y_{n-1} - y_{n-2}) \\ 0 \end{pmatrix}$$

Summary of Spline Interpolation

Consider the data points (x_1, y_1), (x_2, y_2), $\ldots$, (x_n, y_n).

- Calculate the values $h_{i+1} = x_{i+1} - x_i$ for $i = 1, 2, \ldots, n-1$
- Set the matrix $\boldsymbol{A}$ and right hand side vector $\boldsymbol{b}$ for the spline.
- Solve the $n \times n$ linear system $\boldsymbol{Ac} = \boldsymbol{b}$ for the $\boldsymbol{c} = (c_1, c_2, \ldots, c_n)^T$.
- Once the coefficients $c_1, c_2, \ldots, c_n$ have been determined, the remaining coefficients can be computed as follows:

$$\begin{aligned} a_i &= y_i \\ b_i &= \frac{1}{h_{i+1}}(a_{i+1} - a_i) - \frac{h_{i+1}}{3}(c_{i+1} + 2c_i) \\ d_i &= \frac{c_{i+1} - c_i}{3h_{i+1}}, \end{aligned}$$

  for $i = 1, \ldots, n-1$
- On each sub-interval $x \in [x_i, x_{i+1})$, the spline function is

$$s(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

MATLAB Code

```
n = 5;
x = [0 1/2 1 3/2 2]';
y = [3 -4 5 -6 7]';
plot(x,y,'o', 'MarkerSize', 10);
xlim([-0.5 2.5]);
ylim([-7.5 8.5]);

% calculate the difference
dx = x(2:n)-x(1:(n-1)); % This is h
dy = y(2:n)-y(1:(n-1));

% marix A
A = zeros(n,n);
A(1,1) = 1;
A(n,n) = 1;

% vector b
b = zeros(n,1);
```

```
% assign values
for i = 2:(n-1)
  A(i,i-1) = dx(i-1);
  A(i,i) = 2*(dx(i-1)+dx(i));
  A(i,i+1) = dx(i);
  b(i) = 3*(dy(i)/dx(i)-dy(i-1)/dx(i-1));
end

% solve linear equation
c = A\b;
%c = linsolve(A,b);

% calculate the polynomial coefficients
pa = y;
pc = c;
pb = dy./dx-dx.*(pc(2:n)+2*pc(1:(n-1)))/3;
pd = (pc(2:n)-pc(1:(n-1)))./(3*dx);
```

```
% plot the spline function
% s(x) = a+b(x-xi)+c(x-xi)^2+d(x-xi)^3
hold on;
for i = 1:(n-1)
  xx = linspace(x(i),x(i+1),100);
  yy = pa(i)+pb(i)*(xx-x(i))+pc(i)*(xx-x(i)).^2+pd(i)*(xx-x(i)).^3;
  plot(xx,yy);
end
```

Thank You!