Lecture 14

Interpolation/Splines

David Semeraro

University of Illinois at Urbana-Champaign

October 15, 2013

Recall

Given $n+1$ distinct points $x_0, \ldots, x_n$, and values $y_0, \ldots, y_n$, there exists a unique polynomial $p(x)$ of degree at most n so that

$$p(x_i) = y_i \quad i = 0, \ldots, n$$

Recall: Monomials

Obvious attempt: try picking

$$p(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

So for each x_i we have

$$p(x_i) = a_0 + a_1 x_i + a_2 x_i^2 + \cdots + a_n x_i^n = y_i$$

OR

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^n \\ 1 & x_1 & x_1^2 & \ldots & x_1^n \\ & & & \vdots & \\ 1 & x_n & x_n^2 & \ldots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}$$

That is,

$$a = M^{-1} y$$

Very bad matrix: terribly ill-conditioned, inverse entries are **large**
Very bad evaluation: values are **huge**

Recall: Shifted Monomials

Evaluating

$$p(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

may have **huge** values. Partial fix:

$$p(x) = a_0 + a_1(x - \bar{x}) + a_2(x - \bar{x})^2 + \cdots + a_n(x - \bar{x})^n$$

Then $M = vander(x - \bar{x})$ and

$$a = M^{-1} y$$

Still, a very bad matrix.

Recall: Lagrange

The general Lagrange form is

$$\ell_k(x) = \prod_{i=0, i\neq k}^{n} \frac{x - x_i}{x_k - x_i}$$

The resulting interpolating polynomial is

$$p(x) = \sum_{k=0}^{n} \ell_k(x) y_k$$

Example

Find the equation of a quadratic passing through the points (0,-1), (1,-1), and (2,7).

$x_0 = 0, x_1 = 1, x_2 = 2 \qquad y_0 = -1, y_1 = -1, y_2 = 7$

1. Form the Lagrange basis functions, $\ell_i(x)$ with $\ell_i(x_j) = \delta_{ij}$
2. Combine the Lagrange basis functions

$$\begin{aligned} p_2(x) &= y_0\ell_0(x) + y_1\ell_1(x) + y_2\ell_2(x) \\ &= (-1)\frac{(x-1)(x-2)}{2} + (-1)\frac{x(x-2)}{-1} + (7)\frac{x(x-1)}{2} \end{aligned}$$

Evaluate is *nice*, but *expensive*: no easy nested form.

Recall: Newton Polynomials

- Newton Polynomials are of the form

$$p_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + a_3(x - x_0)(x - x_1)(x - x_2) + \ldots$$

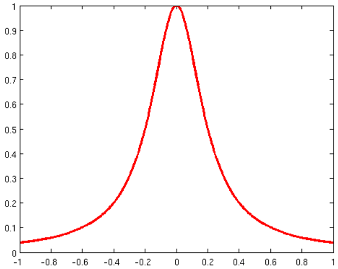
- The basis used is thus

function	order
1	0
$x - x_0$	1
$(x - x_0)(x - x_1)$	2
$(x - x_0)(x - x_1)(x - x_2)$	3

- More stable evaluation than monomials
- More computationally efficient (nested iteration) than using Lagrange

How bad is polynomial interpolation?

Let's take something very smooth function



How does interpolation behave?

Some analysis...

what can we say about

$$e(t) = f(t) - p_n(t)$$

at some point x? Consider $p = 1$: linear interpolation of a function at $x = x_0, x_1$

- want: error at x, $e(x)$
- look at

$$g(t) = e(t) - \frac{(t - x_0)(t - x_1)}{(x - x_0)(x - x_1)} e(x)$$

- $g(t)$ is 0 at $t = x_0, x_1, x$
- so $g'(t)$ is zero at two points
- so $g''(t)$ is zero at one point, call it c
- then

$$0 = g''(c) = e''(t) - 2\frac{e(x)}{(x - x_0)(x - x_1)}$$

$$= f''(t) - 2\frac{e(x)}{(x - x_0)(x - x_1)}$$

$$e(x) = \frac{(x - x_0)(x - x_1)}{2} f''(c)$$

Theorem: Interpolation Error I

If $p_n(x)$ is the (at most) n degree polynomial interpolating $f(x)$ at $n+1$ distinct points and if $f^{(n+1)}$ is continuous, then

$$e(x) = f(x) - p_n(x) = \frac{1}{(n+1)!} f^{(n+1)}(c) \prod_{i=0}^{n} (x - x_i)$$

Theorem: Bounding Lemma

Suppose x_i are equispaced in $[a, b]$ for $i = 0, \ldots, n$. Then

$$\prod_{i=0}^{n} |x - x_i| \leqslant \frac{h^{n+1}}{4} n!$$

Theorem: Interpolation Error II

Let $|f^{(n+1)}(x)| \leqslant M$, then with the above,

$$|f(x) - p_n(x)| \leqslant \frac{M h^{n+1}}{4(n+1)}$$

Fixes

We have two options:

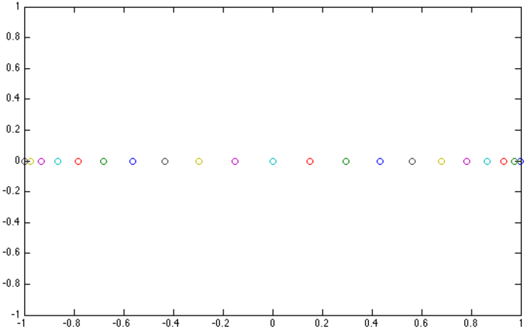
1. move the nodes: Chebychev nodes
2. piecewise polynomials (splines)

Option #1: Chebychev nodes in $[-1, 1]$

$$x_i = cos(\pi \frac{2i+1}{2n+2}), \quad i = 0, \ldots, n$$

Option #2: piecewise polynomials...

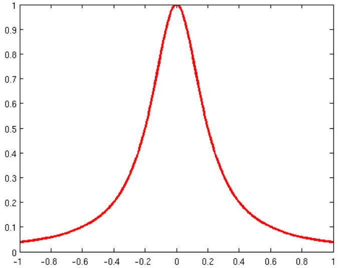
Chebychev Nodes



- Can obtain nodes from equidistant points on a circle projected down
- Nodes are non uniform and non nested

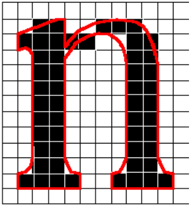
Chebychev Nodes

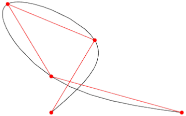
High degree polynomials using equispaced points suffer from many oscillations



- Points are bunched at the ends of the interval
- Error is distributed more evenly

Why Splines?





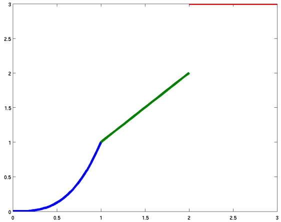
- truetype fonts, postscript, metafonts
- graphics surfaces
- smooth surfaces are needed
- how do we interpolate smoothly a set of data?
- keywords: Bezier Curves, splines, B-splines, NURBS
- basic tool: piecewise interpolation

Piecewise Polynomial

A function $f(x)$ is considered a piecewise polynomial on $[a, b]$ if there exists a (finite) partition P of $[a, b]$ such that $f(x)$ is a polynomial on each $[t_i, t_{i+1}] \in P$.

Example

$$f(x) = \begin{cases} x^3 & x \in [0, 1] \\ x & x \in (1, 2) \\ 3 & x \in [2, 3] \end{cases}$$

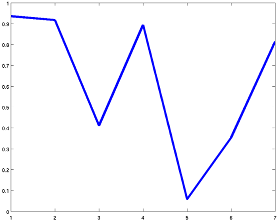


What do we want?

- we would like the piecewise polynomial to do two things
  1. interpolate (or be close to) some set of data points
  2. look nice (smooth)
- one option is called a *spline*

splines

- A *spline* is a piecewise polynomial with a certain level of smoothness.
- take Matlab: `plot(1:7,rand(7,1))`
- this is linear and continuous, but not very smooth
- the function changes behavior at *knots* $t_0, \ldots, t_n$



degree 1 spline

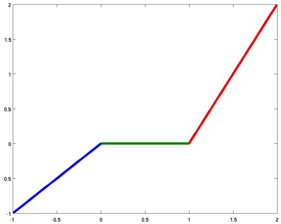
definition

A function $S(x)$ is a spline of degree 1 if:

1. The domain of $S(x)$ is an interval $[a, b]$
2. $S(x)$ is continuous on $[a, b]$
3. There is a partition $a = t_0 < t_1 < \cdots < t_n = b$ such that $S(x)$ is linear on each subinterval $[t_i, t_{i+1}]$.

Example

$$S(x) = \begin{cases} x & x \in [-1, 0] \\ 1 & x \in (0, 1) \\ 2x - 2 & x \in [1, 2] \end{cases}$$



degree 1 spline

Given data $t_0, \dots, t_n$ and $y_0, \dots, y_n$, how do we form a spline?

We need two things to hold in the interval $[a, b] = [t_0, t_n]$:

1. $S(t_i) = y_i$ for $i = 0, \dots, n$
2. $S_i(x) = a_i x + b_i$ for $i = 0, \dots, n$

Write $S_i(x)$ in point-slope form

$$\begin{aligned} S_i(x) &= y_i + m_i(x - t_i) \\ &= y_i + \frac{y_{i+1} - y_i}{t_{i+1} - t_i}(x - t_i) \end{aligned}$$

Done.

degree 1 spline

```
input t,y vectors of data
input evaluation location x
find interval i with x ∈ [t_i, t_{i+1}]
S = y_i + (x-t_i) m_i
```

degree 1 spline

Interesting:

- input $n+1$ data points $t_0, \ldots, t_n, y_0, \ldots, y_n$
- in each interval we have $S_i(x) = a_i x + b_i$
- 2 unknowns per interval $[t_i, t_{i+1}]$
- or $2n$ total unknowns
- the $n+1$ pieces of input contraints $S(t_i) = y_i$ gives 2 constraints per interval
- or $2n$ total constraints

degree 2 splines

definition

A function $S(x)$ is a spline of degree 2 if:

1. The domain of $S(x)$ is an interval $[a, b]$
2. $S(x)$ is continuous on $[a, b]$
3. $S'(x)$ is continuous on $[a, b]$
4. There is a partition $a = t_0 < t_1 < \cdots < t_n = b$ such that $S(x)$ is quadratic on each subinterval $[t_i, t_{i+1}]$.

degree 2 splines

$$S(x) = \begin{cases} S_0(x) & x \in [t_0, t_1] \\ S_1(x) & x \in [t_1, t_2] \\ \vdots & \vdots \\ S_{n-1}(x) & x \in [t_{n-1}, t_n] \end{cases}$$

for each i we have

$$S_i(x) = a_i x^2 + b_i x + c_i$$

What are a_i, b_i, c_i?

degree 2 splines

- 3 unknowns in each interval
- $3n$ total unknowns
- $2n$ constraints for matching up the input data (2 per interval)
- $n-1$ interior points require continuity of the derivative: $S_i'(x_{i+1}) = S_{i+1}'(x_{i+1})$
- but this is just $n-1$ constraints
- total of $3n-1$ constraints
- extra consraint: $S'(x_0) =$given, for example.

degree 3 spline: cubic spline

definition

A function $S(x)$ is a spline of degree 3 if:

1. The domain of $S(x)$ is an interval $[a, b]$
2. $S(x)$ is continuous on $[a, b]$
3. $S'(x)$ is continuous on $[a, b]$
4. $S''(x)$ is continuous on $[a, b]$
5. There is a partition $a = t_0 < t_1 < \cdots < t_n = b$ such that $S(x)$ is cubic on each subinterval $[t_i, t_{i+1}]$.

degree 3 spline: cubic spline

In each intervale $[t_i, t_{i+1}]$, $S(x)$ looks like

$$S_i(x) = a_{0,i} + a_{1,i}x + a_{2,i}x^2 + a_{3,i}x^3$$

- n intervals, $n+1$ knots, 4 unknowns per interval
- $4n$ unknowns

degree 3 spline: cubic spline

In each intervale $[t_i, t_{i+1}]$, $S(x)$ looks like

$$S_i(x) = a_{0,i} + a_{1,i}x + a_{2,i}x^2 + a_{3,i}x^3$$

- n intervals, $n+1$ knots, 4 unknowns per interval
- $4n$ unknowns
- $2n$ constraints by continuity

degree 3 spline: cubic spline

In each intervale $[t_i, t_{i+1}]$, $S(x)$ looks like

$$S_i(x) = a_{0,i} + a_{1,i}x + a_{2,i}x^2 + a_{3,i}x^3$$

- n intervals, $n+1$ knots, 4 unknowns per interval
- $4n$ unknowns
- $2n$ constraints by continuity
- $n-1$ constraints by continuity of $S'(x)$

degree 3 spline: cubic spline

In each intervale $[t_i, t_{i+1}]$, $S(x)$ looks like

$$S_i(x) = a_{0,i} + a_{1,i}x + a_{2,i}x^2 + a_{3,i}x^3$$

- n intervals, $n+1$ knots, 4 unknowns per interval
- $4n$ unknowns
- $2n$ constraints by continuity
- $n-1$ constraints by continuity of $S'(x)$
- $n-1$ constraints by continuity of $S''(x)$

degree 3 spline: cubic spline

In each intervale $[t_i, t_{i+1}]$, $S(x)$ looks like

$$S_i(x) = a_{0,i} + a_{1,i}x + a_{2,i}x^2 + a_{3,i}x^3$$

- n intervals, $n+1$ knots, 4 unknowns per interval
- $4n$ unknowns
- $2n$ constraints by continuity
- $n-1$ constraints by continuity of $S'(x)$
- $n-1$ constraints by continuity of $S''(x)$
- $4n-2$ total constraints

degree 3 spline: cubic spline

In each intervale $[t_i, t_{i+1}]$, $S(x)$ looks like

$$S_i(x) = a_{0,i} + a_{1,i}x + a_{2,i}x^2 + a_{3,i}x^3$$

- n intervals, $n+1$ knots, 4 unknowns per interval
- $4n$ unknowns
- $2n$ constraints by continuity
- $n-1$ constraints by continuity of $S'(x)$
- $n-1$ constraints by continuity of $S''(x)$
- $4n-2$ total constraints
- This leaves 2 extra degrees of freedom. The cubic spline is not yet unique!

degree 3 spline: cubic spline

Some options:

- natural cubic spline: $S''(t_0) = S''(t_n) = 0$
- fixed-slope: $S'(t_0) = a$, $S'(t_n) = b$
- not-a-knot: $S'''(x)$ continuous at t_1 and t_{n-1}
- periodic: S' and S'' are periodic at the ends

natural cubic spline

How do we find $a_{0,i}, a_{1,i}, a_{2,i}, a_{3,i}$ for each i? Consider knots $t_0, \ldots, t_n$. Follow our example with the following steps:

1. Assume we knew $S''(t_i)$ for each i
2. $S_i''(x)$ is linear, so construct it
3. Get $S_i(x)$ by integrating $S_i''(x)$ twice
4. Impose continuity
5. Differentiate $S_i(x)$ to impose continuity on $S'(x)$

natural cubic spline: Step 1

Assume we knew $S''(t_i)$ for each i

We know $S''(x)$ is continuous. So assume

$$z_i = S''(t_i)$$

(we don't actually know z_i, not yet at least)

natural cubic spline: Step 2

$S_i''(x)$ is linear, so construct it

Since $S_i''(x)$ is linear, and

$$S_i''(t_i) = z_i$$
$$S_i''(t_{i+1}) = z_{i+1}$$

we can write $S_i''(x)$ as

$$\begin{aligned} S_i''(x) &= z_i \frac{t_{i+1} - x}{t_{i+1} - t_i} + z_{i+1} \frac{x - t_i}{t_{i+1} - t_i} \\ &= \frac{z_i}{h_i}(t_{i+1} - x) + \frac{z_{i+1}}{h_i}(x - t_i) \end{aligned}$$

natural cubic spline: Step 3

Get $S_i(x)$ by integrating $S_i''(x)$ twice

Take

$$S_i''(x) = \frac{z_i}{h_i}(t_{i+1} - x) + \frac{z_{i+1}}{h_i}(x - t_i)$$

and integrate once:

$$S_i'(x) = -\frac{z_i}{2h_i}(t_{i+1} - x)^2 + \frac{z_{i+1}}{2h_i}(x - t_i)^2 + \hat{C}_i$$

twice:

$$S_i(x) = \frac{z_i}{6h_i}(t_{i+1} - x)^3 + \frac{z_{i+1}}{6h_i}(x - t_i)^3 + \hat{C}_i x + \hat{D}_i$$

adjust:

$$S_i(x) = \frac{z_i}{6h_i}(t_{i+1} - x)^3 + \frac{z_{i+1}}{6h_i}(x - t_i)^3 + C_i(x - t_i) + D_i(t_{i+1} - x)$$

natural cubic spline: Step 4

Impose continuity

For each interval $[t_i, t_{i+1}]$, we require $S_i(t_i) = y_i$ and $S_i(t_{i+1}) = y_{i+1}$:

$$y_i = S_i(t_i) = \frac{z_i}{6h_i}(t_{i+1} - t_i)^3 + \frac{z_{i+1}}{6h_i}(t_i - t_i)^3 + C_i(t_i - t_i) + D_i(t_{i+1} - t_i)$$

$$= \frac{z_i}{6}h_i^2 + D_i h_i$$

$$D_i = \frac{y_i}{h_i} - \frac{h_i}{6}z_i$$

and

$$y_{i+1} = S_i(t_{i+1}) = \frac{z_i}{6h_i}(t_{i+1} - t_{i+1})^3 + \frac{z_{i+1}}{6h_i}(t_{i+1} - t_i)^3 + C_i(t_{i+1} - t_i) + D_i(t_{i+1} - t_{i+1})$$

$$= \frac{z_{i+1}}{6}(h_i)^2 + C_i h_i$$

$$C_i = \frac{y_{i+1}}{h_i} - \frac{h_i}{6}z_{i+1}$$

natural cubic spline: Step 4

Impose continuity

So far we have

$$S_i(x) = \frac{z_i}{6h_i}(t_{i+1}-x)^3 + \frac{z_{i+1}}{6h_i}(x-t_i)^3 + \left(\frac{y_{i+1}}{h_i} - \frac{h_i}{6}z_{i+1}\right)(x-t_i) + \left(\frac{y_i}{h_i} - \frac{h_i}{6}z_i\right)(t_{i+1}-x)$$

natural cubic spline: Step 5

Differentiate $S_i(x)$ to impose continuity on $S'(x)$

$$S_i'(x) = -\frac{z_i}{2h_i}(t_{i+1} - x)^2 + \frac{z_{i+1}}{2h_i}(x - t_i)^2 + \frac{y_{i+1}}{h_i} - \frac{h_i}{6} z_{i+1} - \frac{y_i}{h_i} + \frac{h_i}{6} z_i$$

We need $S_i'(t_i) = S_{i-1}'(t_i)$:

$$S_i'(t_i) = -\frac{h_i}{6} z_{i+1} - \frac{h_i}{3} z_i + \underbrace{\frac{1}{h_i}(y_{i+1} - y_i)}_{b_i}$$

$$S_{i-1}'(t_i) = \frac{h_{i-1}}{6} z_{i-1} + \frac{h_{i-1}}{3} z_i + \underbrace{\frac{1}{h_{i-1}}(y_i - y_{i-1})}_{b_{i-1}}$$

Thus z_i is defined by

$$h_{i-1} z_{i-1} + 2(h_i + h_{i-1}) z_i + h_i z_{i+1} = 6(b_i - b_{i-1})$$

natural cubic spline: Step 6

solve

z_i is defined by

$$h_{i-1}z_{i-1} + 2(h_i + h_{i-1})z_i + h_i z_{i+1} = 6(b_i - b_{i-1})$$

- This is $n-1$ equations, $n-1$ unknowns ($z_0 = z_n = 0$ already)
- an $(n-1)\times(n-1)$ tridiagonal system

$$\begin{bmatrix} 1 & & & & & & & \\ h_0 & u_1 & h_1 & & & & & \\ & h_1 & u_2 & h_2 & & & & \\ & & h_2 & u_3 & h_3 & & & \\ & & & \ddots & \ddots & \ddots & & \\ & & & & h_{n-3} & u_{n-2} & h_{n-2} & \\ & & & & & h_{n-2} & u_{n-1} & h_{n-1} \\ & & & & & & & 1 \end{bmatrix} \begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_{n-2} \\ z_{n-1} \\ z_n \end{bmatrix} = \begin{bmatrix} 0 \\ v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{n-2} \\ v_{n-1} \\ 0 \end{bmatrix}$$

$$\begin{aligned} u_i &= 2(h_i + h_{i-1}) \\ v_i &= 6(b_i - b_{i-1}) \end{aligned}$$

example

Find the natural cubic spline for

x	-1	0	1
y	1	2	-1

1. Determine h_i, b_i, u_i, v_i

$$h = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad b = \begin{bmatrix} 1 \\ -3 \end{bmatrix} \quad u = \begin{bmatrix} 4 \end{bmatrix} \quad v = \begin{bmatrix} -24 \end{bmatrix}$$

2. Solve

$$\begin{bmatrix} 1 & & \\ 1 & 4 & 1 \\ & & 1 \end{bmatrix} \begin{bmatrix} z_0 \\ z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 0 \\ -24 \\ 0 \end{bmatrix}$$

3. Result:

$$\begin{bmatrix} z_0 \\ z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 0 \\ -6 \\ 0 \end{bmatrix}$$

example

Find the natural cubic spline for

x	-1	0	1
y	1	2	-1

1. Plug z_i into

$$S_i(x) = \frac{z_i}{6h_i}(t_{i+1}-x)^3 + \frac{z_{i+1}}{6h_i}(x-t_i)^3 + \left(\frac{y_{i+1}}{h_i} - \frac{h_i}{6}z_{i+1}\right)(x-t_i) + \left(\frac{y_i}{h_i} - \frac{h_i}{6}z_i\right)(t_{i+1}-x)$$

$$S(x) = \begin{cases} -(x+1)^3 + 3(x+1) - x & -1 \leqslant x < 0 \\ -(1-x)^3 - x + 3(1-x) & 0 \leqslant x < 1 \end{cases}$$

Algorithm: page 391 NMC6 (page 403, NMC5)

1. Compute for $i = 0, \dots, n-1$

$$h_i = t_{i+1} - t_i \qquad b_i = \frac{1}{h_i}(y_{i+1} - y_i)$$

2. Set u, v:
3. tridiagonal solve to get z
4. substitute into the nested form for $S(x)$ equation 12, page 392 NMC6 (NMC5: equation 10 page 404)

Next time...

- more on the cubic algorithm
- the B-splines/Bezier Curves