Interpolation and Approximation of functions Part 1

Dinesh K. Pai

Some parts in
Textbook Chapter 9

1

Today

- Announcements
  - Quiz 2 results will be available on Monday
- Assignment 2 spotlights
- Interpolation, continued

2

Today:

Generalize

(1) Higher Dimensions: easy

Interpolate each coordinate separately

If $\bar{c}_0 = \begin{pmatrix} c_{0x} \\ c_{0y} \\ c_{0z} \end{pmatrix}$ $\qquad \bar{c}_1 = \begin{pmatrix} c_{1x} \\ c_{1y} \\ c_{1z} \end{pmatrix}$

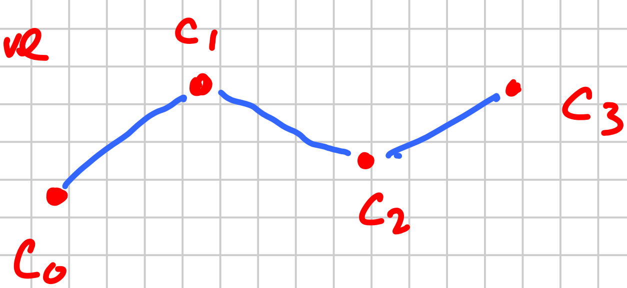
$$c_x = c_{0x}(1-t) + c_{1x}\, t$$

$\vdots$

or write as a vector

$$\bar{c} = \bar{c}_0(1-t) + \bar{c}_1\, t$$

Eg. 3D curve



(2) Higher degree polynomials: quadratic, cubic, ...

quadratic: deg = 2 $\qquad$ cubic: $\boxed{\text{deg} = 3}$ ← sweetspot

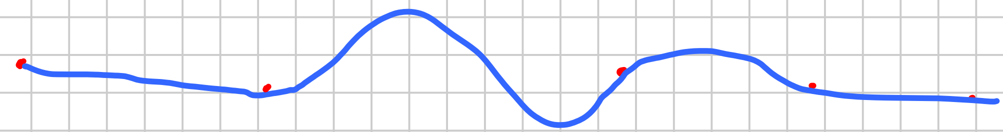
quartic, quintic, ...

deg = 4 $\qquad$ deg = 5

Rarely use deg > 3

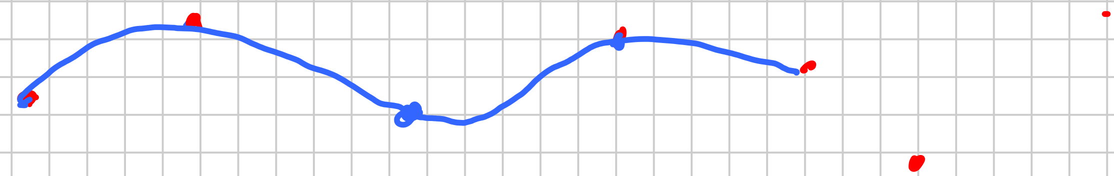
Linear is also widely used

Higher degree polynomials are too wiggly.



Can get "overfitting"

(3) Splines, i.e., piecewise polynomials

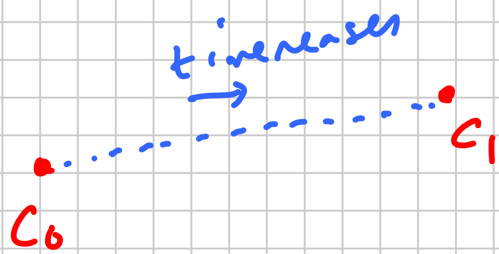


Let's look at linear interpolation again:

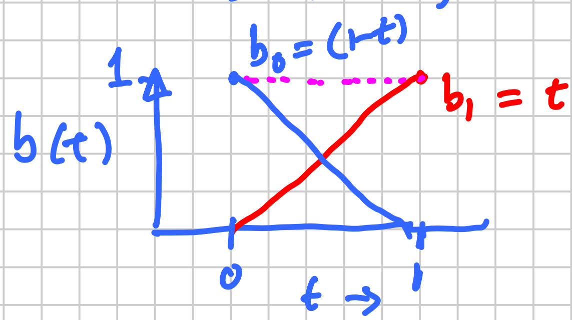
$$c(t) = c_0 \underbrace{(1-t)}_{b_0(t)} + c_1 \underbrace{t}_{b_1(t)} \leftarrow \text{definition}$$

$$= \sum_{i=0}^{1} c_i \, b_i(t)$$

This is really just a weighted average of points with weights parameterized by t



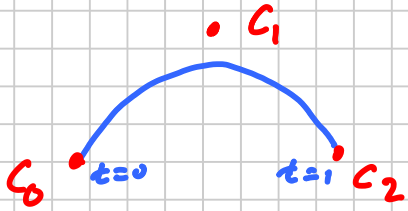
b_i are called blending weights or blending functions



$b_0 + b_1 = (1-t) + t = 1$

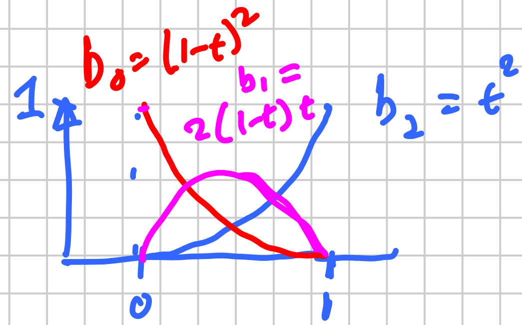
So b_i form a "partition of unity"

How can we get higher degree interpolation, with the same properties?



deg = 2

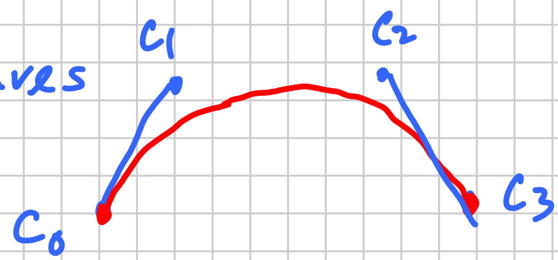
Want $C(t) = C_0\, b_0(t)$
$+ C_1\, b_1(t)$
$+ C_2\, b_2(t)$



It turns out these correspond Bernstein polynomials

	b_0	b_1	b_2	b_3
deg 0	1			
deg 1	$(1-t)$	t		
deg 2	$(1-t)^2$	$2(1-t)t$	t^2	
deg 3	$(1-t)^3$	$3(1-t)^2t$	$3(1-t)t^2$	t^3

§ Bezier Curves

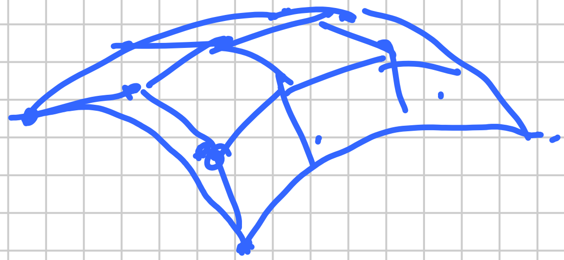


Adobe
eg. Illustrator's
Pen tool

Curves specified by control points

– Catmull-Rom, similar, used in animation

- Can be generalized to produce smooth surfaces



"Bezier patch"

other types "NURBS"

CPSC 424 for more details