BÉZIER CURVES

Rodrigo Silveira

Curve and Surface Design
Facultat d'Informàtica de Barcelona
Universitat Politècnica de Catalunya

Interpolation or... approximation!

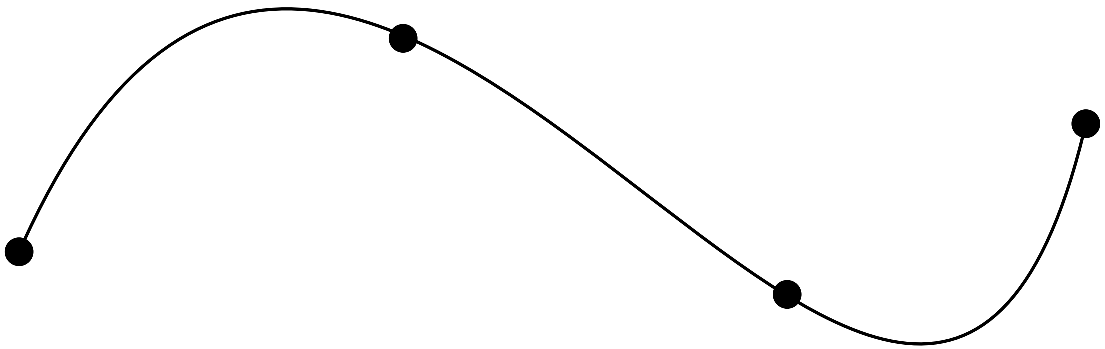
Previous curve design methods based on **interpolation**

Interpolation or... approximation!

Previous curve design methods based on **interpolation**

Interpolation or... approximation!

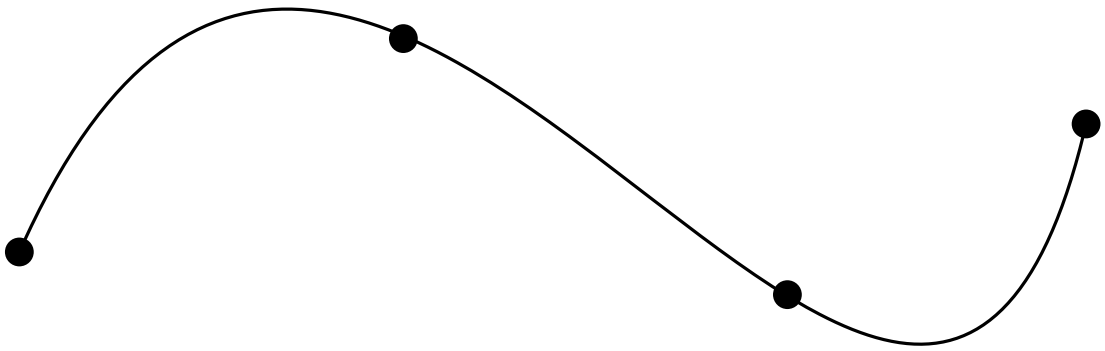
Previous curve design methods based on **interpolation**



Interpolating curve

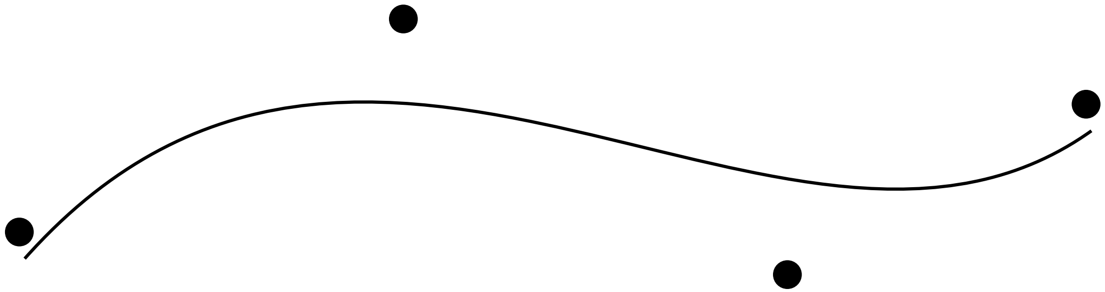
Interpolation or... approximation!

Previous curve design methods based on **interpolation**



Interpolating curve

Curve passes exactly through given points

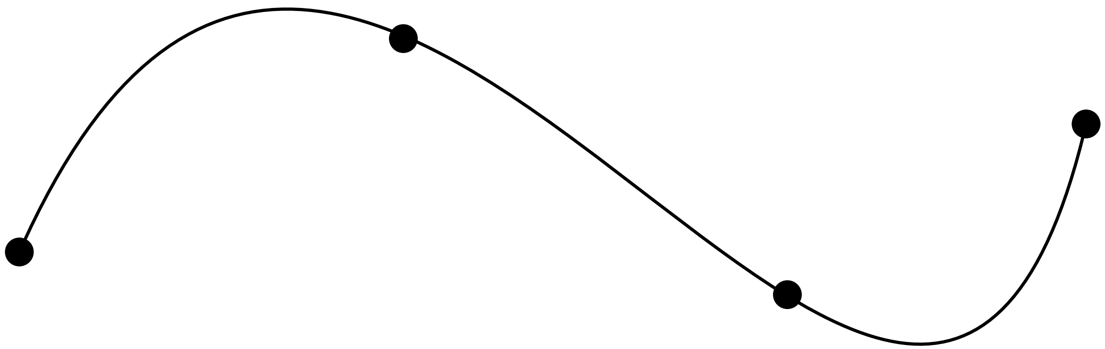


Approximating curve

Curve passes near the given points

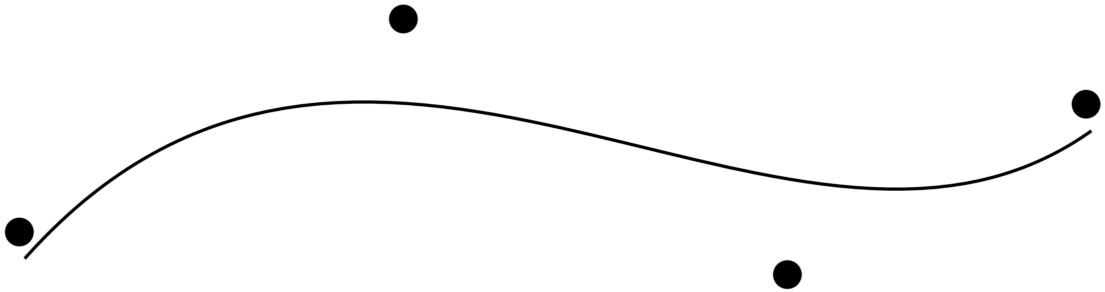
Interpolation or... approximation!

Previous curve design methods based on **interpolation**



Interpolating curve

Curve passes exactly through given points



Approximating curve

Curve passes near the given points

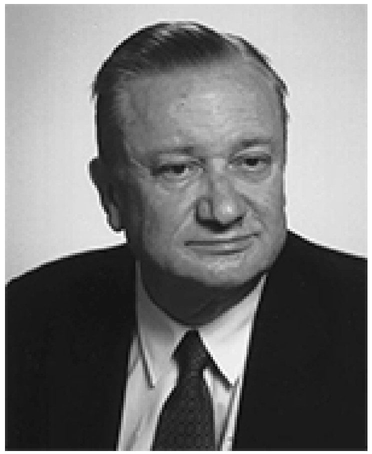
What's wrong with interpolation? Curve change when moving points can be unpredictable

Approximating curves can provide better "shape control"

Bézier curves

Named after Pierre Bézier (1910-1999)

- Worked on automatizing the process of designing cars
- Paul de Casteljau (1930-2022) developed similar methods for Citröen, but were published later



INTRODUCTION TO BÉZIER CURVES

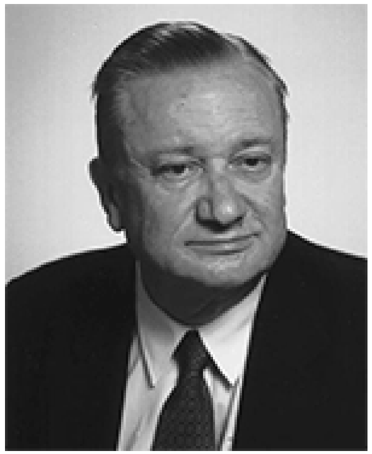
Bézier curves

Named after Pierre Bézier (1910-1999)

- Worked on automatizing the process of designing cars
- Paul de Casteljau (1930-2022) developed similar methods for Citröen, but were published later

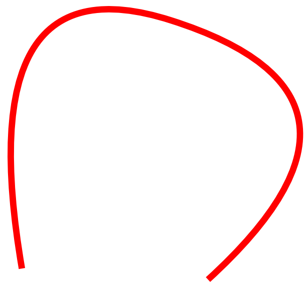
Bézier curve

- Parametric $(P(t))$
- Polynomial
- Based on *control points*

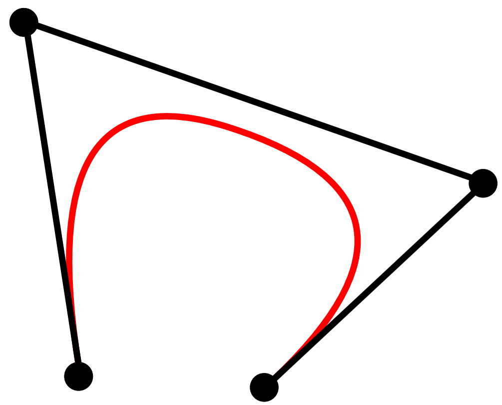


Some examples of Bézier curves

Some examples of Bézier curves

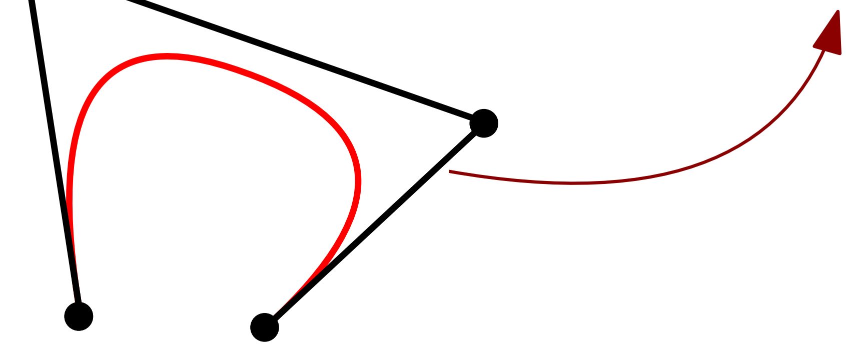


Some examples of Bézier curves



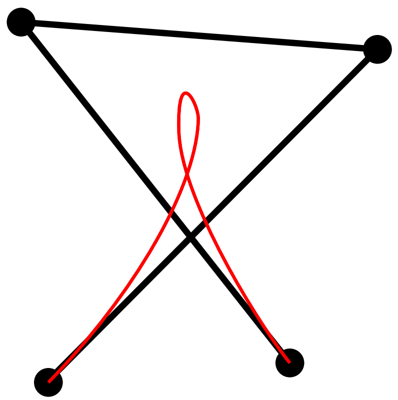
Some examples of Bézier curves

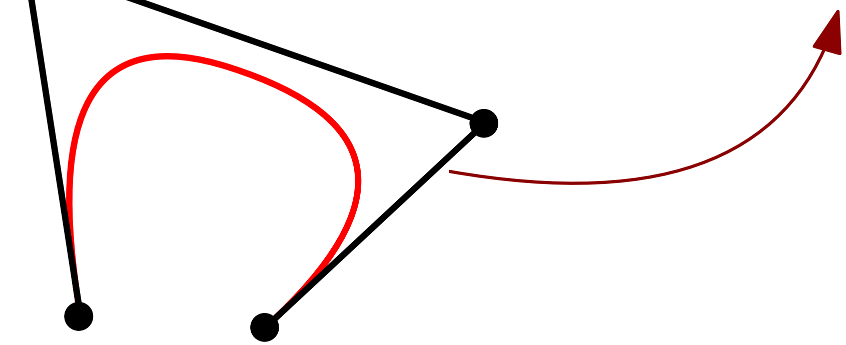
Control polygonal line, made of control points.
Each control point exerts a pull on the curve, pulling it towards itself



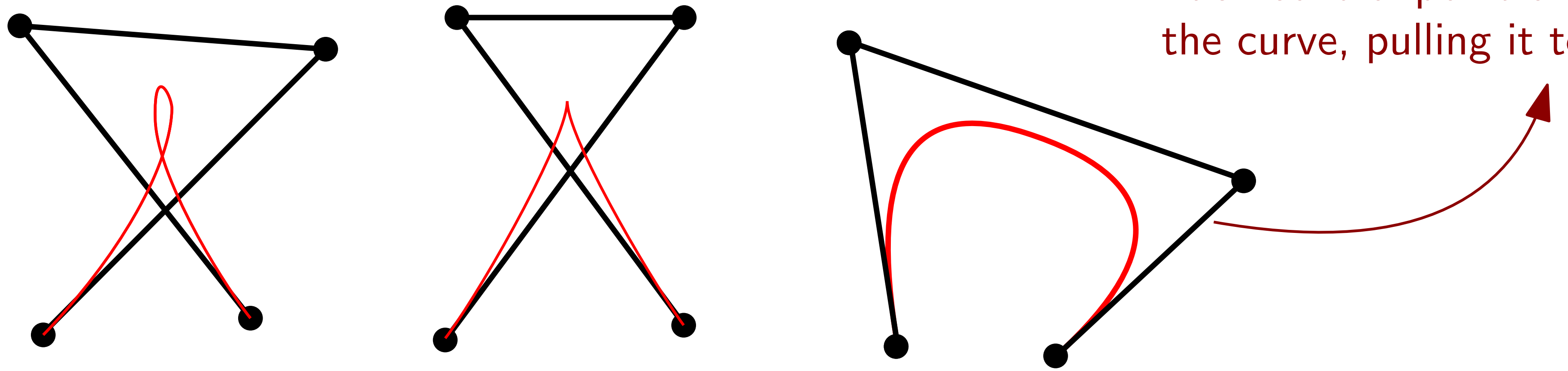
Some examples of Bézier curves

Control polygonal line, made of control points.
Each control point exerts a pull on the curve, pulling it towards itself



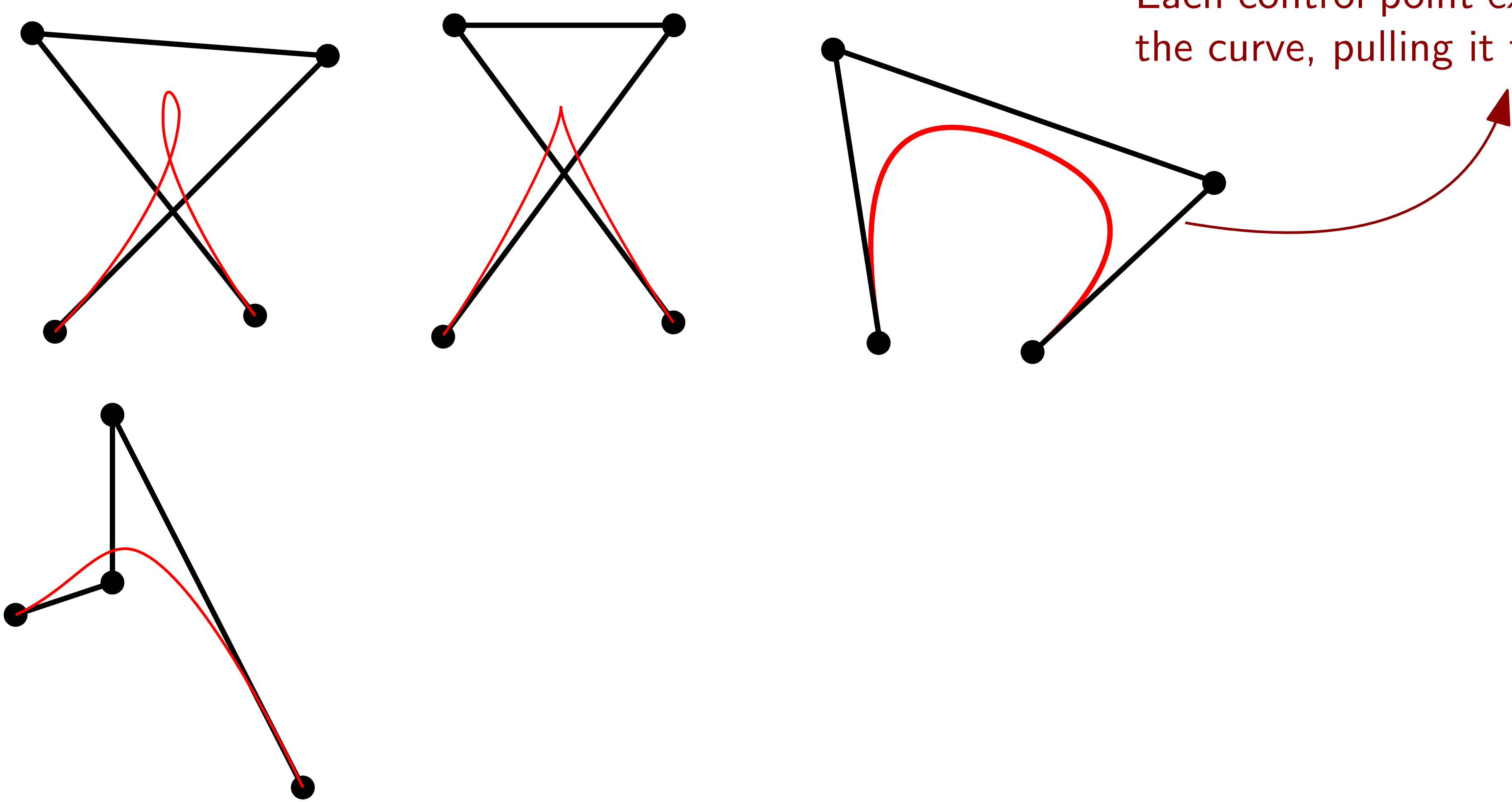


Some examples of Bézier curves



Control polygonal line, made of control points.
Each control point exerts a pull on the curve, pulling it towards itself

Some examples of Bézier curves

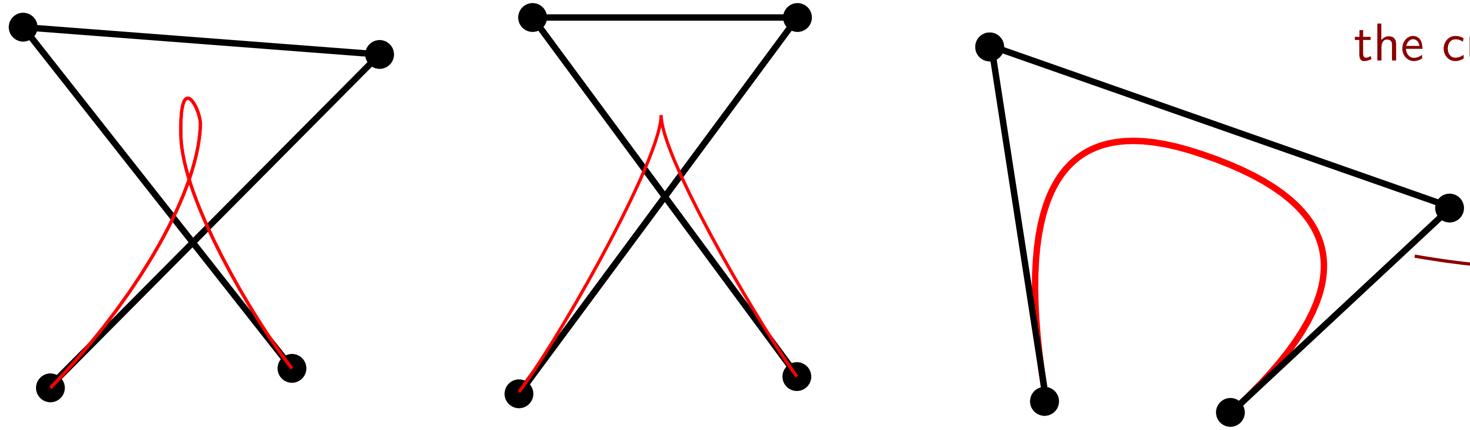


Control polygonal line, made of control points.

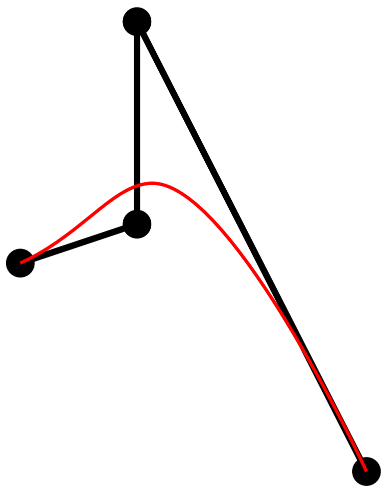
Each control point exerts a pull on the curve, pulling it towards itself

INTRODUCTION TO BÉZIER CURVES

Some examples of Bézier curves



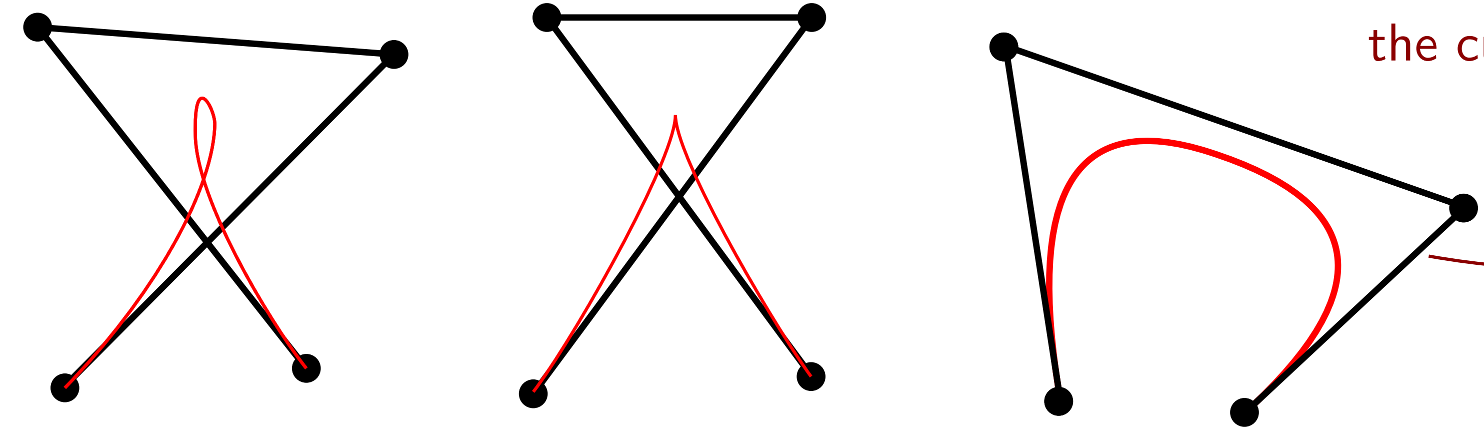
Control polygonal line, made of control points.
Each control point exerts a pull on the curve, pulling it towards itself

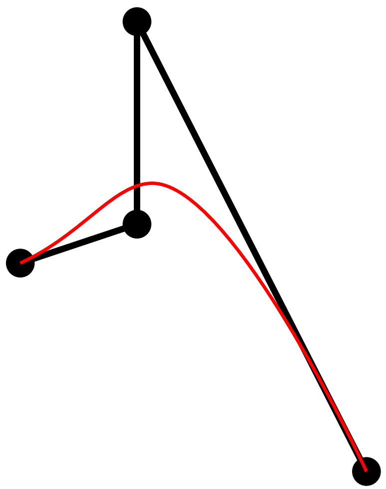


Each curve here is a polynomial, of degree....

Some examples of Bézier curves

Control polygonal line, made of control points.
Each control point exerts a pull on the curve, pulling it towards itself

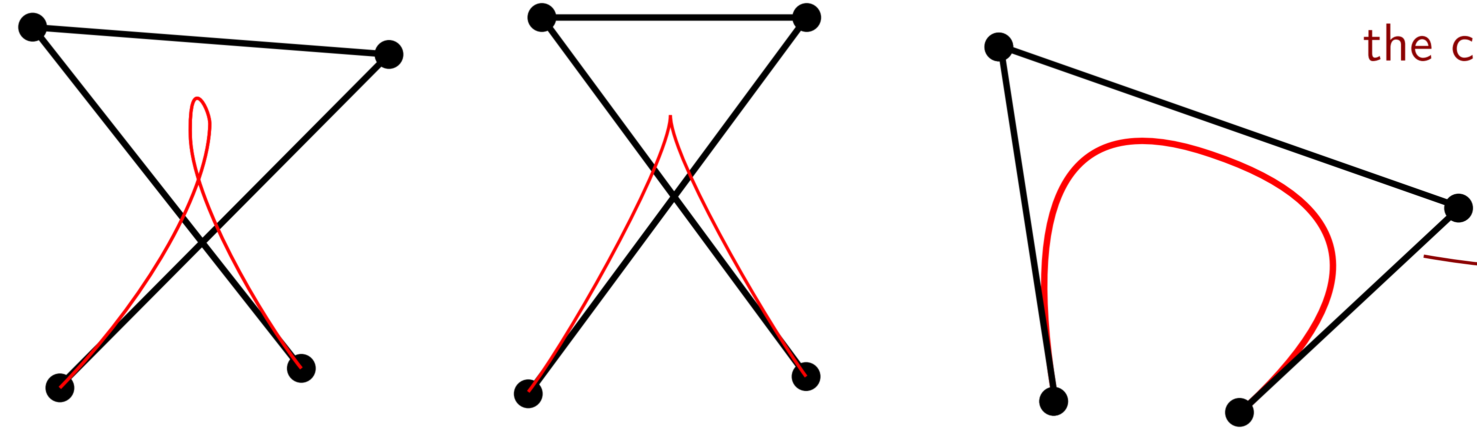


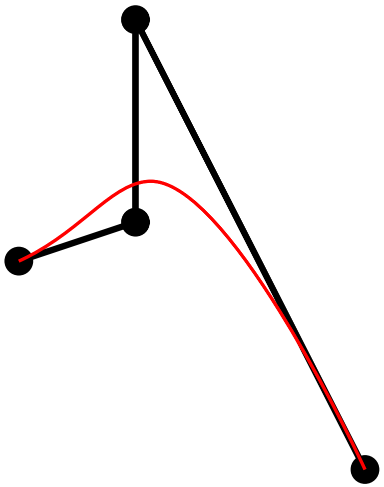


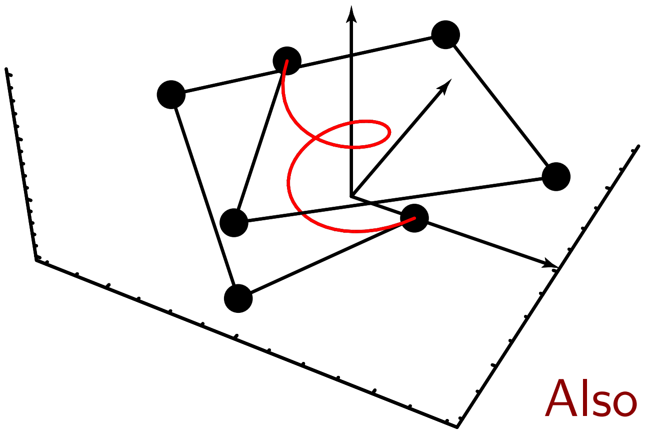
Each curve here is a polynomial, of degree.... 3

Some examples of Bézier curves

Control polygonal line, made of control points.
Each control point exerts a pull on the curve, pulling it towards itself







Also works in 3D!

What is a Bézier curve?

General form

$$P(t) = \sum_{i=0}^{n} P_i f_i(t) \qquad t \in [0, 1]$$

What is a Bézier curve?

General form

$$P(t) = \sum_{i=0}^{n} P_i f_i(t) \qquad t \in [0, 1]$$

control point ← P_i ; f_i → **basis function**: gives weight of each point as function of t

What is a Bézier curve?

General form

$$P(t) = \sum_{i=0}^{n} P_i f_i(t) \qquad t \in [0,1]$$

control point ← P_i ; f_i → **basis function**: gives weight of each point as function of t

Bézier looked for basis functions that gave the following properties:

What is a Bézier curve?

General form

$$P(t) = \sum_{i=0}^{n} P_i f_i(t) \qquad t \in [0, 1]$$

control point ← P_i ; f_i → **basis function**: gives weight of each point as function of t

Bézier looked for basis functions that gave the following properties:

- Interpolates the first and last point (to have control on first and last point)

What is a Bézier curve?

General form

$$P(t) = \sum_{i=0}^{n} P_i f_i(t) \qquad t \in [0, 1]$$

control point ← P_i ; f_i → **basis function**: gives weight of each point as function of t

Bézier looked for basis functions that gave the following properties:

- Interpolates the first and last point (to have control on first and last point)
- Tangent vectors: at P_0 must be $P_1 - P_0$, at P_n must be $P_n - P_{n-1}$ (to have control of directions at the beginning and end)

What is a Bézier curve?

General form

$$P(t) = \sum_{i=0}^{n} P_i f_i(t) \qquad t \in [0, 1]$$

control point ← P_i ; f_i → **basis function**: gives weight of each point as function of t

Bézier looked for basis functions that gave the following properties:

- Interpolates the first and last point (to have control on first and last point)
- Tangent vectors: at P_0 must be $P_1 - P_0$, at P_n must be $P_n - P_{n-1}$ (to have control of directions at the beginning and end)
- Similar to 2), for higher order derivatives: $P^{(k)}(0)$ should depend on $P_0, \ldots, P_k$ only (e.g., $P''(0)$ should depend only on P_0, P_1, and P_2)

What is a Bézier curve?

General form

$$P(t) = \sum_{i=0}^{n} P_i f_i(t) \qquad t \in [0,1]$$

control point ← P_i ; f_i → **basis function**: gives weight of each point as function of t

Bézier looked for basis functions that gave the following properties:

- Interpolates the first and last point (to have control on first and last point)
- Tangent vectors: at P_0 must be $P_1 - P_0$, at P_n must be $P_n - P_{n-1}$ (to have control of directions at the beginning and end)
- Similar to 2), for higher order derivatives: $P^{(k)}(0)$ should depend on $P_0, \ldots, P_k$ only (e.g., $P''(0)$ should depend only on P_0, P_1, and P_2)
- The basis functions must be symmetric with respect to t and $(1-t)$ (so reversing the parameter and the order of control points gives the same curve)

What is a Bézier curve?

General form

$$P(t) = \sum_{i=0}^{n} P_i f_i(t) \qquad t \in [0, 1]$$

control point ← P_i ; f_i → **basis function**: gives weight of each point as function of t

Bézier looked for basis functions that gave the following properties:

- Interpolates the first and last point (to have control on first and last point)
- Tangent vectors: at P_0 must be $P_1 - P_0$, at P_n must be $P_n - P_{n-1}$ (to have control of directions at the beginning and end)
- Similar to 2), for higher order derivatives: $P^{(k)}(0)$ should depend on $P_0, \ldots, P_k$ only (e.g., $P''(0)$ should depend only on P_0, P_1, and P_2)
- The basis functions must be symmetric with respect to t and $(1-t)$ (so reversing the parameter and the order of control points gives the same curve)
- Control point weights are barycentric (=affine sum): shape independent from coordinate system. That is: $P(t)$ is an affine combination of control points, so curve is affine invariant

Basis functions

The family a functions used are **Bernstein polynomials**

$$f_i(t) = B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

recall that
$0 \le i \le n$,
$\binom{n}{i} = \frac{n!}{i!(n-i)!}$, $0! = 1$
and assume $0^0 = 1$

Basis functions

The family a functions used are **Bernstein polynomials**

$$f_i(t) = B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

recall that
$0 \leq i \leq n$,
$\binom{n}{i} = \frac{n!}{i!(n-i)!}$, $0! = 1$
and assume $0^0 = 1$

note the n: the basis depends on the number of control points

Basis functions

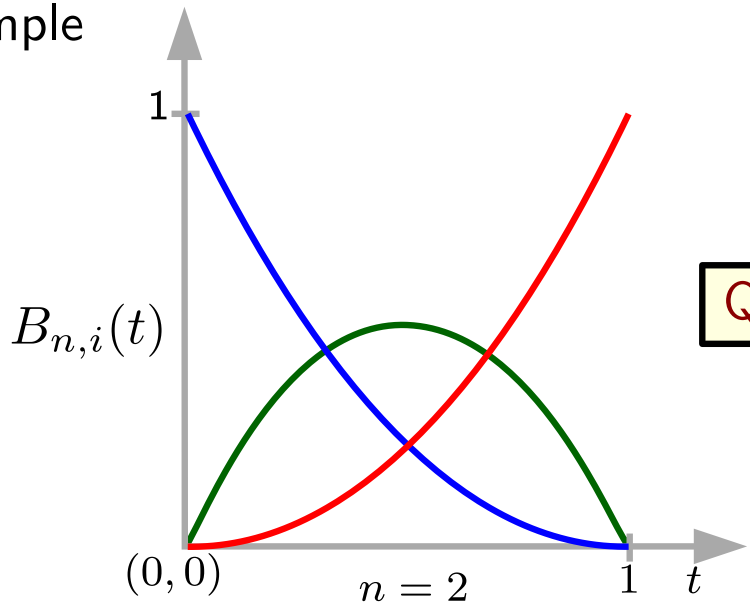
The family a functions used are **Bernstein polynomials**

$$f_i(t) = B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

note the n: the basis depends on the number of control points

recall that
$0 \leq i \leq n$,
$\binom{n}{i} = \frac{n!}{i!(n-i)!}$, $0! = 1$
and assume $0^0 = 1$

Example



Question: Which basis function is which?

Basis functions

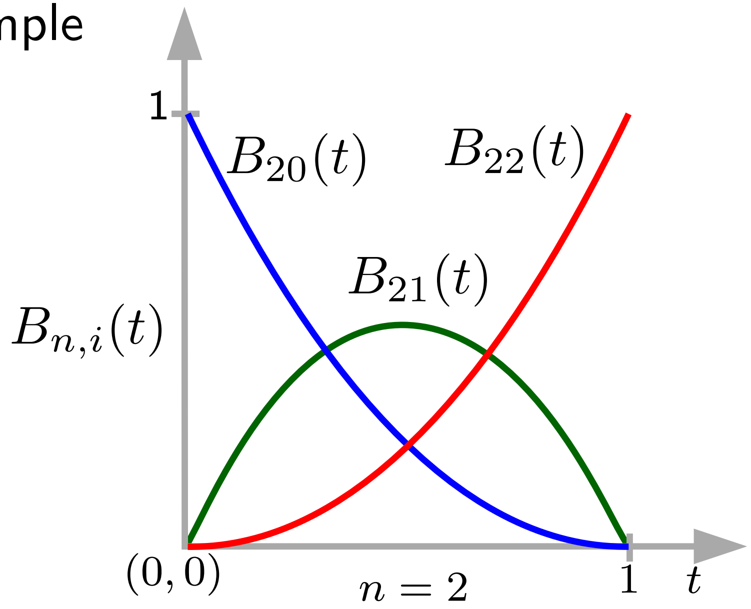
The family a functions used are **Bernstein polynomials**

$$f_i(t) = B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

recall that
$0 \le i \le n$,
$\binom{n}{i} = \frac{n!}{i!(n-i)!}$, $0! = 1$
and assume $0^0 = 1$

note the n: the basis depends on the number of control points

Example



Basis functions

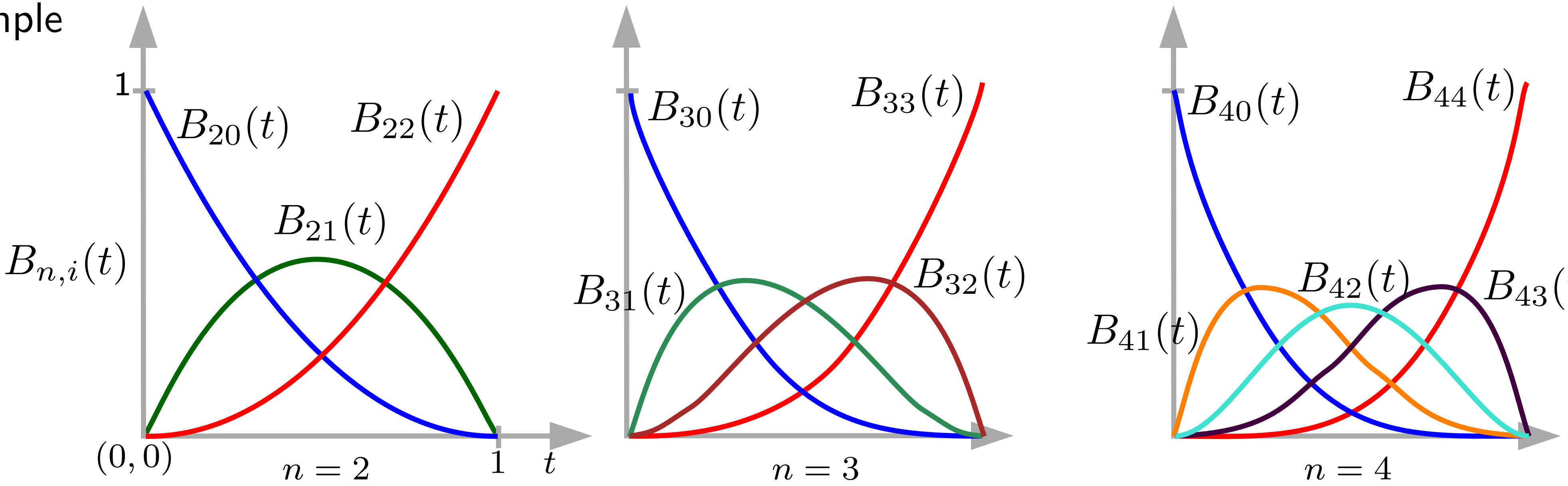
The family a functions used are **Bernstein polynomials**

$$f_i(t) = B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

recall that
$0 \leq i \leq n$,
$\binom{n}{i} = \frac{n!}{i!(n-i)!}$, $0! = 1$
and assume $0^0 = 1$

note the n: the basis depends on the number of control points

Example



Basis functions

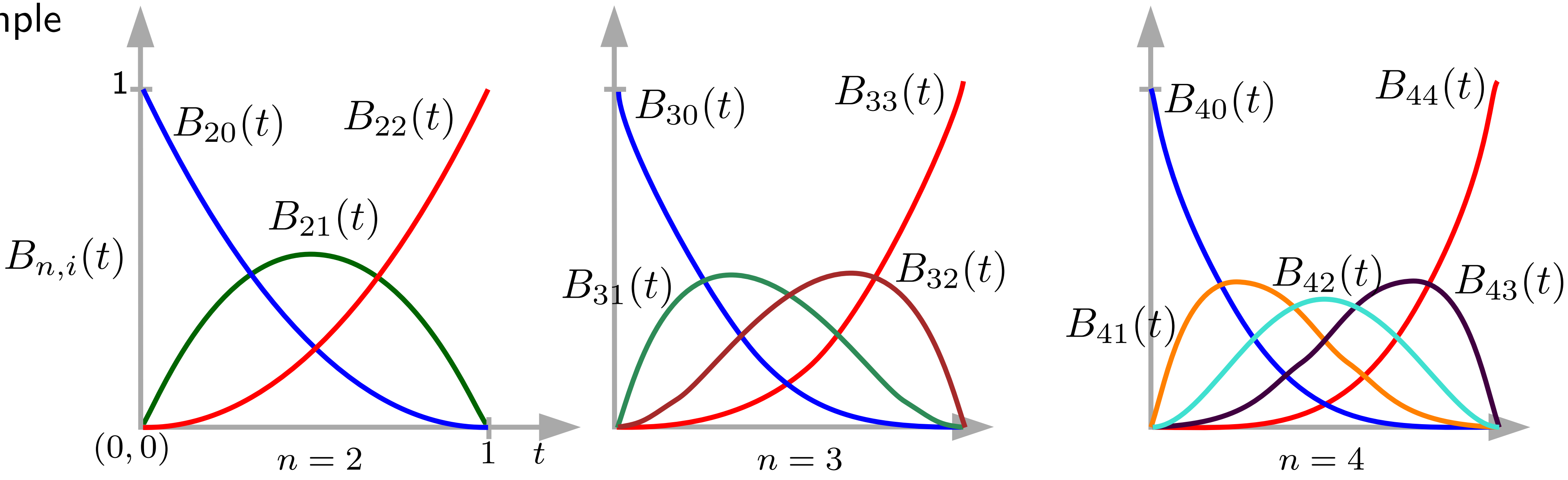
The family a functions used are **Bernstein polynomials**

$$f_i(t) = B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

recall that
$0 \le i \le n$,
$\binom{n}{i} = \frac{n!}{i!(n-i)!}$, $0! = 1$
and assume $0^0 = 1$

note the n: the basis depends on the number of control points

Example



The Bézier curve becomes $P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t) \qquad t \in [0,1]$

Example: degree-2 Bézier curve

$$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

Example: degree-2 Bézier curve

$$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

For $n = 2$, we have $B_{n,i}(t) = \binom{2}{i} t^i (1-t)^{2-i}$, for $0 \leq i \leq 2$

So, for $n = 2$, these are the three Bernstein polynomials:

- $B_{2,0}(t) = \binom{2}{0} t^0 (1-t)^{2-0} = (1-t)^2$
- $B_{2,1}(t) = \binom{2}{1} t^1 (1-t)^{2-1} = 2t(1-t)$
- $B_{2,2}(t) = \binom{2}{2} t^2 (1-t)^{2-2} = t^2$

Example: degree-2 Bézier curve

$$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

For $n = 2$, we have $B_{n,i}(t) = \binom{2}{i} t^i (1-t)^{2-i}$, for $0 \leq i \leq 2$

So, for $n = 2$, these are the three Bernstein polynomials:

- $B_{2,0}(t) = \binom{2}{0} t^0 (1-t)^{2-0} = (1-t)^2$
- $B_{2,1}(t) = \binom{2}{1} t^1 (1-t)^{2-1} = 2t(1-t)$
- $B_{2,2}(t) = \binom{2}{2} t^2 (1-t)^{2-2} = t^2$

So the quadratic Bézier curve is

$$P(t) = (1-t)^2 P_0 + 2t(1-t) P_1 + t^2 P_2$$

Example?

Example: degree-2 Bézier curve

$$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

For $n = 2$, we have $B_{n,i}(t) = \binom{2}{i} t^i (1-t)^{2-i}$, for $0 \leq i \leq 2$

So, for $n = 2$, these are the three Bernstein polynomials:

- $B_{2,0}(t) = \binom{2}{0} t^0 (1-t)^{2-0} = (1-t)^2$
- $B_{2,1}(t) = \binom{2}{1} t^1 (1-t)^{2-1} = 2t(1-t)$
- $B_{2,2}(t) = \binom{2}{2} t^2 (1-t)^{2-2} = t^2$

So the quadratic Bézier curve is

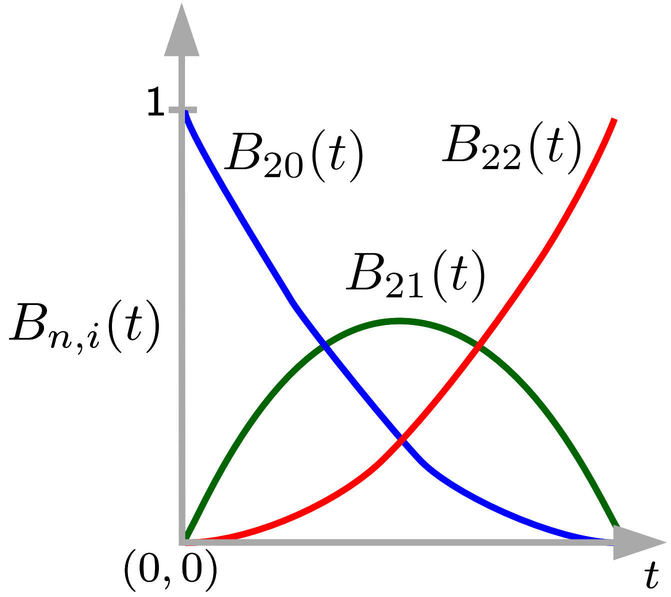
$$P(t) = (1-t)^2 P_0 + 2t(1-t) P_1 + t^2 P_2$$

Example?

Question: Does this curve satisfy the properties in the previous slide?

Properties of Bézier curves

1. Endpoint interpolation
2. Symmetry
3. Affine invariance
4. Invariance under affine parameter transformations
5. Convex hull property
6. Pseudolocal control
7. Variation-diminishing property



$$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

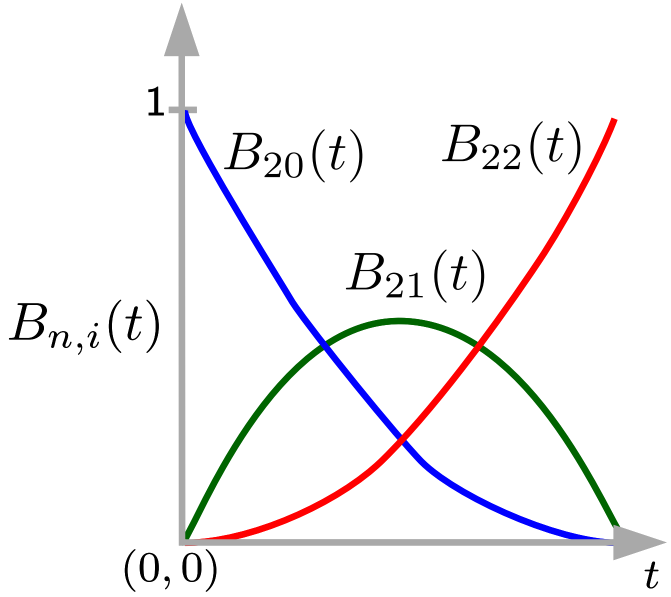
$$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$$

recall that
$0 \le i \le n$,
$\binom{n}{i} = \frac{n!}{i!(n-i)!}$, $0! = 1$
and assume $0^0 = 1$

BÉZIER CURVES

Properties of Bézier curves

1. Endpoint interpolation ✓
2. Symmetry ✓
3. Affine invariance
4. Invariance under affine parameter transformations
5. Convex hull property
6. Pseudolocal control
7. Variation-diminishing property



$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$

$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$

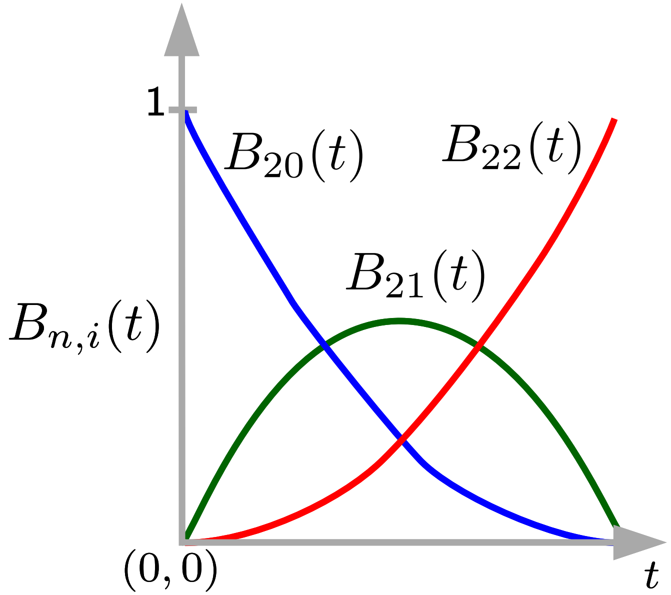
recall that
$0 \leq i \leq n$,
$\binom{n}{i} = \frac{n!}{i!(n-i)!}$, $0! = 1$
and assume $0^0 = 1$

Properties of Bézier curves

3. Affine invariance

Applying an affine transformation to the curve is the same as applying the transformation to the control points

More precisely: $f(P(t)) = \sum_{i=0}^{n} f(P_i) B_{n,i}(t)$, for any affine map f, i.e., $f(v) = Av + W$



$$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

$$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$$

Properties of Bézier curves

3. Affine invariance

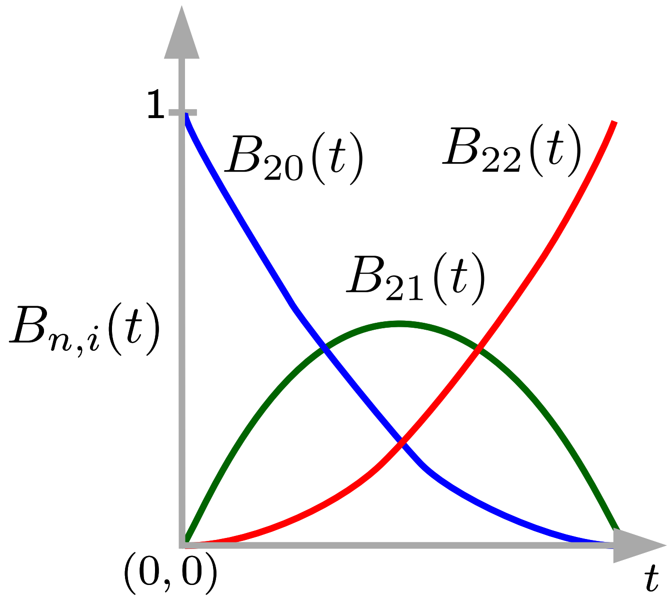
Applying an affine transformation to the curve is the same as applying the transformation to the control points

More precisely: $f(P(t)) = \sum_{i=0}^{n} f(P_i) B_{n,i}(t)$, for any affine map f, i.e., $f(v) = Av + W$

We can prove this by showing that $\sum_{i=0}^{n} B_{n,i}(t) = 1$

This follows from the binomial theorem:
$(a+b)^n = \sum_{i=0}^{n} \binom{n}{i} a^i b^{n-i}$, with $a = t$ and $b = (1-t)$

Therefore Bézier curves are invariant under affine transformations!



$$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

$$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$$

Properties of Bézier curves

3. Affine invariance

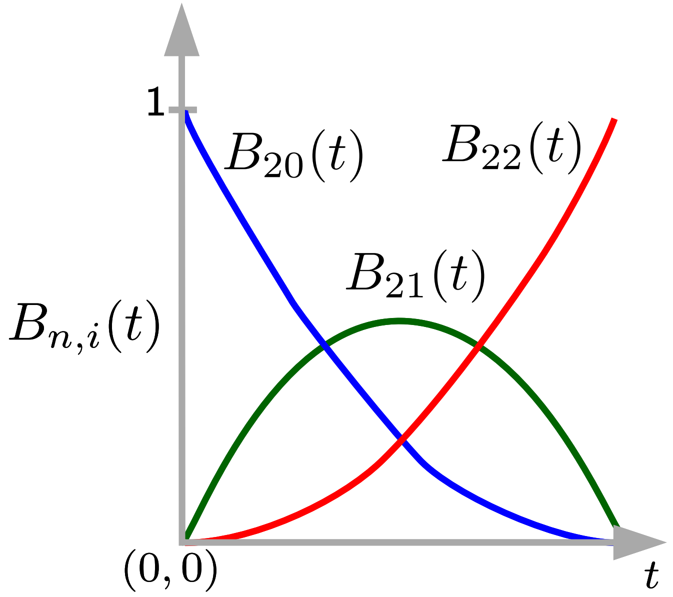
Applying an affine transformation to the curve is the same as applying the transformation to the control points

More precisely: $f(P(t)) = \sum_{i=0}^{n} f(P_i)B_{n,i}(t)$, for any affine map f, i.e., $f(v) = Av + W$

We can prove this by showing that $\sum_{i=0}^{n} B_{n,i}(t) = 1$

This follows from the binomial theorem:
$(a+b)^n = \sum_{i=0}^{n} \binom{n}{i} a^i b^{n-i}$, with $a = t$ and $b = (1-t)$

Therefore Bézier curves are invariant under affine transformations!



$$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

$$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$$

4. Invariance under affine parameter transformations

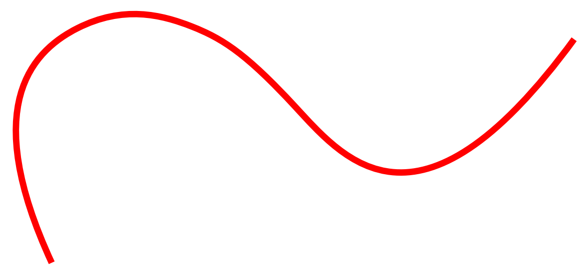
If $C(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$ for $t \in [0,1]$, and $D(t) = \sum_{i=0}^{n} P_i B_{n,i}(\frac{u-a}{b-a})$ for $u \in [a,b]$, then curves C and D are equal

Practical consecuence: it is easy to have a curve defined over $[a,b]$ instead of $[0,1]$

Properties of Bézier curves

5. Convex hull property

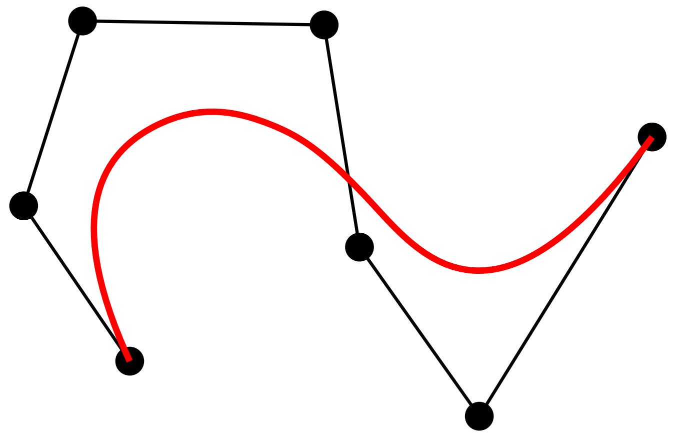
The curve lies inside the convex hull of the control points



Properties of Bézier curves

5. Convex hull property

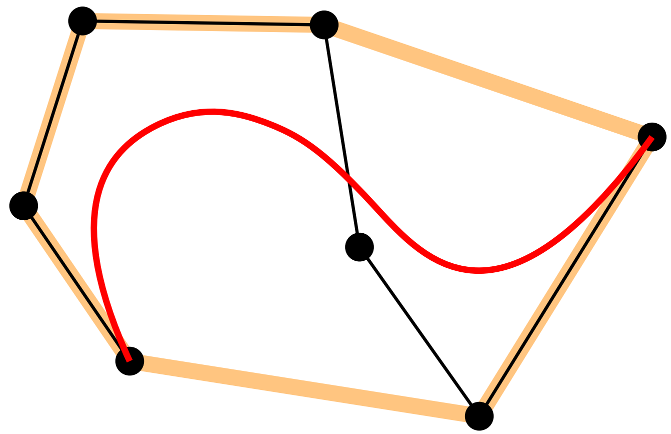
The curve lies inside the convex hull of the control points



Properties of Bézier curves

5. Convex hull property

The curve lies inside the convex hull of the control points

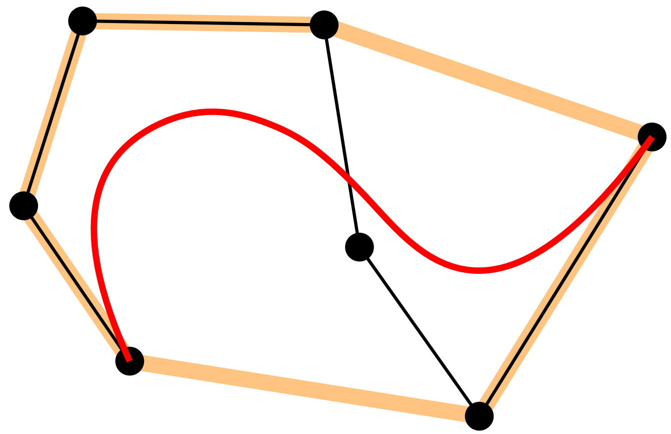


Properties of Bézier curves

5. Convex hull property

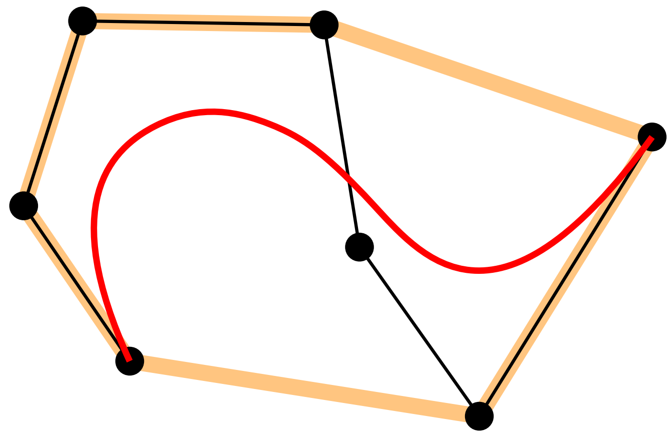
The curve lies inside the convex hull of the control points

Why is this important?



Properties of Bézier curves

5. Convex hull property

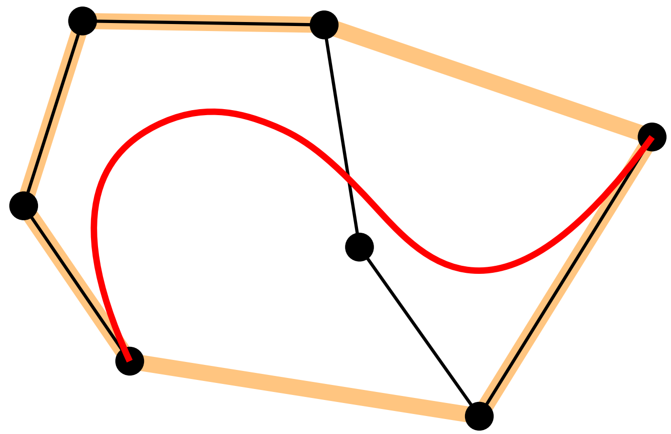


The curve lies inside the convex hull of the control points

Why is this important?

- ensures some control (remember Runge's phenomenon)
- helps in checking if two curves intersect (**Question**: how?)

Properties of Bézier curves



5. Convex hull property

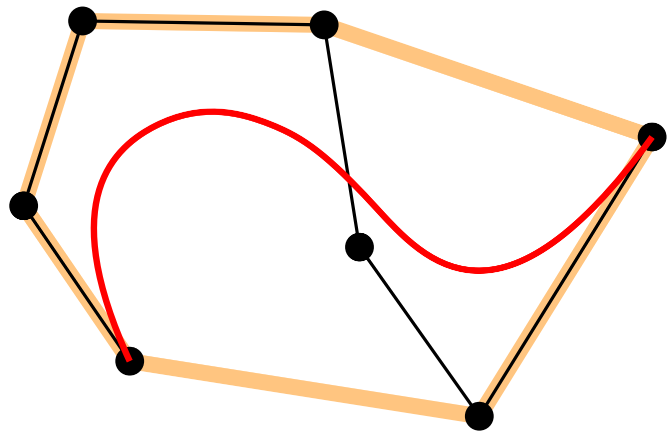
The curve lies inside the convex hull of the control points

Why is this important?

- ensures some control (remember Runge's phenomenon)
- helps in checking if two curves intersect (**Question**: how?)

Why is this property true?

Properties of Bézier curves



5. Convex hull property

The curve lies inside the convex hull of the control points

Why is this important?

- ensures some control (remember Runge's phenomenon)
- helps in checking if two curves intersect (**Question**: how?)

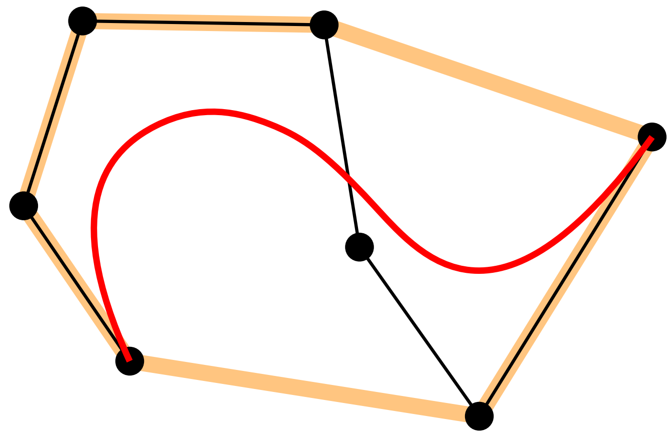
Why is this property true?

$P(t) = \sum_{i=0}^{n} B_{n,i} P_i(t)$ where $\sum_{i=0}^{n} B_{n,i}(t) = 1$ and $B_{n,i}(t) \geq 0 \ \forall n, i$

Point weights add up to one and are in $[0, 1]$, thus $P(t)$ is a **convex combination** of the control points.

The convex hull of the a set of points S is **exactly** the set of all convex combinations of points in S, thus all points in the curve belong to the convex hull.

Properties of Bézier curves



5. Convex hull property

The curve lies inside the convex hull of the control points

Why is this important?

- ensures some control (remember Runge's phenomenon)
- helps in checking if two curves intersect (**Question**: how?)

Why is this property true?

$P(t) = \sum_{i=0}^{n} B_{n,i} P_i(t)$ where $\sum_{i=0}^{n} B_{n,i}(t) = 1$ and $B_{n,i}(t) \geq 0 \ \forall n, i$

Point weights add up to one and are in $[0, 1]$, thus $P(t)$ is a **convex combination** of the control points.

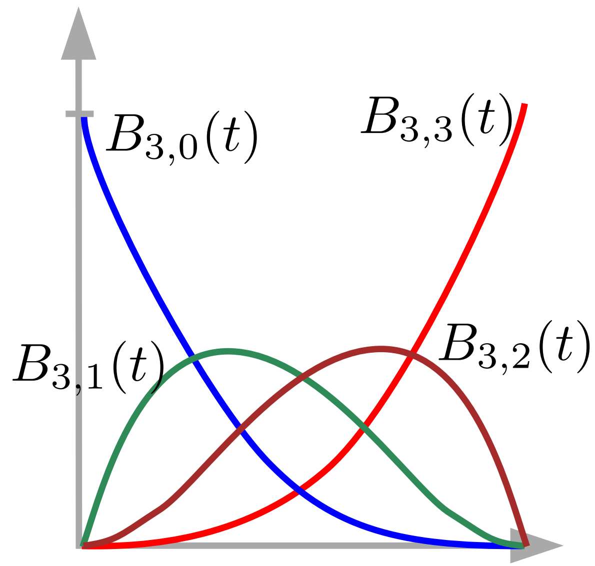
The convex hull of the a set of points S is **exactly** the set of all convex combinations of points in S, thus all points in t

Question: What does this say about collinear control points?

Properties of Bézier curves

6. “Pseudolocal” control

Question: When does a control point influence the curve most?

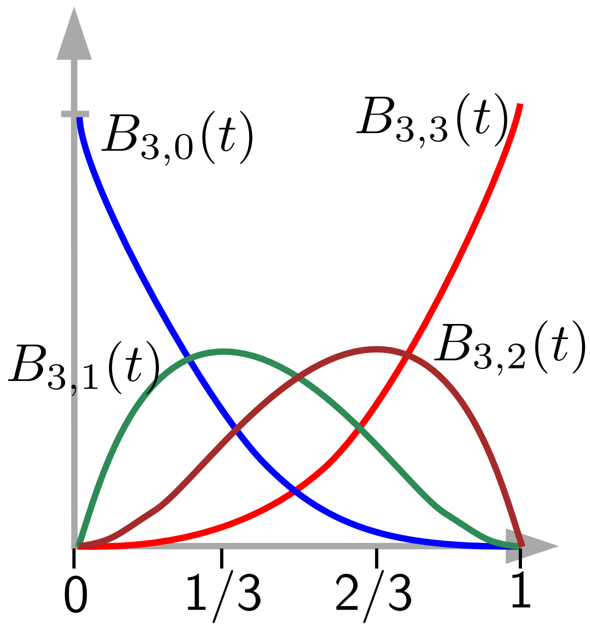


Properties of Bézier curves

6. "Pseudolocal" control

Question: When does a control point influence the curve most?

The ith Bernstein polynomial has only one maximum, at $t = i/n$.



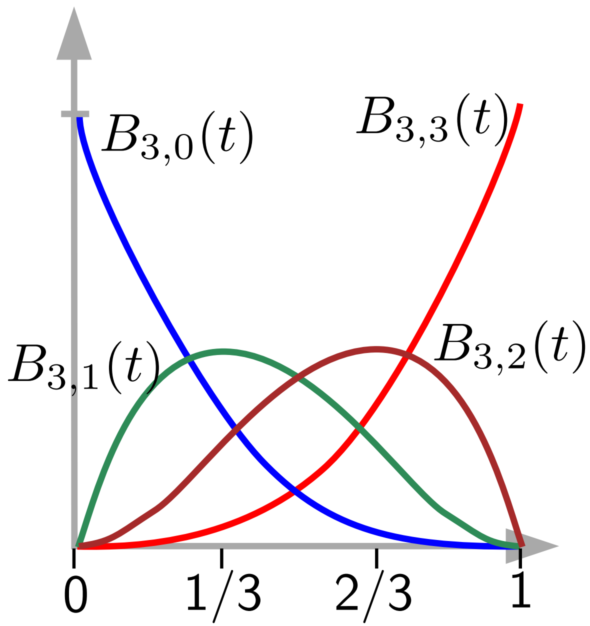
local maxima of $B_{n,i}$s

Properties of Bézier curves

6. "Pseudolocal" control

Question: When does a control point influence the curve most?

The ith Bernstein polynomial has only one maximum, at $t = i/n$.



local maxima of $B_{n,i}$s

Consequence: if we move only one control point, P_i, the curve is mostly affected around $t = i/n$. This makes the effect of the change more or less predictable.

However, note that the change still affects the whole curve (so it is **global control**—not local).

Properties of Bézier curves

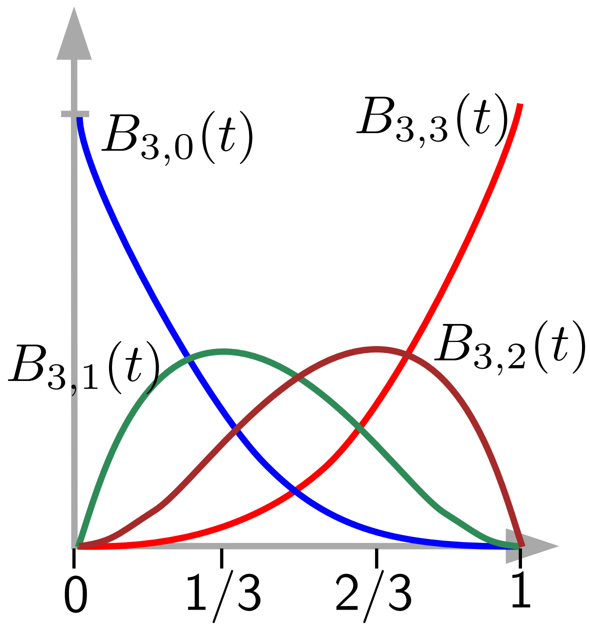
6. "Pseudolocal" control

Question: When does a control point influence the curve most?

The ith Bernstein polynomial has only one maximum, at $t = i/n$.

Consequence: if we move only one control point, P_i, the curve is mostly affected around $t = i/n$. This makes the effect of the change more or less predictable.

However, note that the change still affects the whole curve (so it is **global control**—not local).



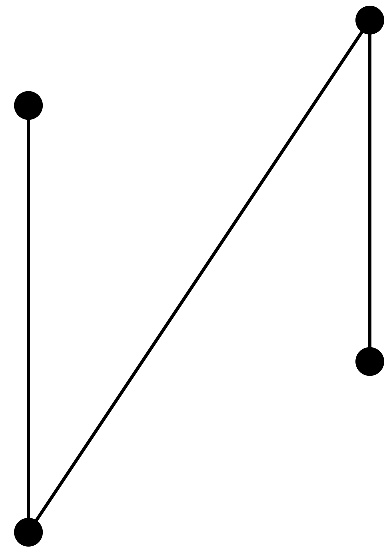
local maxima of $B_{n,i}$s

Question: What happens to $P(t)$ if P_k is moved by a vector (α, β)?

Properties of Bézier curves

7. Variation-diminishing property

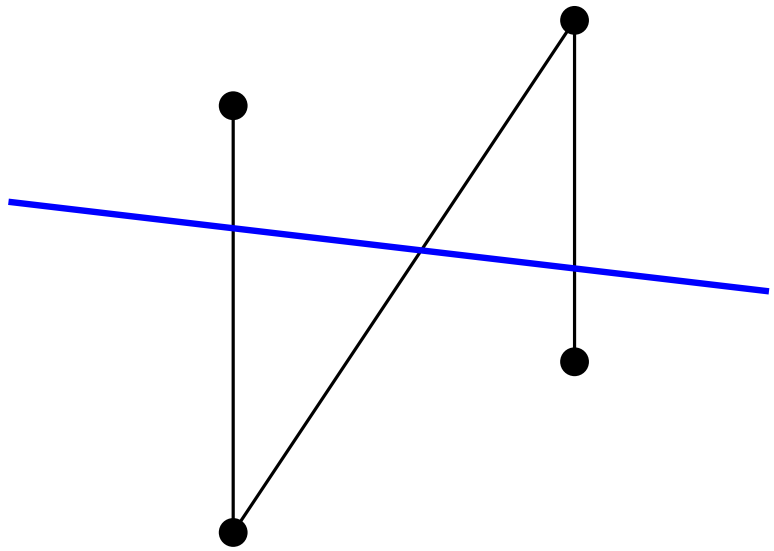
The number of intersections of any line with a Bézier curve is at most the number of intersections of the line with the control polygonal line



Properties of Bézier curves

7. Variation-diminishing property

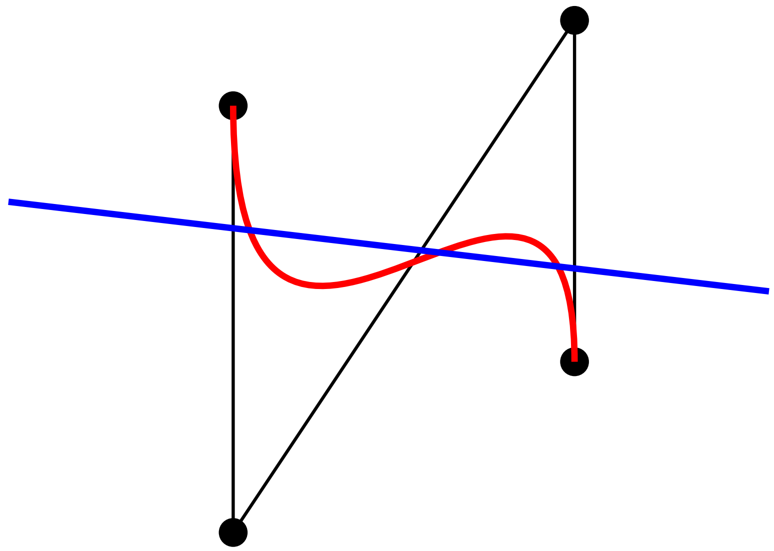
The number of intersections of any line with a Bézier curve is at most the number of intersections of the line with the control polygonal line



Properties of Bézier curves

7. Variation-diminishing property

The number of intersections of any line with a Bézier curve is at most the number of intersections of the line with the control polygonal line

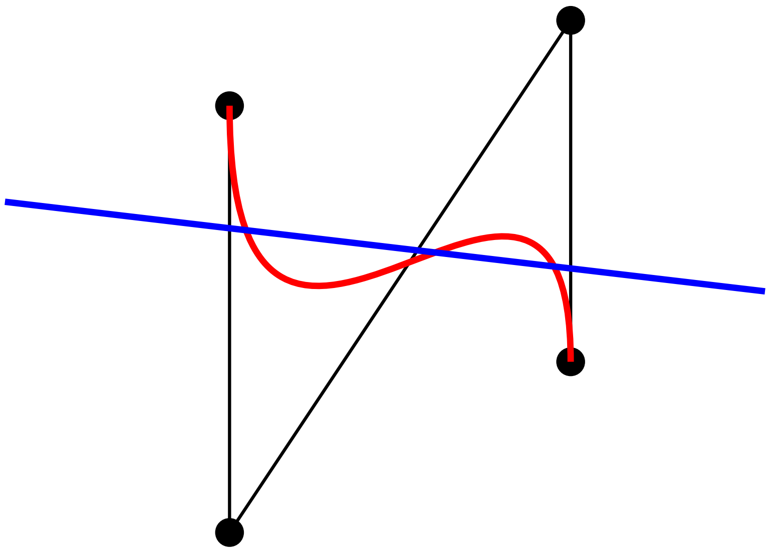


Properties of Bézier curves

7. Variation-diminishing property

The number of intersections of any line with a Bézier curve is at most the number of intersections of the line with the control polygonal line

This means that, to some extent, the curve imitates the shape and is not "rougher" than the corresponding control polygon,



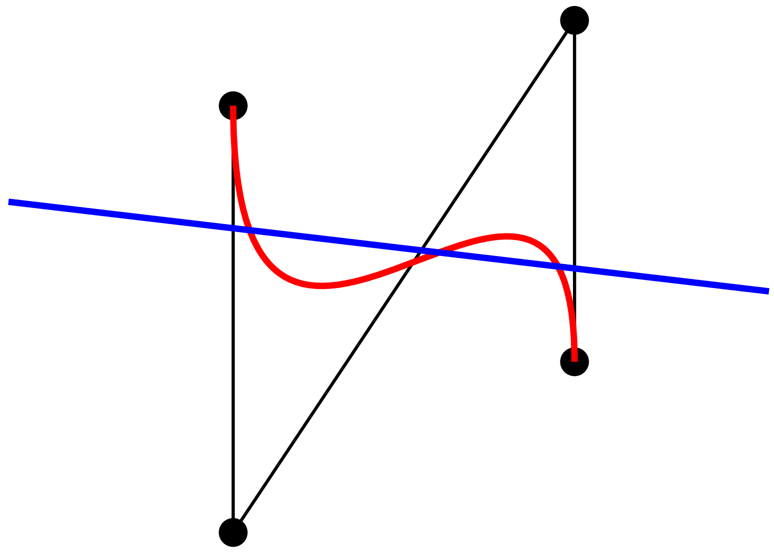
Proof? Later, after looking at **degree elevation**

Properties of Bézier curves

7. Variation-diminishing property

The number of intersections of any line with a Bézier curve is at most the number of intersections of the line with the control polygonal line

This means that, to some extent, the curve imitates the shape and is not "rougher" than the corresponding control polygon,



Proof? Later, after looking at **degree elevation**

Question: What does it imply for control points in convex position?

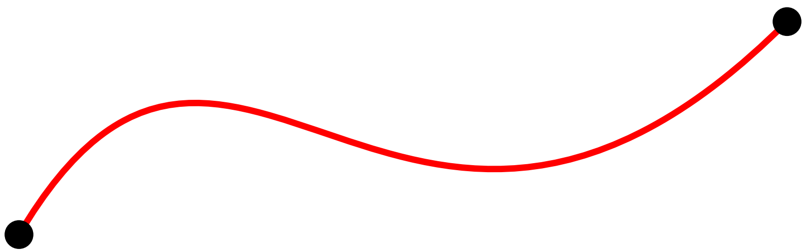
Connecting two curves

- In practice, one should avoid high-degree Bézier curves
- Better use many low-degree curves (they give local control)
- Requires smooth connection between consecutive curves

COMPOSITE BÉZIER CURVES

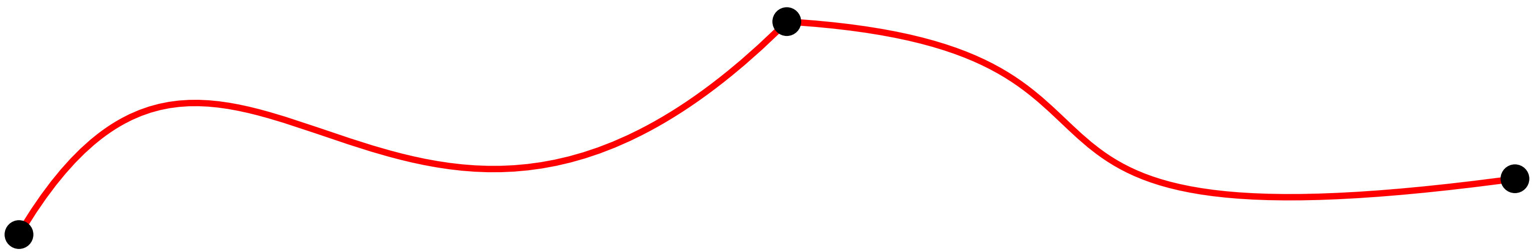
Connecting two curves

- In practice, one should avoid high-degree Bézier curves
- Better use many low-degree curves (they give local control)
- Requires smooth connection between consecutive curves



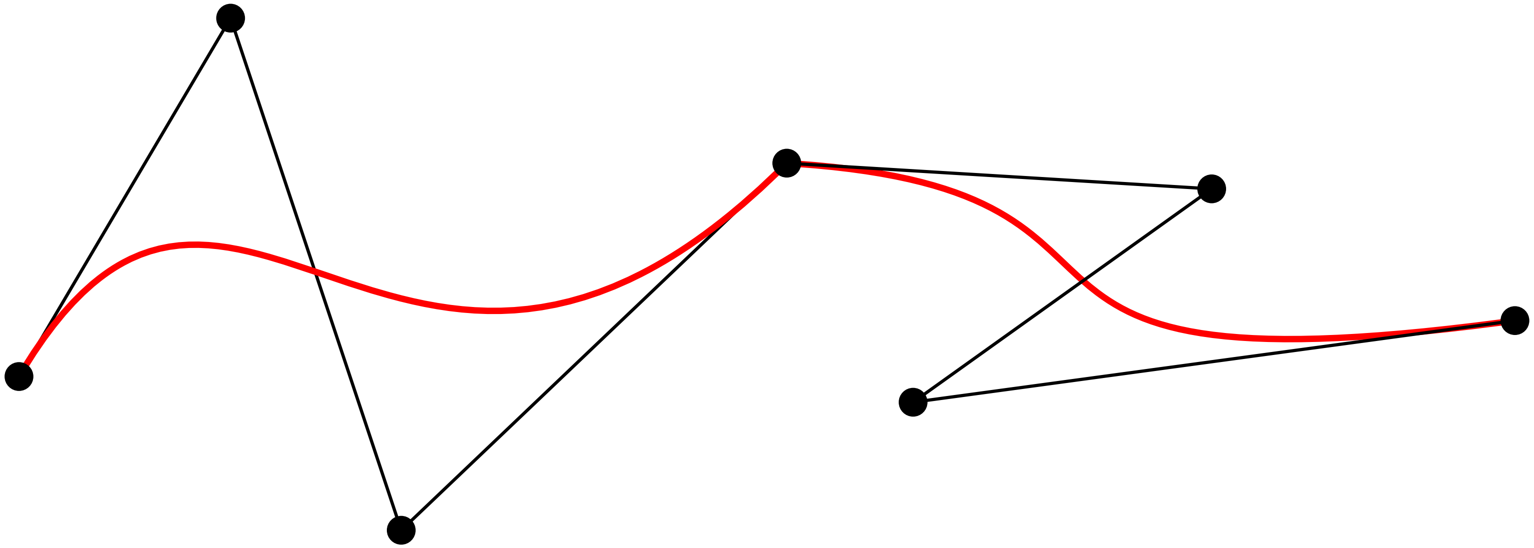
Connecting two curves

- In practice, one should avoid high-degree Bézier curves
- Better use many low-degree curves (they give local control)
- Requires smooth connection between consecutive curves



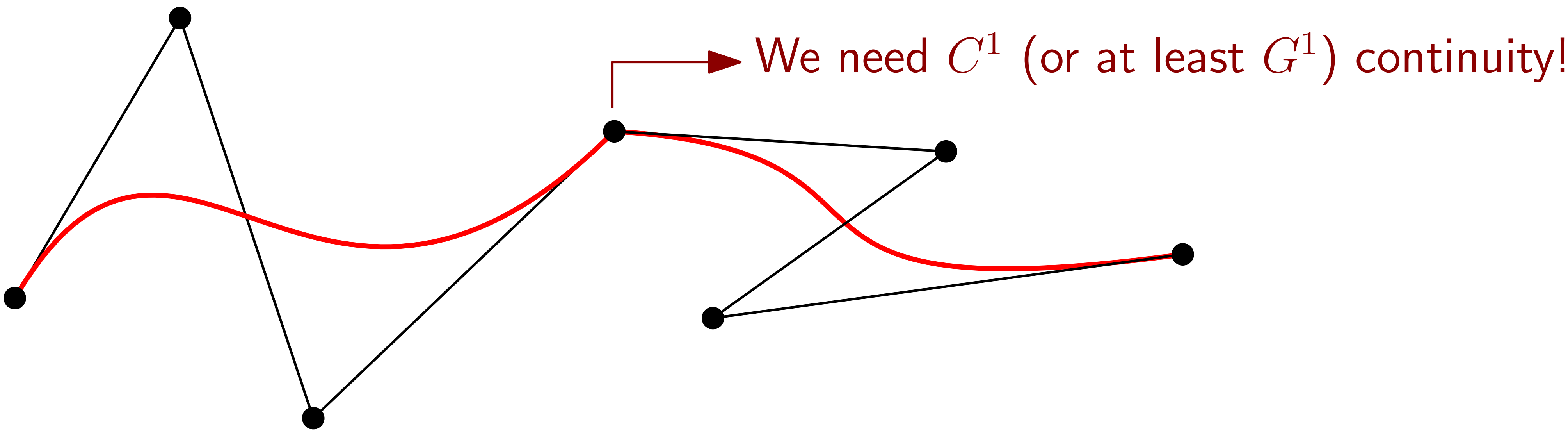
Connecting two curves

- In practice, one should avoid high-degree Bézier curves
- Better use many low-degree curves (they give local control)
- Requires smooth connection between consecutive curves



Connecting two curves

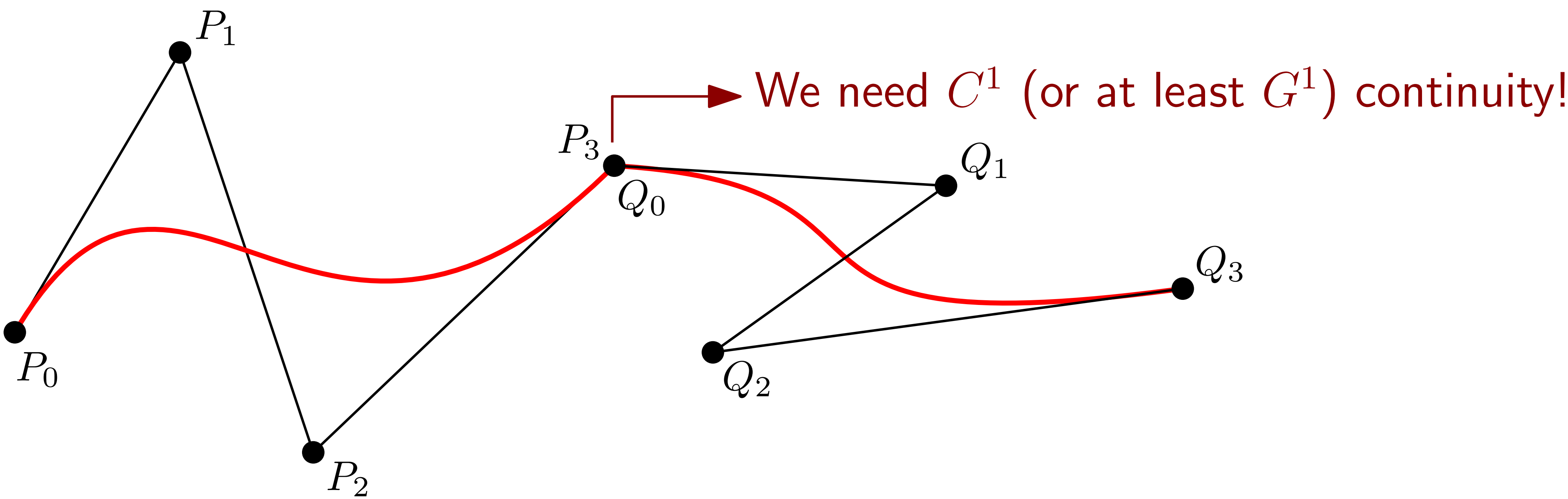
- In practice, one should avoid high-degree Bézier curves
- Better use many low-degree curves (they give local control)
- Requires smooth connection between consecutive curves



COMPOSITE BÉZIER CURVES

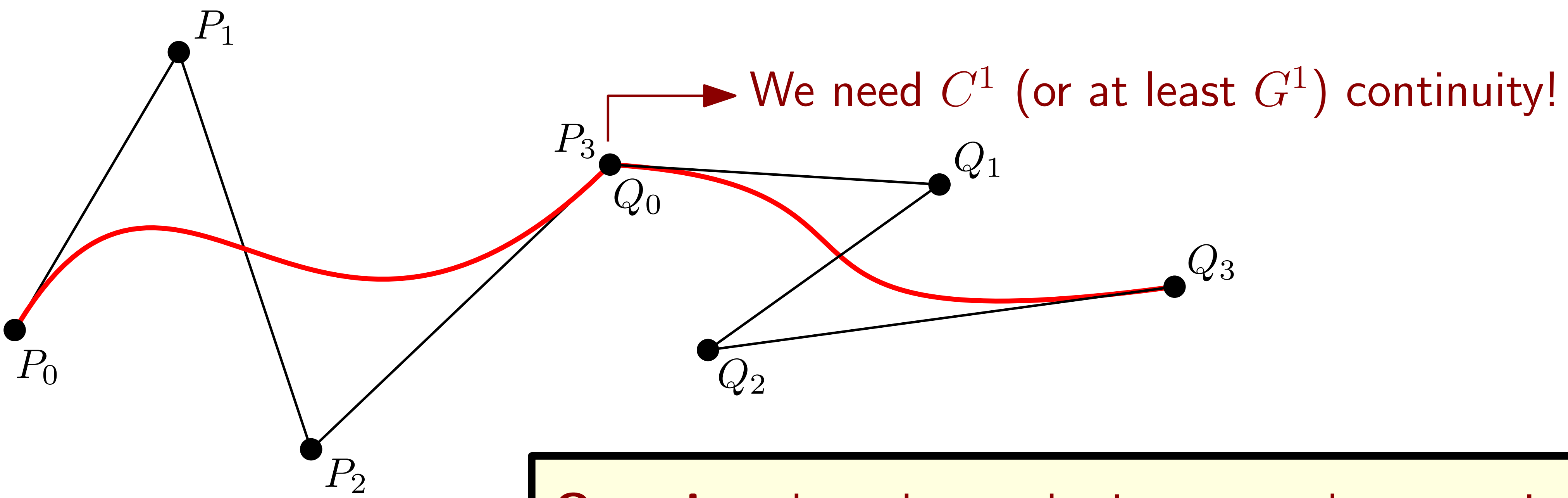
Connecting two curves

- In practice, one should avoid high-degree Bézier curves
- Better use many low-degree curves (they give local control)
- Requires smooth connection between consecutive curves



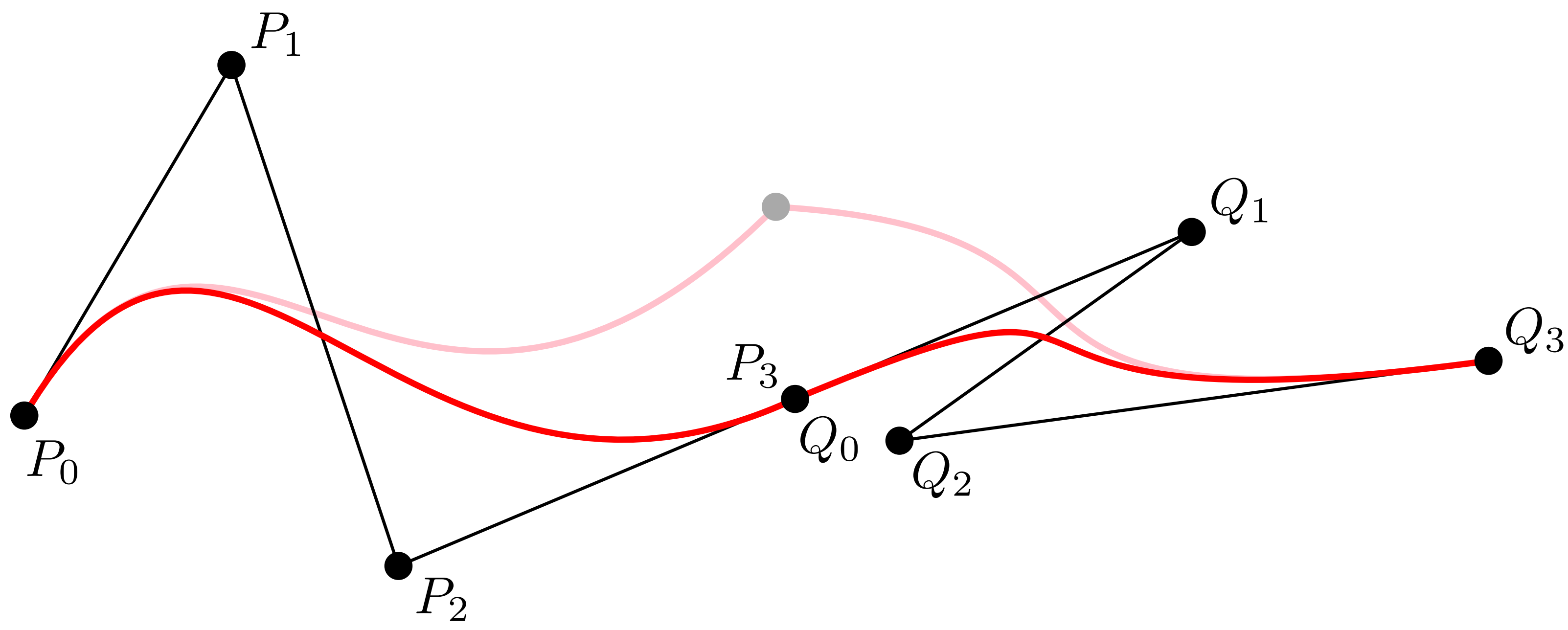
Connecting two curves

- In practice, one should avoid high-degree Bézier curves
- Better use many low-degree curves (they give local control)
- Requires smooth connection between consecutive curves

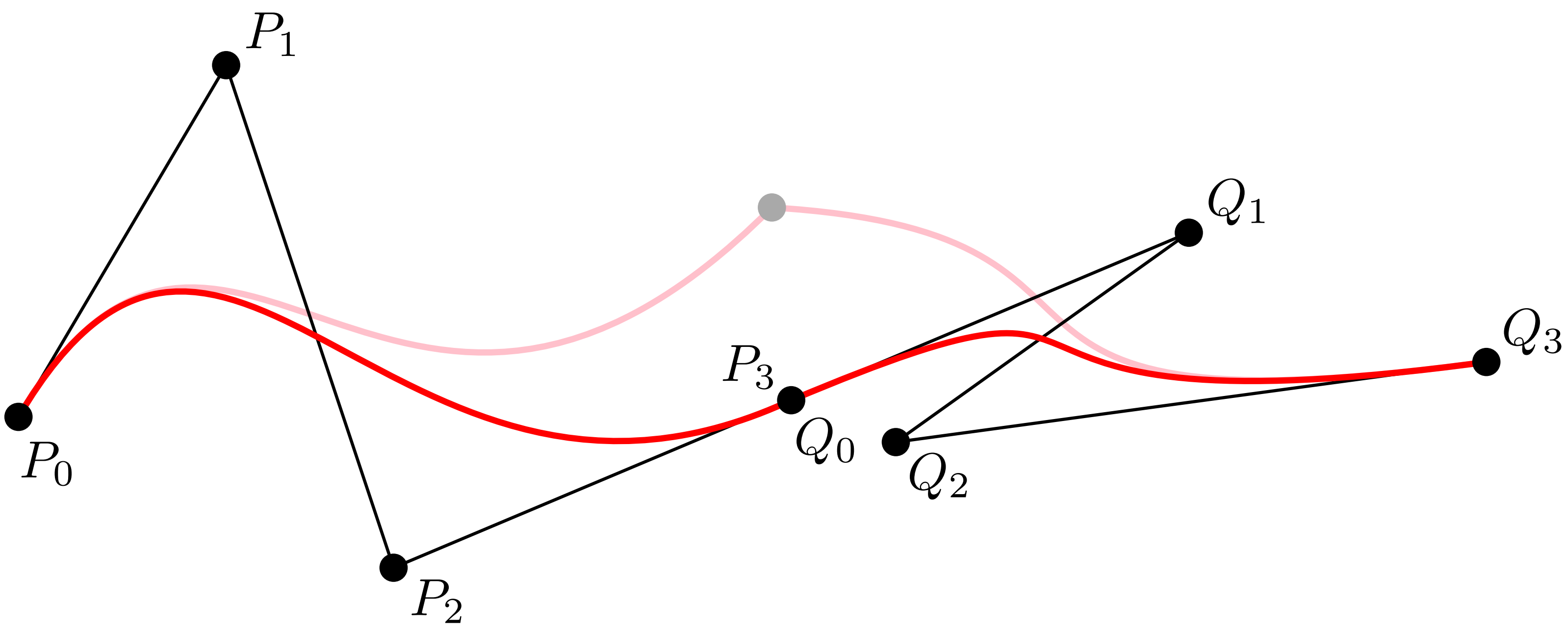


> **Question:** how do we obtain a smooth connection?

Connecting two curves



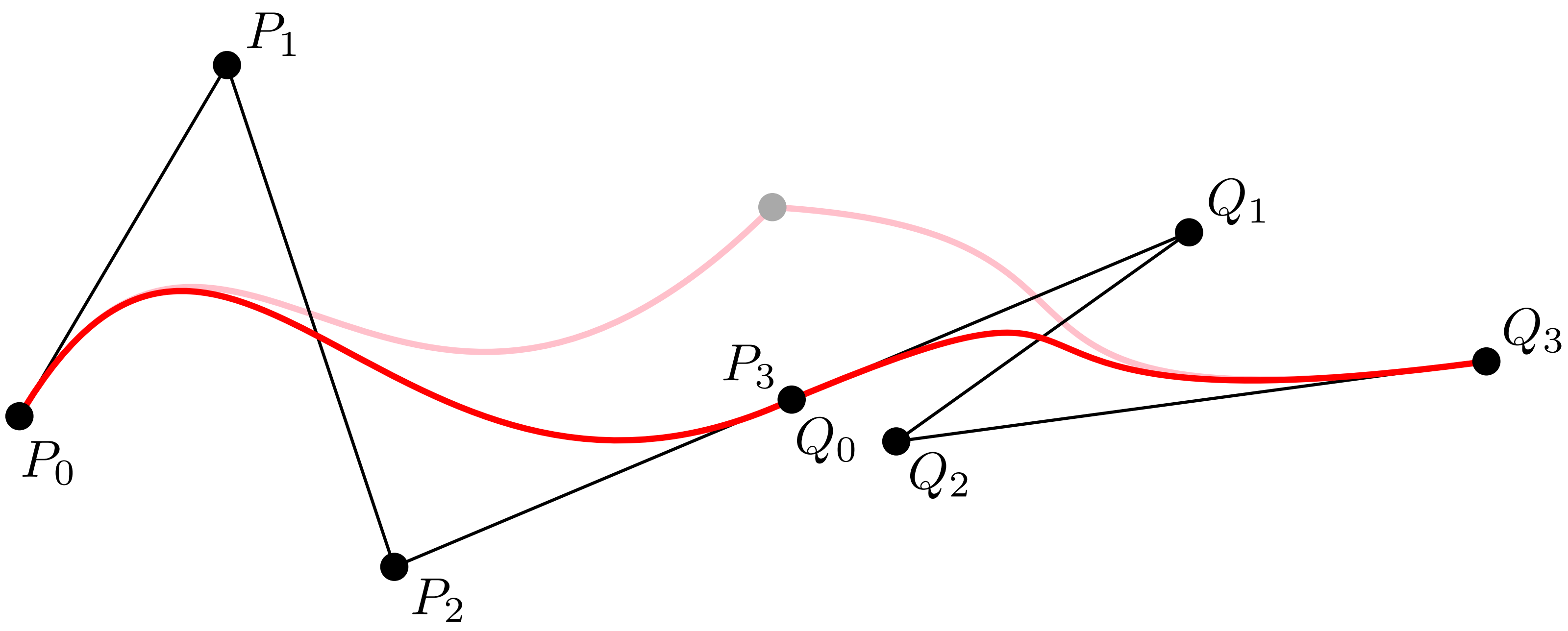
Connecting two curves



In general, for a curve P with $(n+1)$ control points and Q with $(m+1)$, the C^1-continuity condition is

$$Q_0 = P_n = \frac{m}{m+n}Q_1 + \frac{n}{m+n}P_{n-1}$$

Connecting two curves



In general, for a curve P with $(n+1)$ control points and Q with $(m+1)$, the C^1-continuity condition is

$$Q_0 = P_n = \frac{m}{m+n} Q_1 + \frac{n}{m+n} P_{n-1}$$

Question: how can you obtain higher-degree continuity?

? ?

EXAMPLE: FONT DESIGN

Guess how are the fonts you use designed?

EXAMPLE: FONT DESIGN

Guess how are the fonts you use designed?

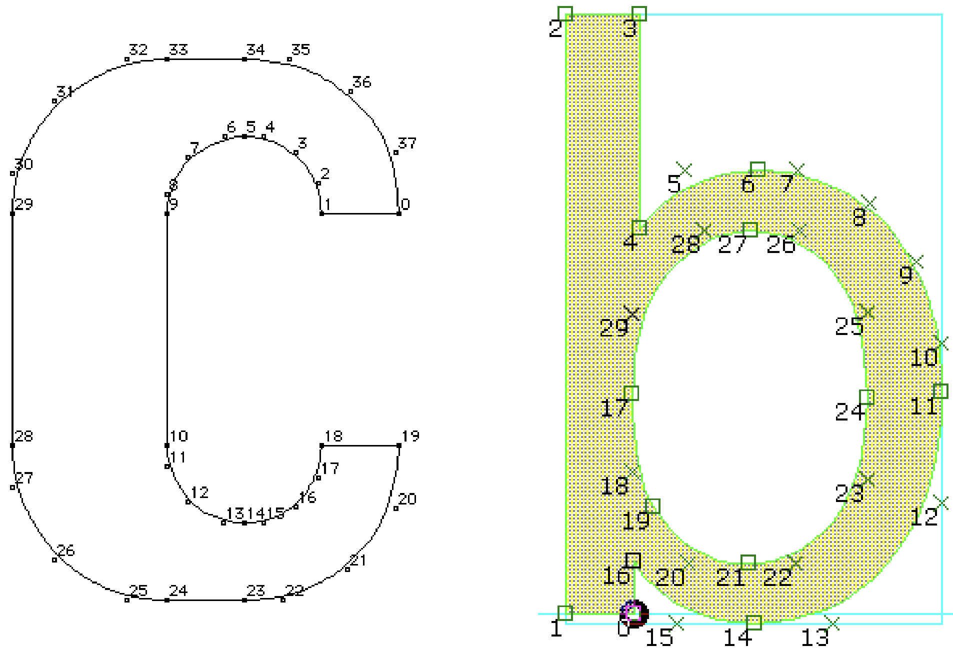
- True-type fonts (Apple, Microsoft): quadratic Bézier curves

Guess how are the fonts you use designed?

- True-type fonts (Apple, Microsoft): quadratic Bézier curves
- PostScript (Adobe, printers,...) and OpenType (Microsoft, Adobe) fonts: cubic Bézier curves

Guess how are the fonts you use designed?

- True-type fonts (Apple, Microsoft): quadratic Bézier curves
- PostScript (Adobe, printers,...) and OpenType (Microsoft, Adobe) fonts: cubic Bézier curves



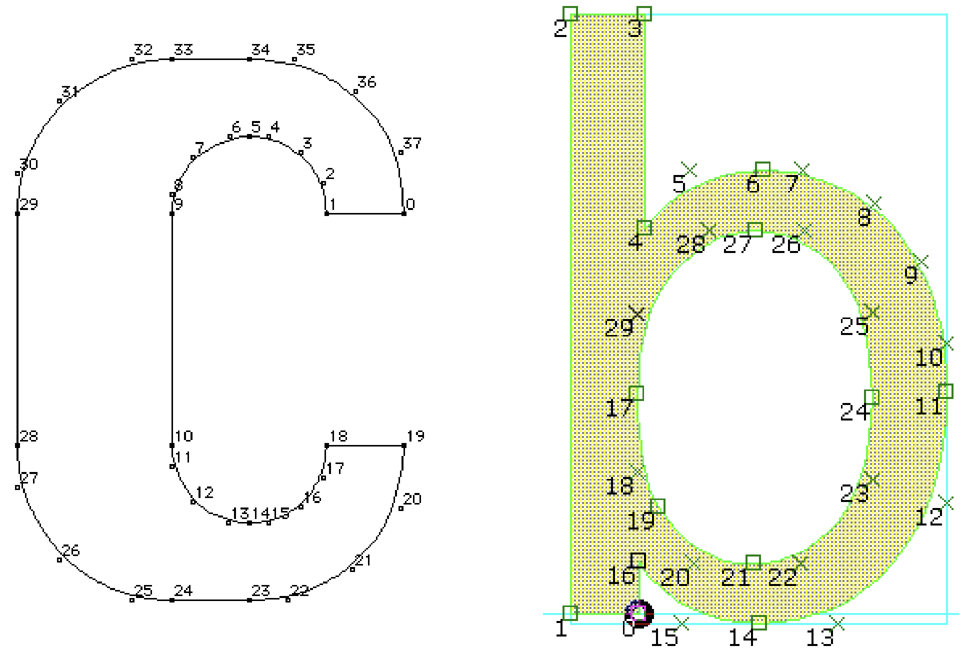
Glyphs of two characters in a true-type font

EXAMPLE: FONT DESIGN

Guess how are the fonts you use designed?

- True-type fonts (Apple, Microsoft): quadratic Bézier curves
- PostScript (Adobe, printers,...) and OpenType (Microsoft, Adobe) fonts: cubic Bézier curves

> **Question:** Can you convert between these types of fonts?



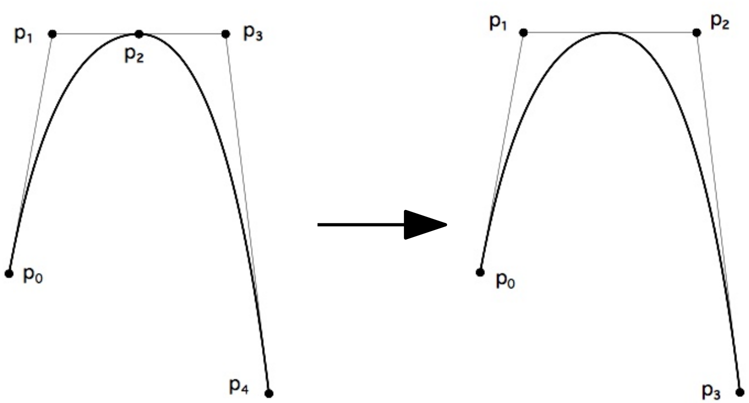
Glyphs of two characters in a true-type font

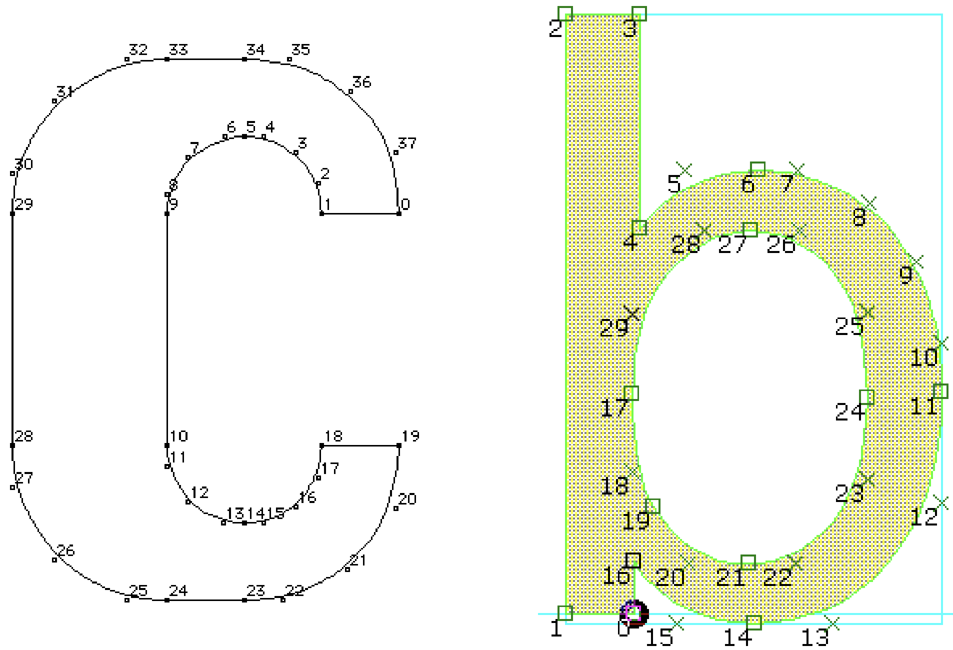
Guess how are the fonts you use designed?

- True-type fonts (Apple, Microsoft): quadratic Bézier curves
- PostScript (Adobe, printers,...) and OpenType (Microsoft, Adobe) fonts: cubic Bézier curves

> **Question:** Can you convert between these types of fonts?

- Storage of glyphs in TTF:





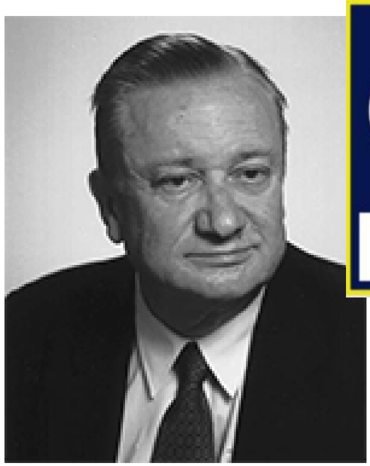
Glyphs of two characters in a true-type font

Difference between points on-curve and off-curve (only off-curve points are stored)

BÉZIER CURVES AS LINEAR INTERPOLATION

An alternative approach to Bézier curves

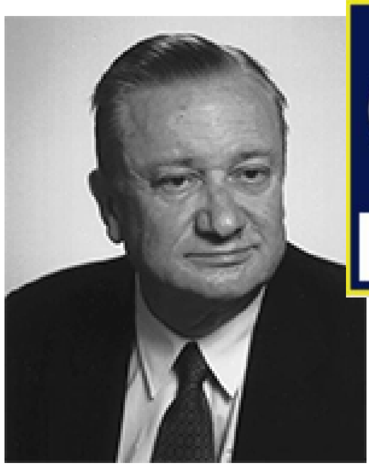
De Casteljau (Citroën) followed a different approach based on **repeated linear interpolation**

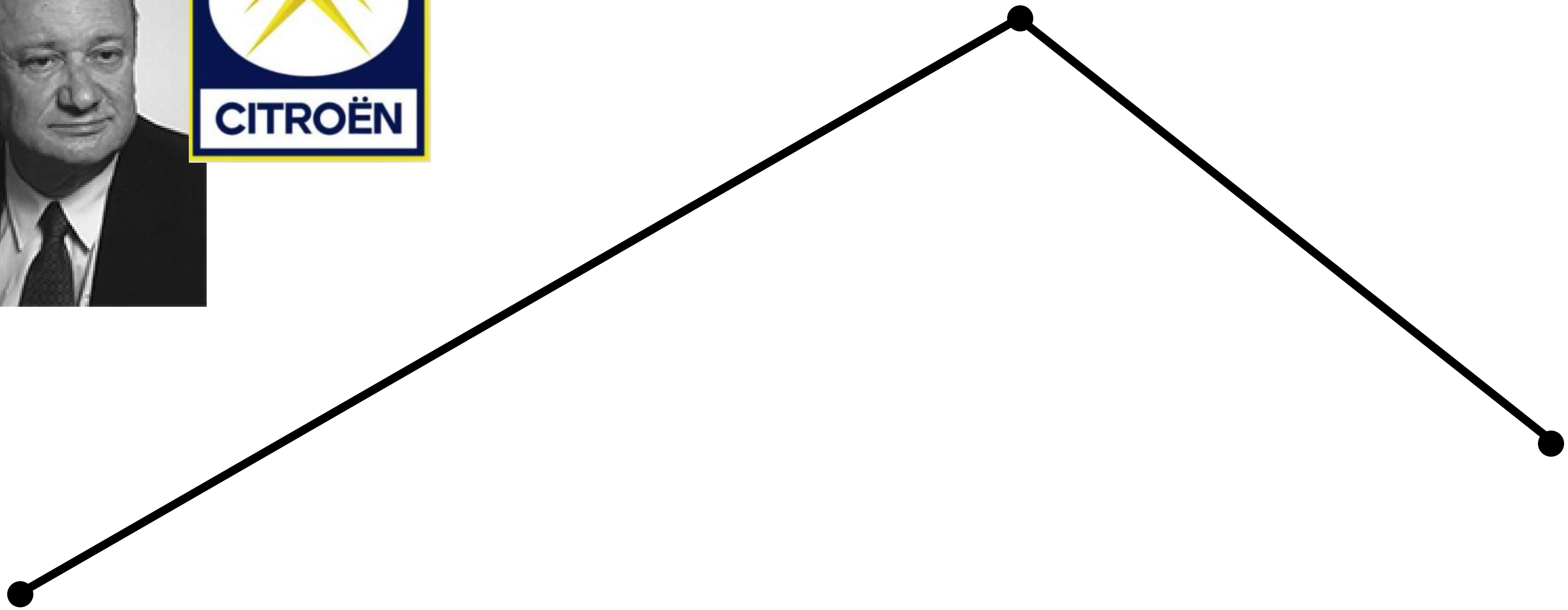


BÉZIER CURVES AS LINEAR INTERPOLATION

An alternative approach to Bézier curves

De Casteljau (Citroën) followed a different approach based on **repeated linear interpolation**

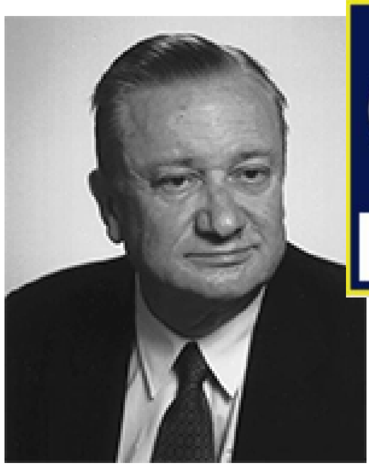


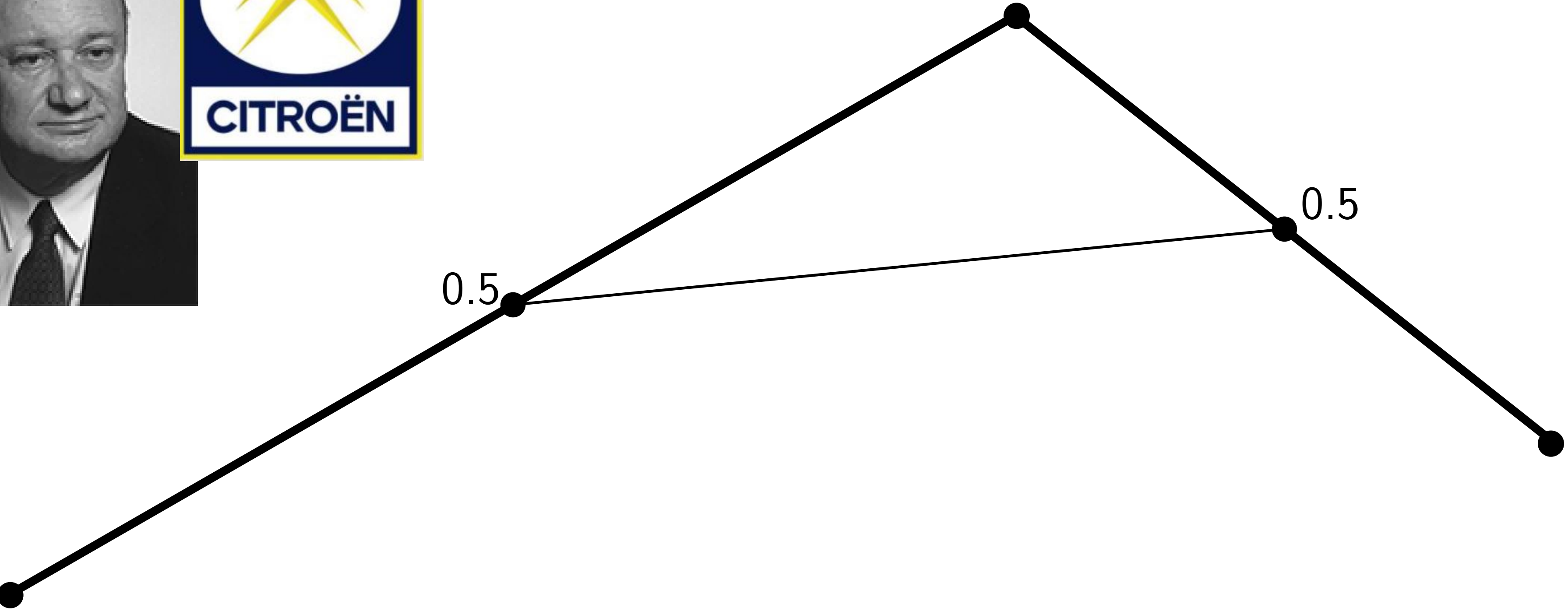


BÉZIER CURVES AS LINEAR INTERPOLATION

An alternative approach to Bézier curves

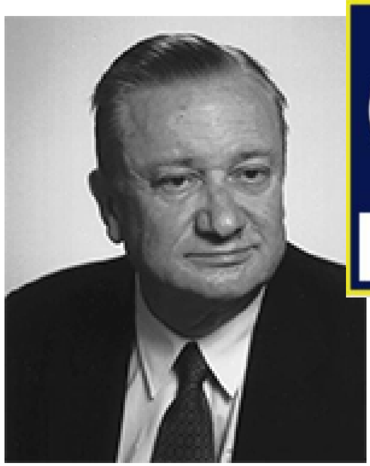
De Casteljau (Citroën) followed a different approach based on **repeated linear interpolation**

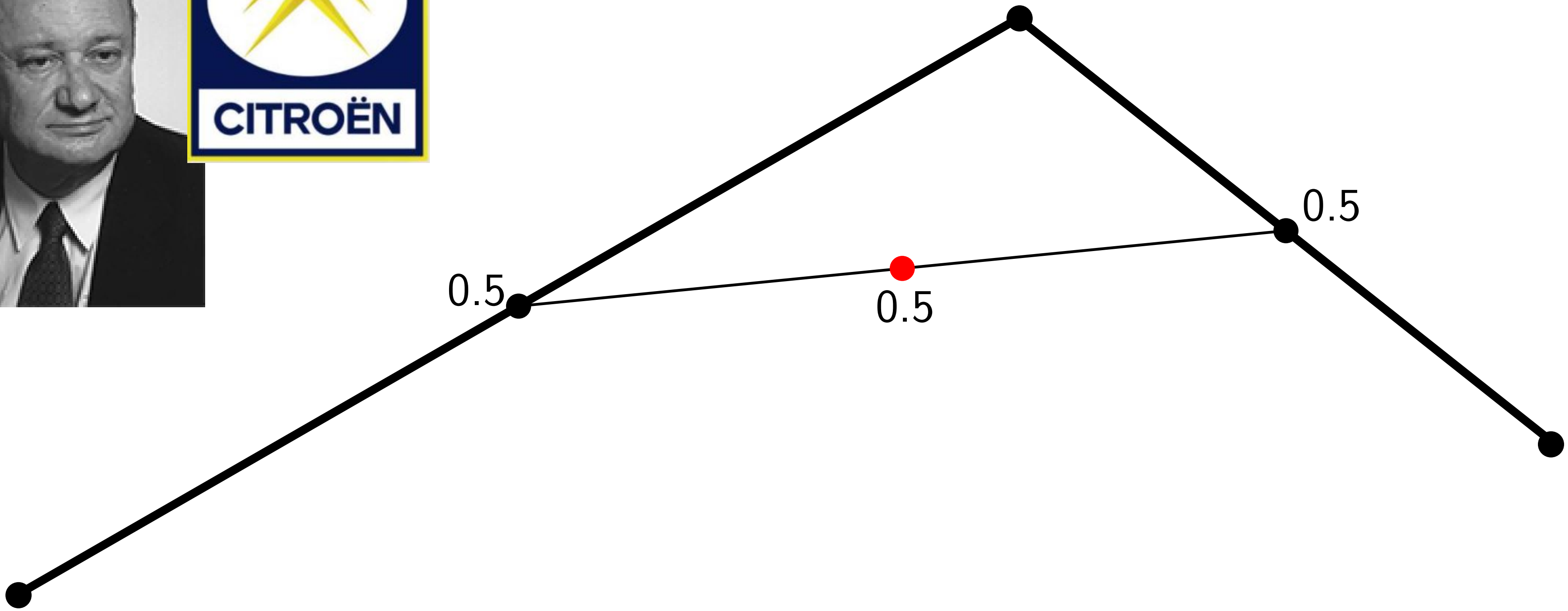




An alternative approach to Bézier curves

De Casteljau (Citroën) followed a different approach based on **repeated linear interpolation**

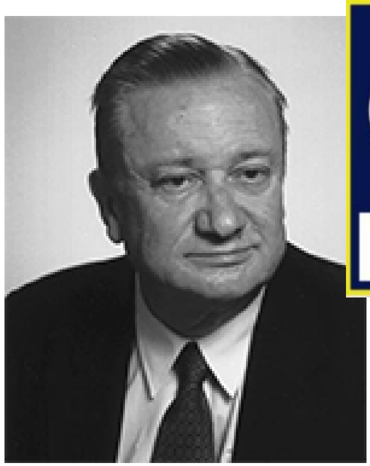


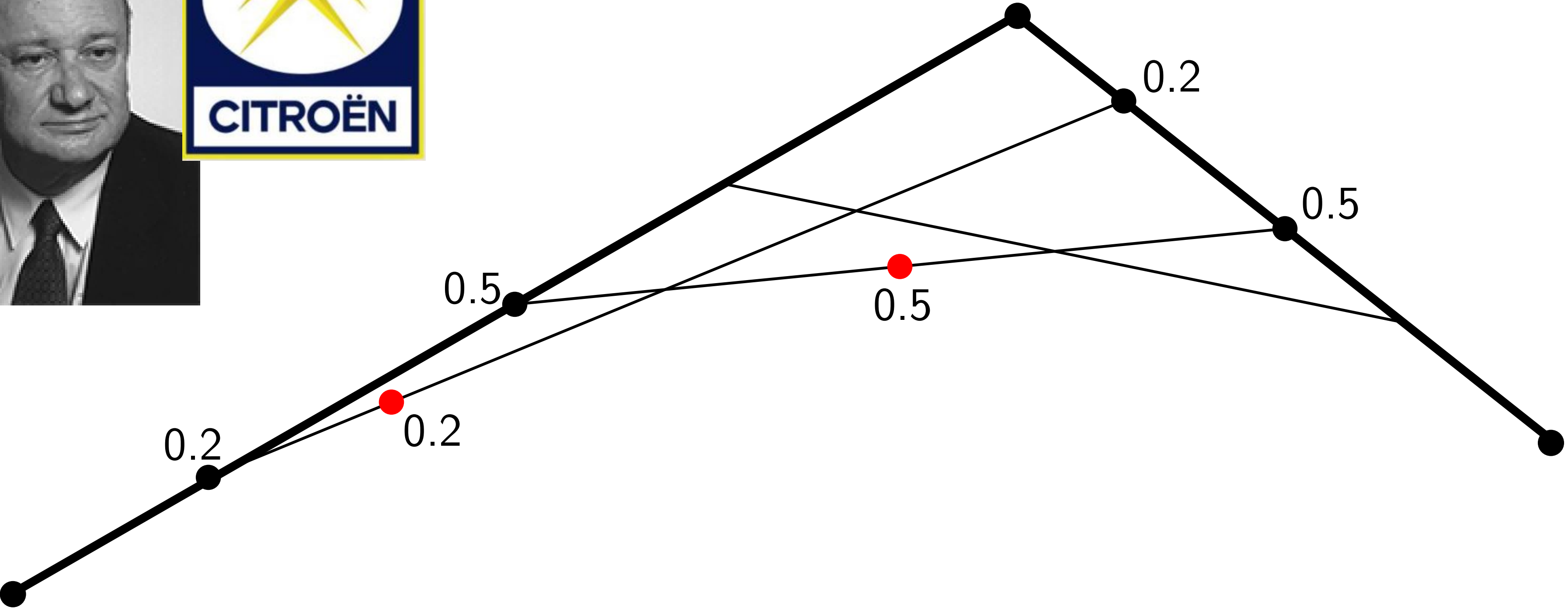


BÉZIER CURVES AS LINEAR INTERPOLATION

An alternative approach to Bézier curves

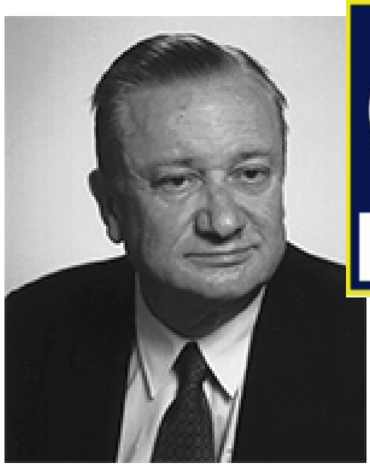
De Casteljau (Citroën) followed a different approach based on **repeated linear interpolation**

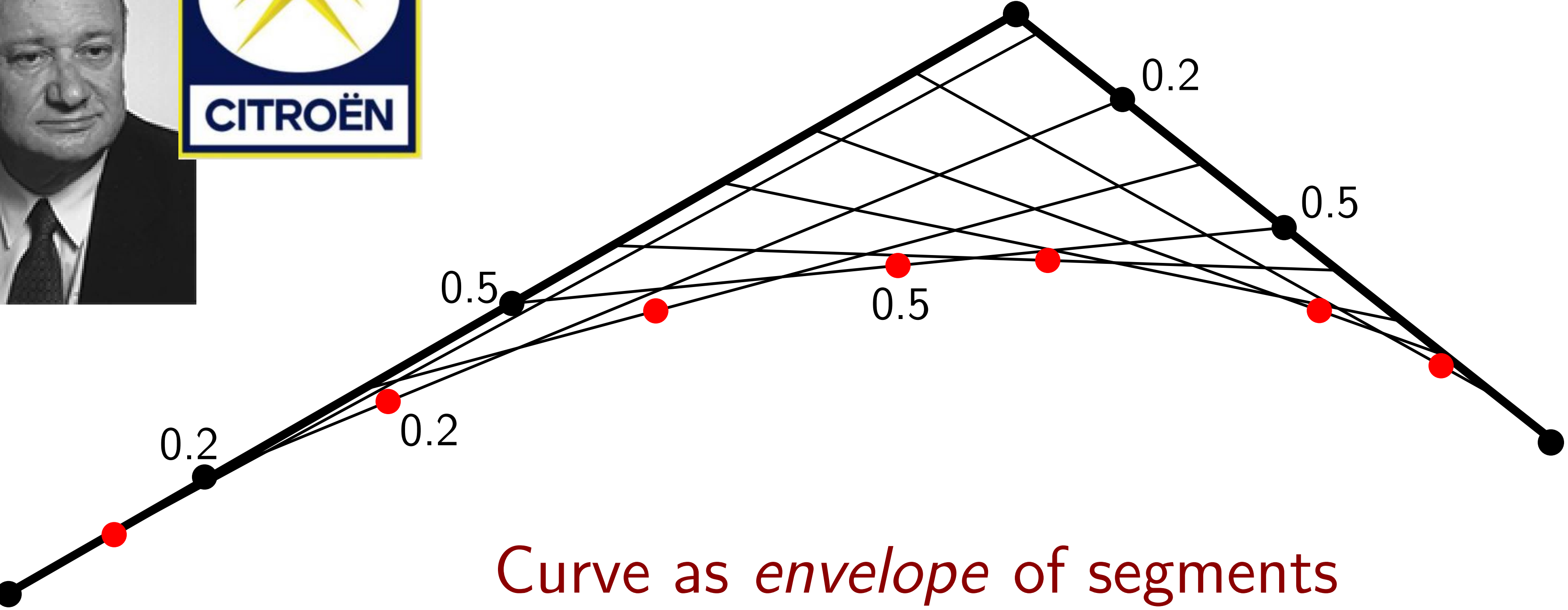




An alternative approach to Bézier curves

De Casteljau (Citroën) followed a different approach based on **repeated linear interpolation**

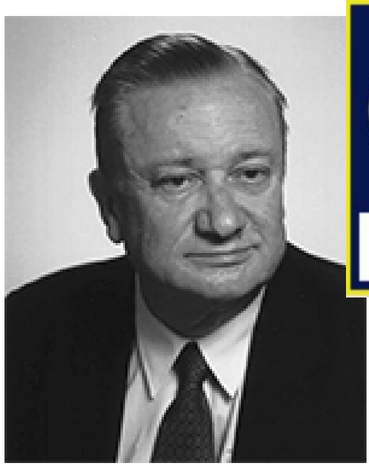


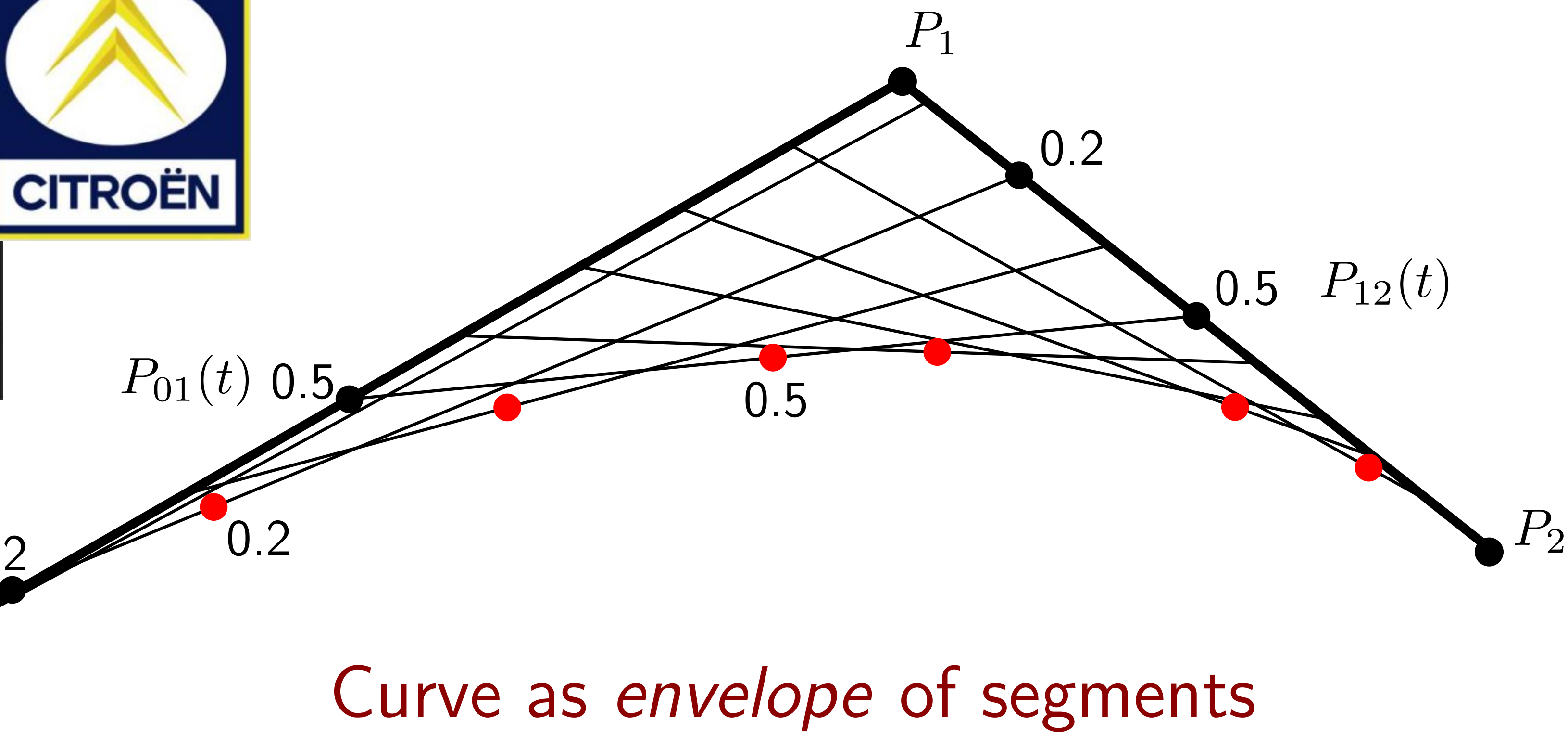


Curve as *envelope* of segments

An alternative approach to Bézier curves

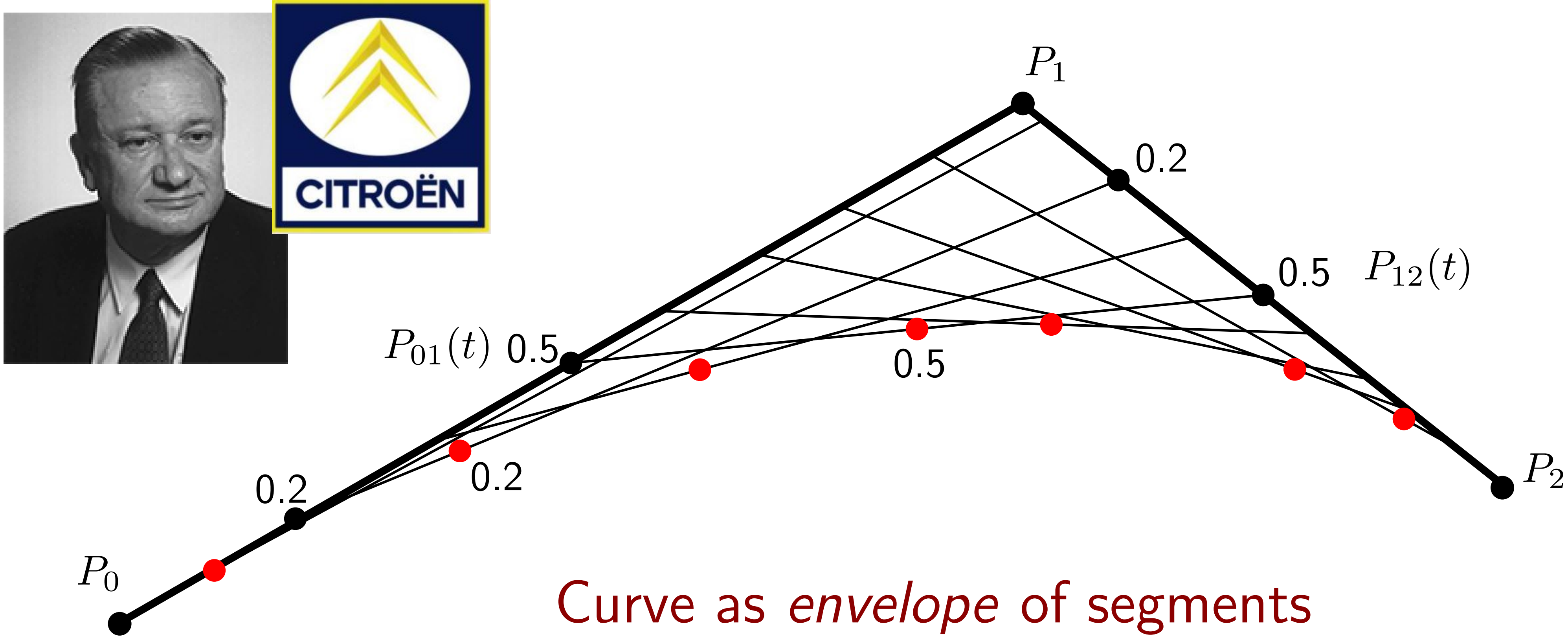
De Casteljau (Citroën) followed a different approach based on **repeated linear interpolation**





An alternative approach to Bézier curves

De Casteljau (Citroën) followed a different approach based on **repeated linear interpolation**



Curve as *envelope* of segments

> **Question:** What is the expression of this envelope, as a function of t?

De Casteljau's algorithm

For 3 points P_0, P_1, P_2, and $0 \leq t \leq 1$, we have:

$$P_{01}(t) = (1-t)P_0 + tP_1$$
$$P_{12}(t) = (1-t)P_1 + tP_2$$
$$P(t) = P_{012}(t) = (1-t)P_{01}(t) + tP_{12}(t)$$

De Casteljau's algorithm

For 3 points P_0, P_1, P_2, and $0 \leq t \leq 1$, we have:

$$P_{01}(t) = (1-t)P_0 + tP_1$$
$$P_{12}(t) = (1-t)P_1 + tP_2$$
$$P(t) = P_{012}(t) = (1-t)P_{01}(t) + tP_{12}(t)$$

This repeated linear interpolation process can be generalized to n points

De Casteljau's algorithm

For 3 points P_0, P_1, P_2, and $0 \leq t \leq 1$, we have:

$$
\begin{aligned}
P_{01}(t) &= (1-t)P_0 + tP_1 \\
P_{12}(t) &= (1-t)P_1 + tP_2 \\
P(t) = P_{012}(t) &= (1-t)P_{01}(t) + tP_{12}(t)
\end{aligned}
$$

This repeated linear interpolation process can be generalized to n points

For n points $P_0, \ldots, P_n$, $0 \leq t \leq 1$, and $0 \leq i \leq j \leq n$, we have:

$$
\begin{aligned}
P_i(t) &= P_i \\
P_{i(i+1)\ldots(j-1)j}(t) &= (1-t)P_{i\ldots(j-1)}(t) + tP_{(i+1)\ldots j}(t)
\end{aligned}
$$

De Casteljau's algorithm

For 3 points P_0, P_1, P_2, and $0 \leq t \leq 1$, we have:

$$P_{01}(t) = (1-t)P_0 + tP_1$$
$$P_{12}(t) = (1-t)P_1 + tP_2$$
$$P(t) = P_{012}(t) = (1-t)P_{01}(t) + tP_{12}(t)$$

This repeated linear interpolation process can be generalized to n points

For n points $P_0, \ldots, P_n$, $0 \leq t \leq 1$, and $0 \leq i \leq j \leq n$, we have:

$$\begin{aligned} P_i(t) &= P_i \\ P_{i(i+1)\ldots(j-1)j}(t) &= (1-t)P_{i\ldots(j-1)}(t) + tP_{(i+1)\ldots j}(t) \end{aligned}$$

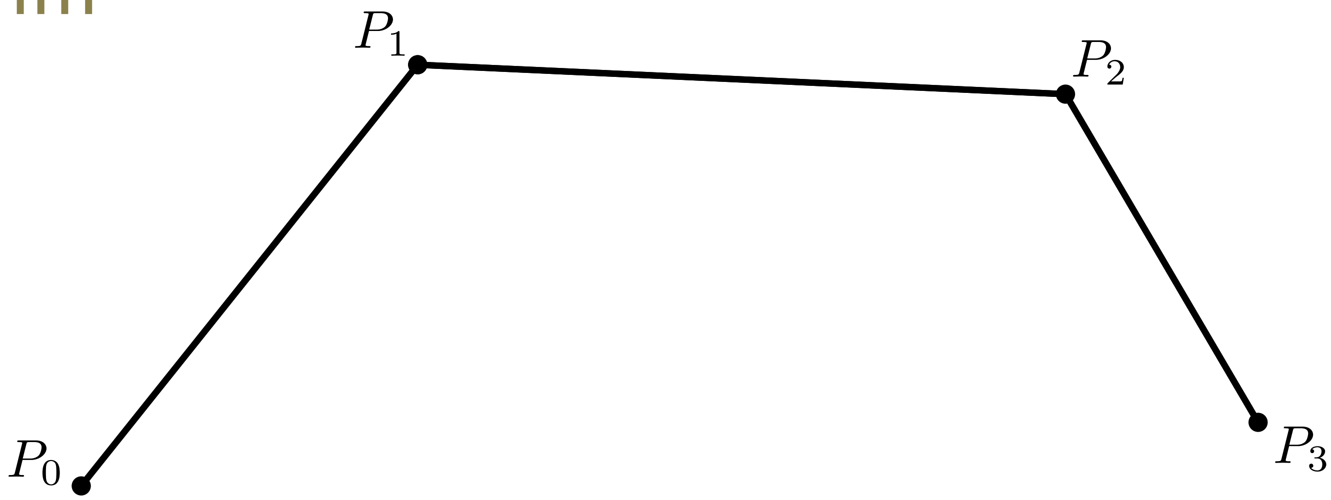
Known as the *Recursive or geometric construction method*

The final curve is given by $P(t) = P_{0\ldots n}(t)$

De Casteljau's algorithm

Example for $n = 3$ and $t = 1/2$

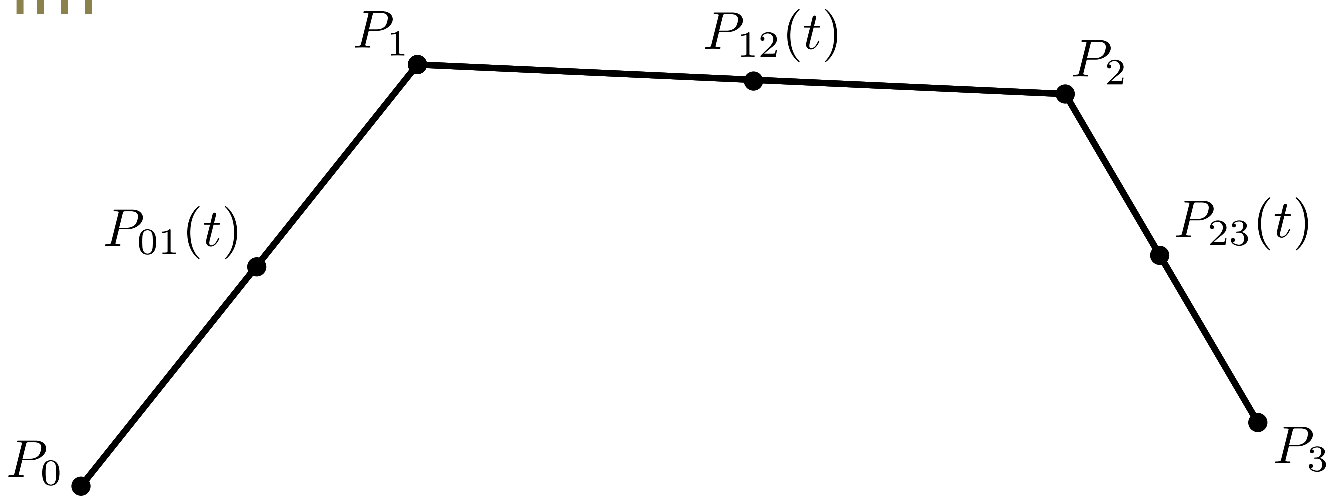
$P(t) = P_{0123}(t)$



De Casteljau's algorithm

Example for $n = 3$ and $t = 1/2$

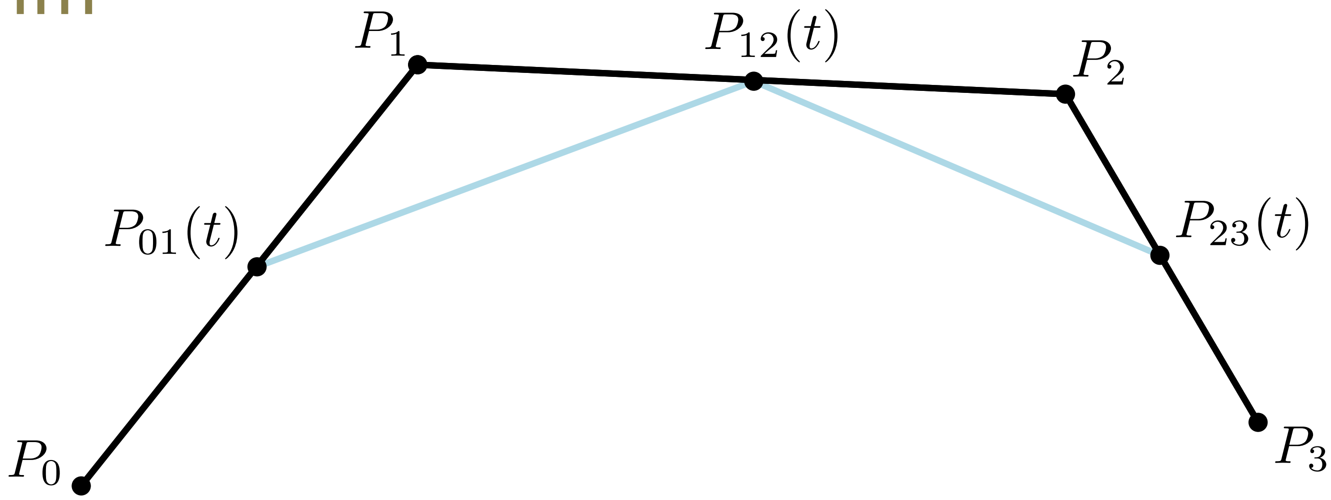
$P(t) = P_{0123}(t)$



De Casteljau's algorithm

Example for $n = 3$ and $t = 1/2$

$P(t) = P_{0123}(t)$

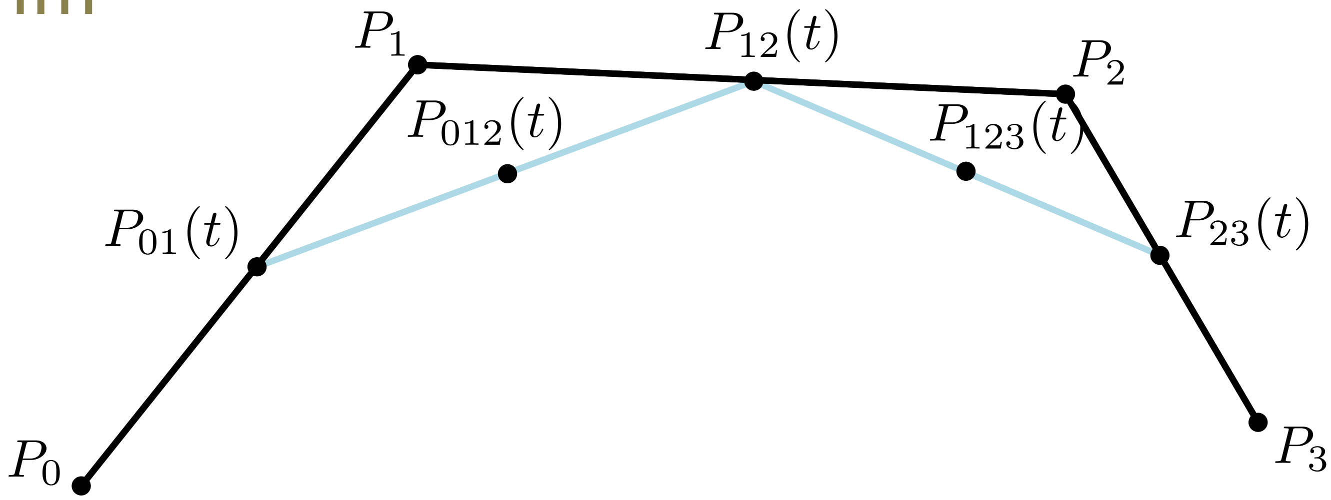


BÉZIER CURVES AS LINEAR INTERPOLATION

De Casteljau's algorithm

Example for $n = 3$ and $t = 1/2$

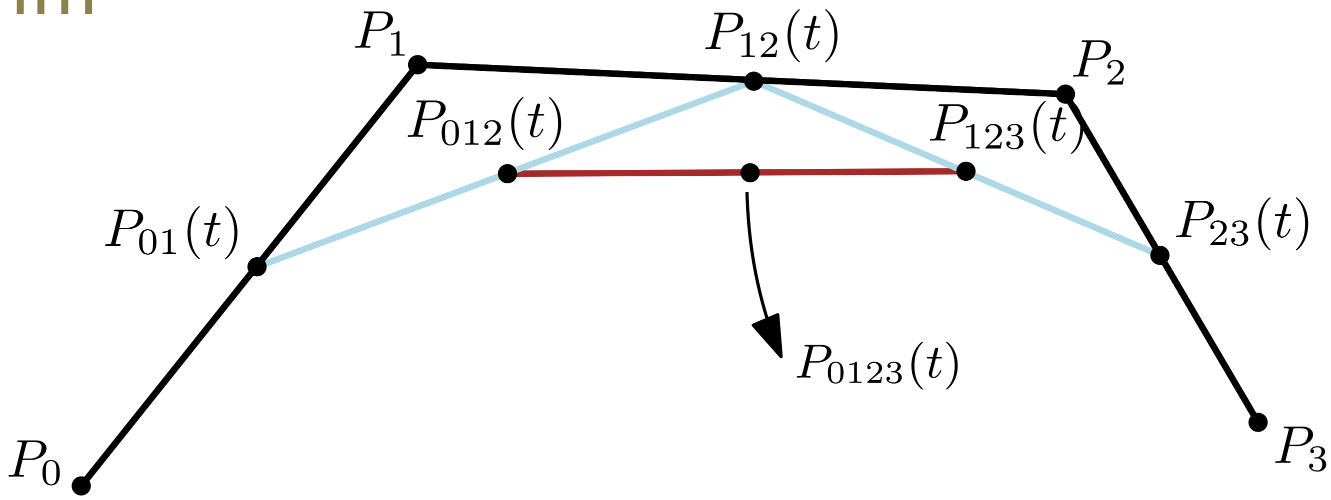
$P(t) = P_{0123}(t)$



De Casteljau's algorithm

Example for $n = 3$ and $t = 1/2$

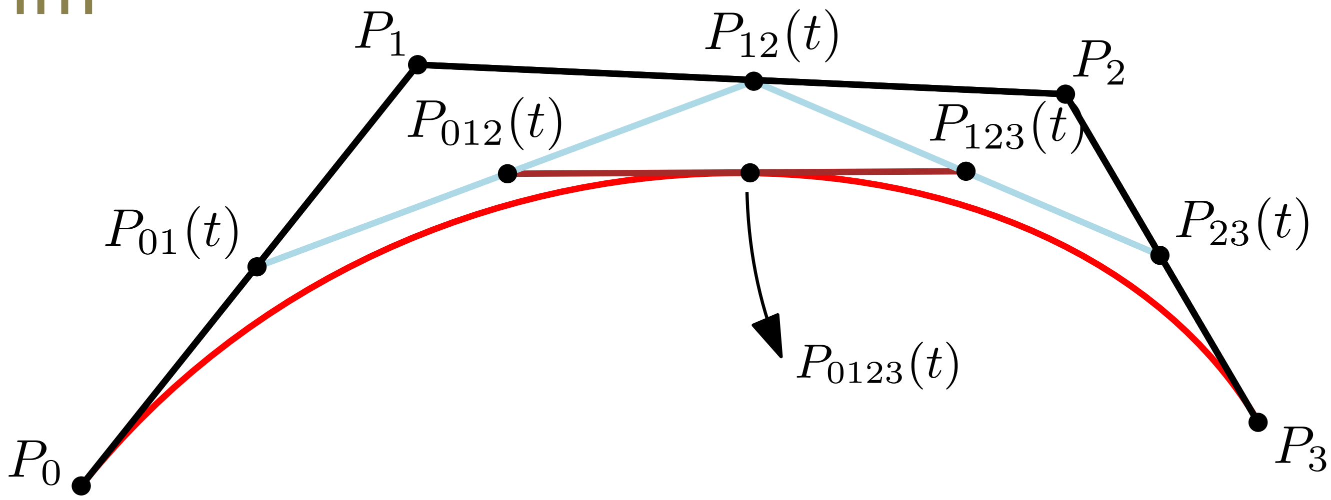
$P(t) = P_{0123}(t)$



De Casteljau's algorithm

Example for $n = 3$ and $t = 1/2$

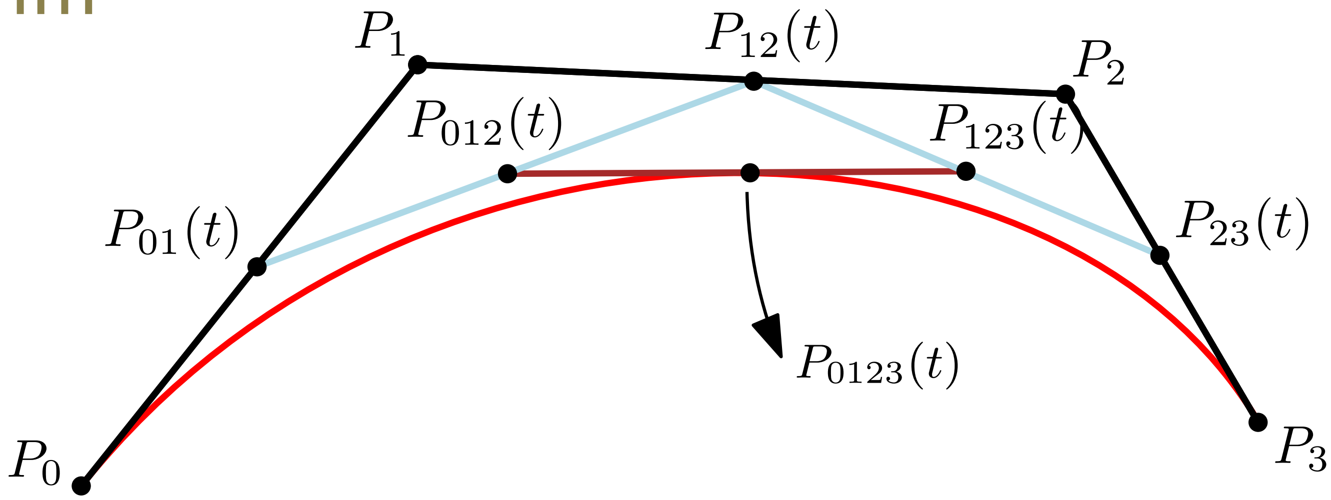
$$P(t) = P_{0123}(t)$$



De Casteljau's algorithm

Example for $n = 3$ and $t = 1/2$

$$P(t) = P_{0123}(t)$$



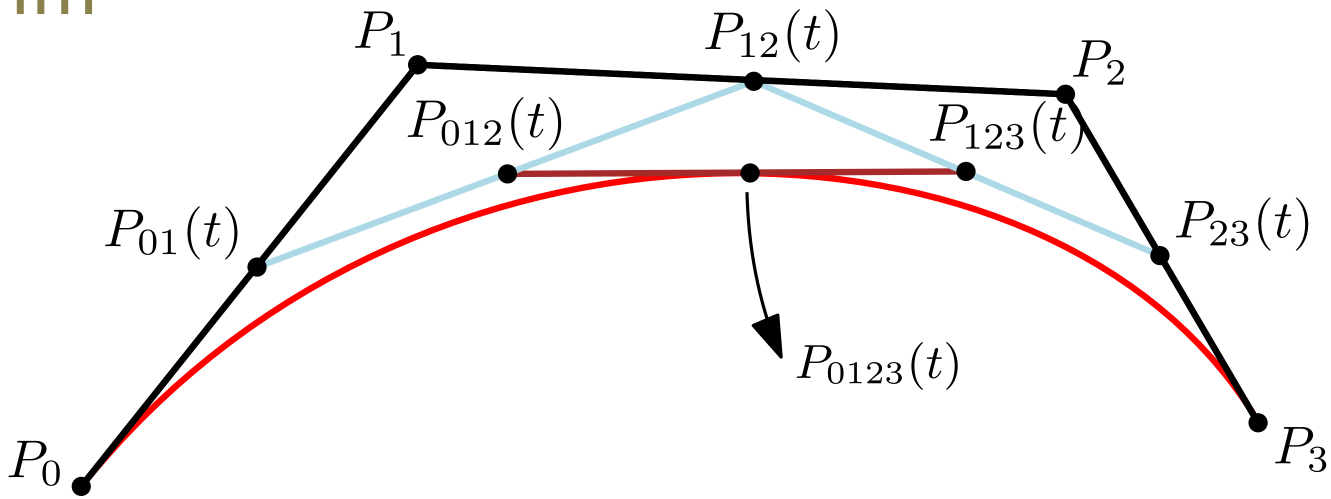
See `https://javascript.info/bezier-curve` for several animations

BÉZIER CURVES AS LINEAR INTERPOLATION

De Casteljau's algorithm

Example for $n = 3$ and $t = 1/2$

$P(t) = P_{0123}(t)$



Implementation of the algorithm

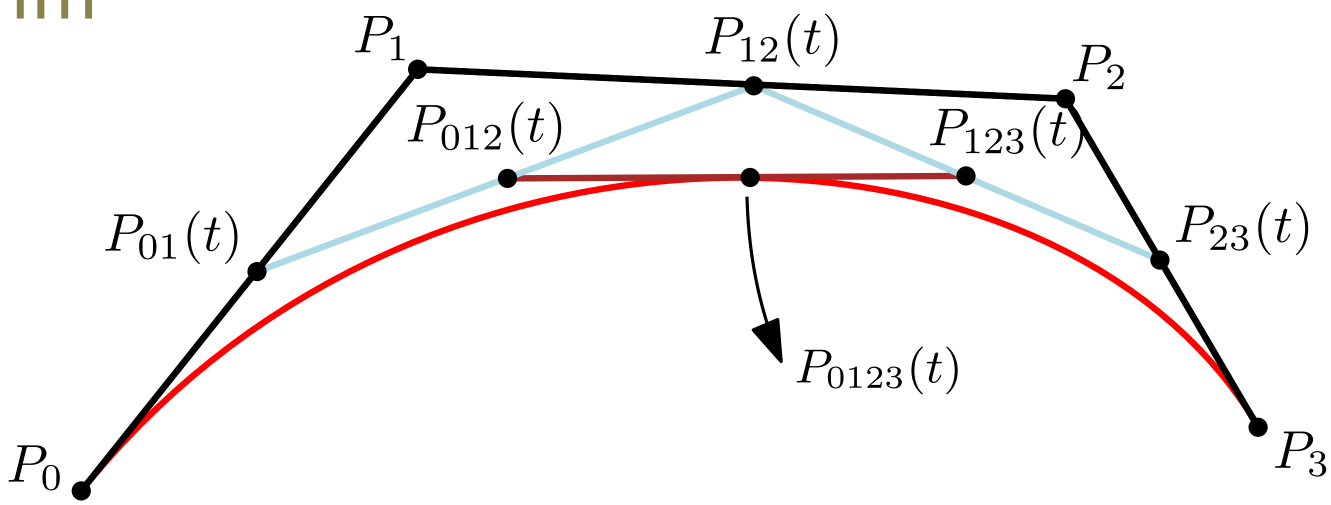
How to evaluate $P(1/2)$?

BÉZIER CURVES AS LINEAR INTERPOLATION

De Casteljau's algorithm

Example for $n = 3$ and $t = 1/2$

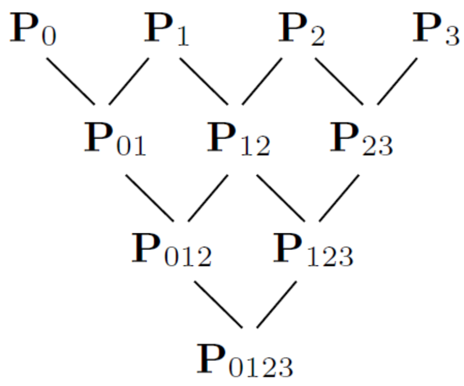
$$P(t) = P_{0123}(t)$$



Implementation of the algorithm

How to evaluate $P(1/2)$?

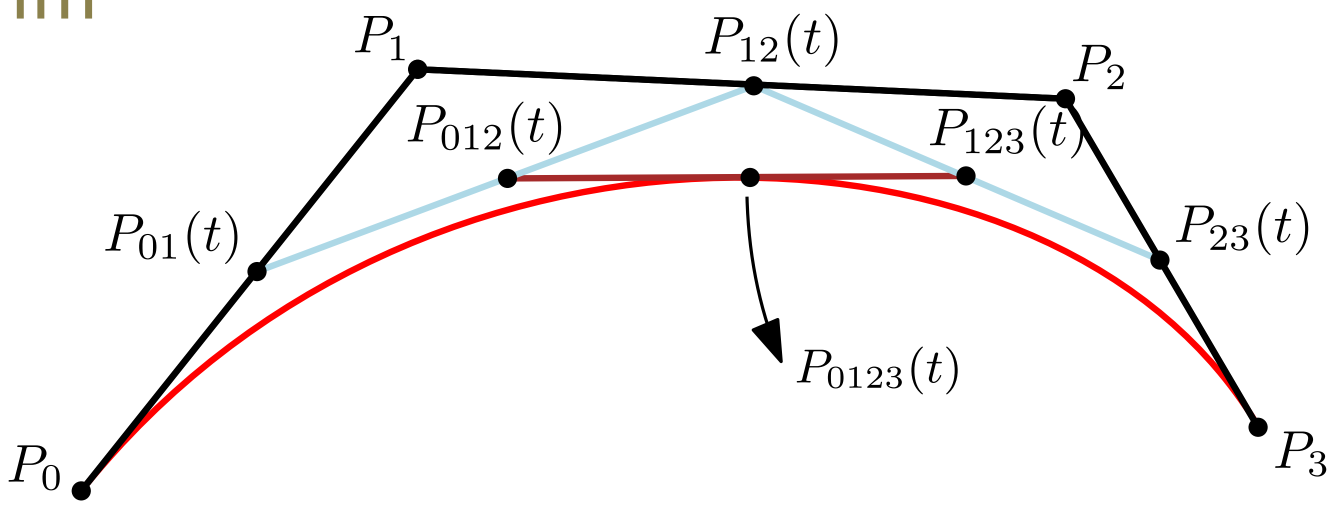
Step	Points constructed	#points
1	$\mathbf{P}_{01}\ \mathbf{P}_{12}\ \mathbf{P}_{23}\ldots\mathbf{P}_{n-1,n}$	n
2	$\mathbf{P}_{012}\ \mathbf{P}_{123}\ \mathbf{P}_{234}\ldots\mathbf{P}_{n-2,n-1,n}$	$n-1$
3	$\mathbf{P}_{0123}\ \mathbf{P}_{1234}\ \mathbf{P}_{2345}\ldots\mathbf{P}_{n-3,n-2,n-1,n}$	$n-2$
$\vdots$	$\vdots$	$\vdots$
n	$\mathbf{P}_{0123\ldots n}$	



De Casteljau's algorithm

Example for $n = 3$ and $t = 1/2$

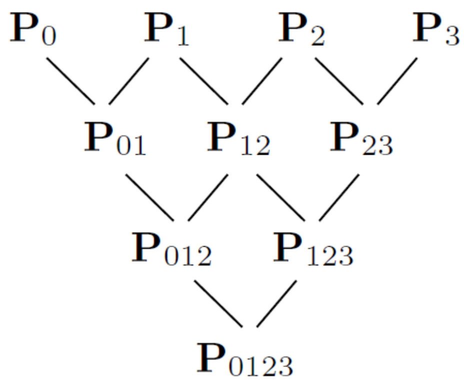
$P(t) = P_{0123}(t)$



Implementation of the algorithm

How to evaluate $P(1/2)$?

Step	Points constructed	#points
1	$\mathbf{P}_{01}\ \mathbf{P}_{12}\ \mathbf{P}_{23} \dots \mathbf{P}_{n-1,n}$	n
2	$\mathbf{P}_{012}\ \mathbf{P}_{123}\ \mathbf{P}_{234} \dots \mathbf{P}_{n-2,n-1,n}$	$n-1$
3	$\mathbf{P}_{0123}\ \mathbf{P}_{1234}\ \mathbf{P}_{2345} \dots \mathbf{P}_{n-3,n-2,n-1,n}$	$n-2$
$\vdots$	$\vdots$	$\vdots$
n	$\mathbf{P}_{0123\dots n}$	



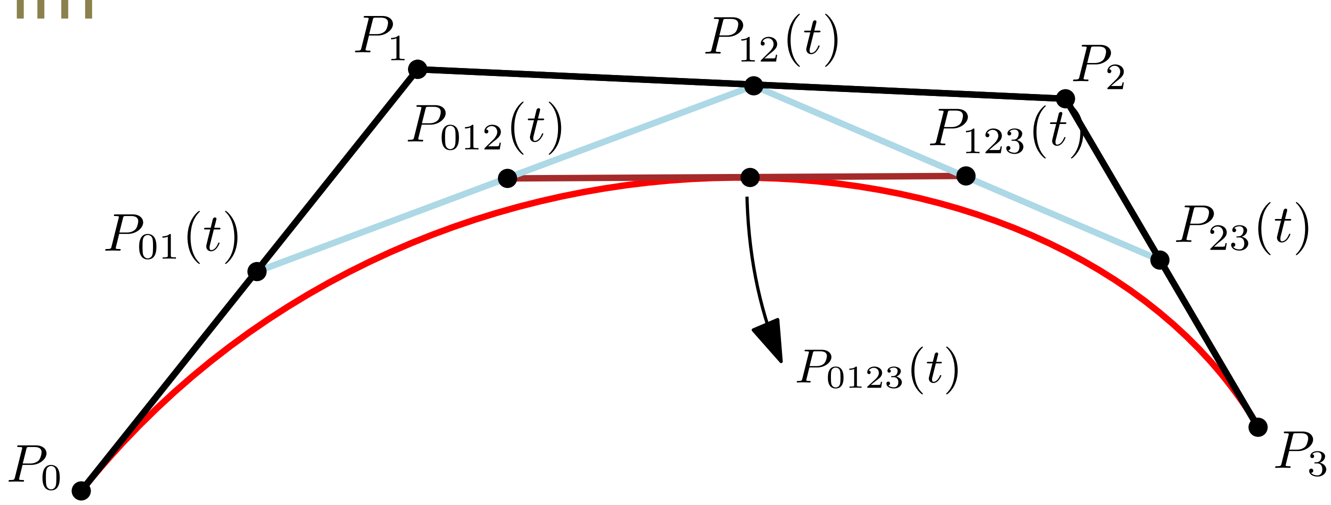
How many points computed in total?

BÉZIER CURVES AS LINEAR INTERPOLATION

De Casteljau's algorithm

Example for $n = 3$ and $t = 1/2$

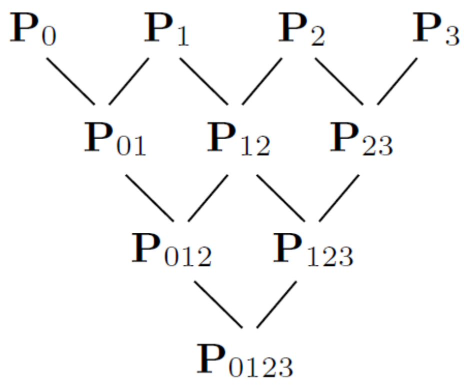
$$P(t) = P_{0123}(t)$$



Implementation of the algorithm

How to evaluate $P(1/2)$?

Step	Points constructed	#points
1	$\mathbf{P}_{01}\ \mathbf{P}_{12}\ \mathbf{P}_{23} \ldots \mathbf{P}_{n-1,n}$	n
2	$\mathbf{P}_{012}\ \mathbf{P}_{123}\ \mathbf{P}_{234} \ldots \mathbf{P}_{n-2,n-1,n}$	$n-1$
3	$\mathbf{P}_{0123}\ \mathbf{P}_{1234}\ \mathbf{P}_{2345} \ldots \mathbf{P}_{n-3,n-2,n-1,n}$	$n-2$
$\vdots$	$\vdots$	$\vdots$
n	$\mathbf{P}_{0123\ldots n}$	



How many points computed in total?

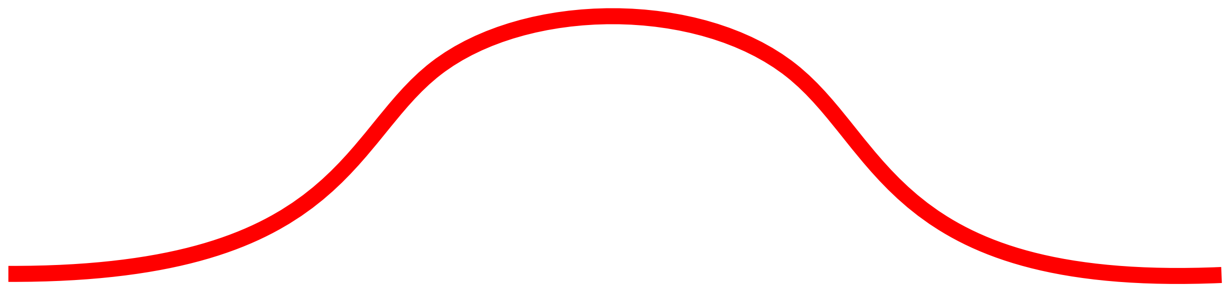
$$n + (n-1) + (n-2) + \cdots + 2 + 1 = n(n+1)/2$$

De Casteljau's algorithm

Note: to generate one point on the curve, $\approx n^2/2$ computations is quite a lot...

De Casteljau's algorithm

Note: to generate one point on the curve, $\approx n^2/2$ computations is quite a lot...



De Casteljau's algorithm

Note: to generate one point on the curve, $\approx n^2/2$ computations is quite a lot...

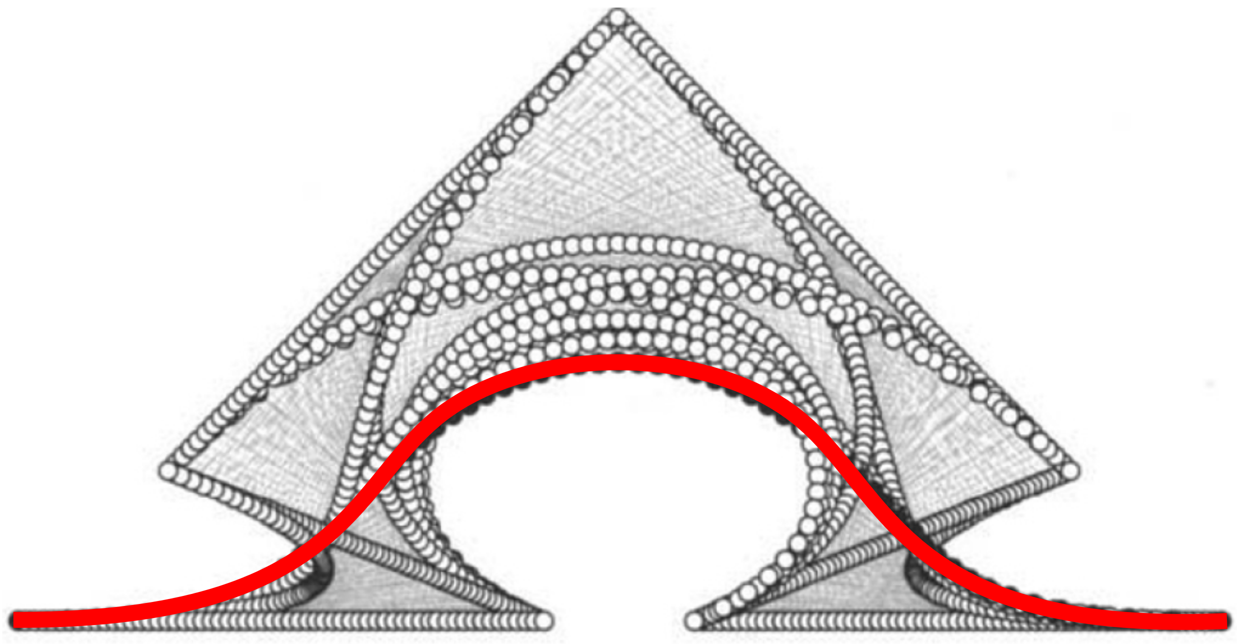


Figure 4.3 The de Casteljau algorithm: 60 points are computed on a degree six curve; all intermedediate points $\mathbf{b}_i^r$ are shown.

Figure from book by Farin (page 47)

De Casteljau's algorithm

Note: to generate one point on the curve, $\approx n^2/2$ computations is quite a lot...

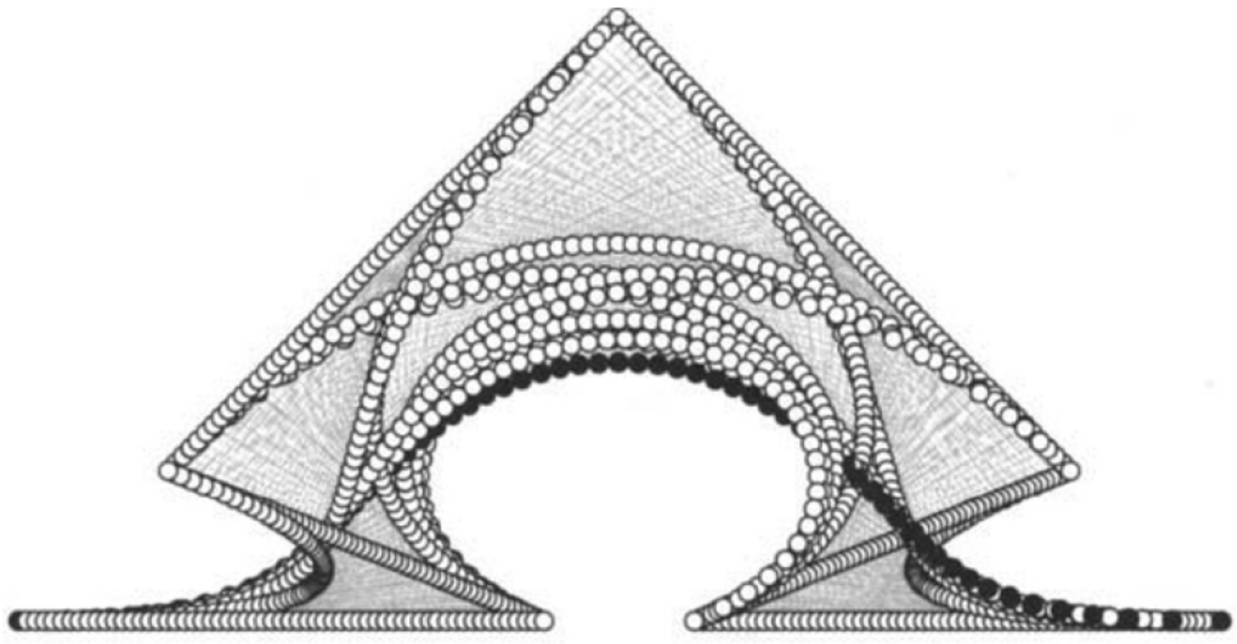


Figure 4.3 The de Casteljau algorithm: 60 points are computed on a degree six curve; all intermediate points $\mathbf{b}_i^r$ are shown.

Figure from book by Farin (page 47)

De Casteljau's algorithm

Note: to generate one point on the curve, $\approx n^2/2$ computations is quite a lot...

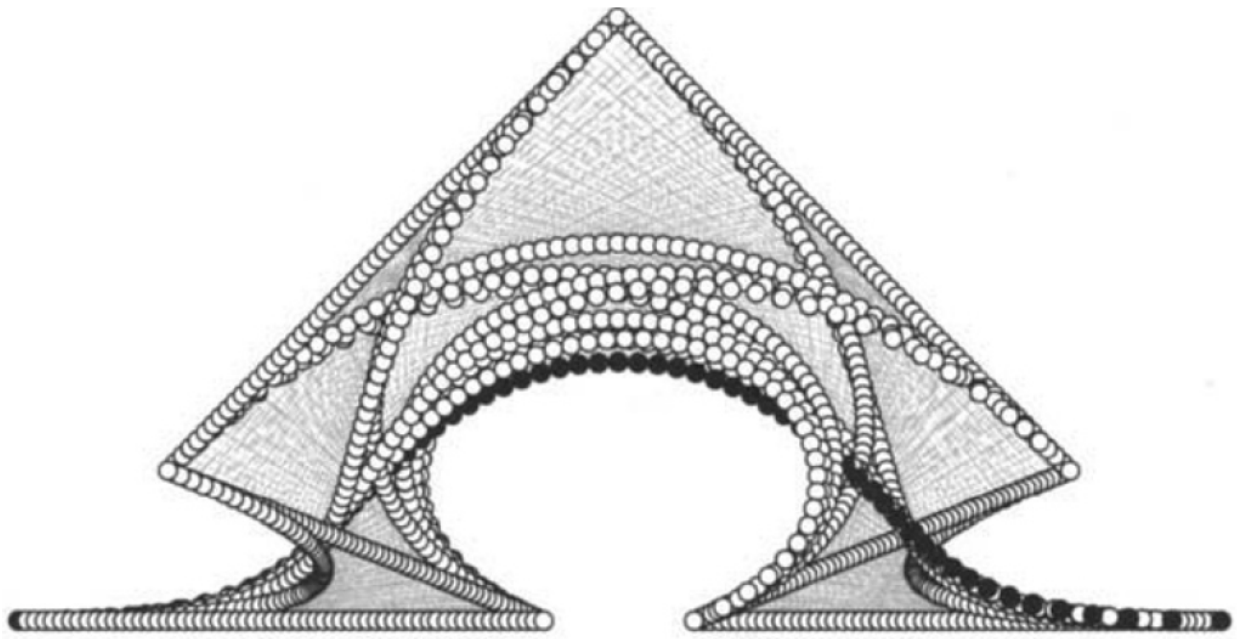


Figure 4.3 The de Casteljau algorithm: 60 points are computed on a degree six curve; all intermede-diate points $\mathbf{b}_i^r$ are shown.

Figure from book by Farin (page 47)

Question for later: Is the computation based on Bernstein polynomials faster?

Using De Casteljau's to subdivide a curve

Using De Casteljau's to subdivide a curve

What if you want to add more points to a curve?
(We need this when we need more flexibility to design the curve)

Goal: increase number of control points, without changing the shape of the curve (at all!)

Using De Casteljau's to subdivide a curve

What if you want to add more points to a curve?
(We need this when we need more flexibility to design the curve)

Goal: increase number of control points, without changing the shape of the curve (at all!)

One way to achieve this: subdivision

Subdivide degree-n curve into two curves, each of degree n

BÉZIER CURVES AS LINEAR INTERPOLATION

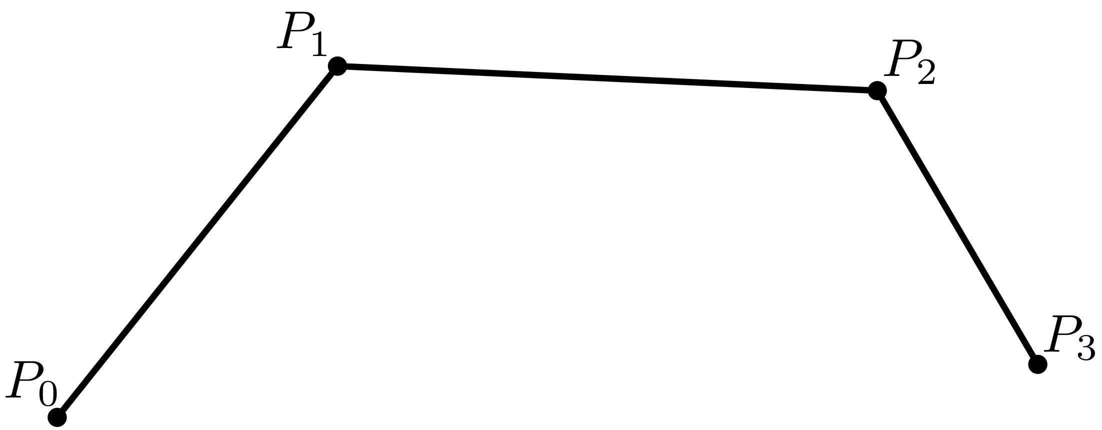
Using De Casteljau's to subdivide a curve

What if you want to add more points to a curve?
(We need this when we need more flexibility to design the curve)

Goal: increase number of control points, without changing the shape of the curve (at all!)

One way to achieve this: subdivision

Subdivide degree-n curve into two curves, each of degree n



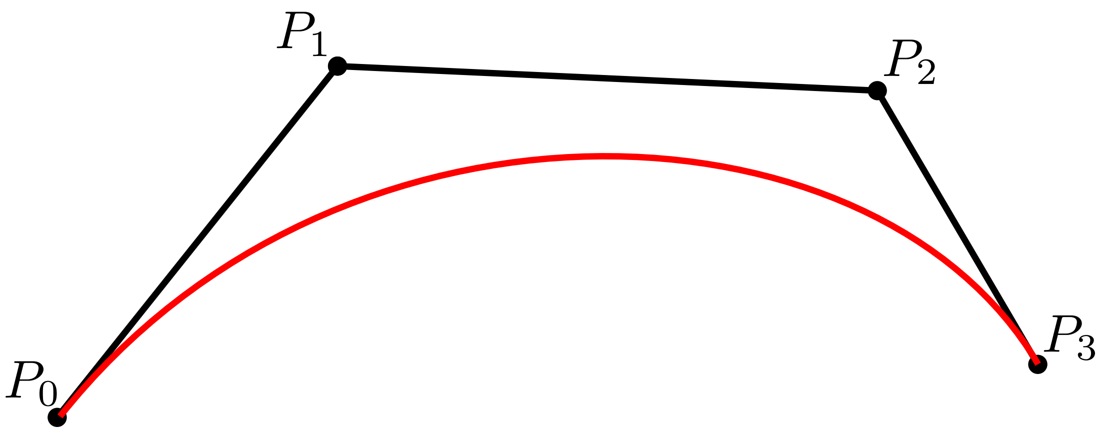
Using De Casteljau's to subdivide a curve

What if you want to add more points to a curve?
(We need this when we need more flexibility to design the curve)

Goal: increase number of control points, without changing the shape of the curve (at all!)

One way to achieve this: subdivision

Subdivide degree-n curve into two curves, each of degree n



BÉZIER CURVES AS LINEAR INTERPOLATION

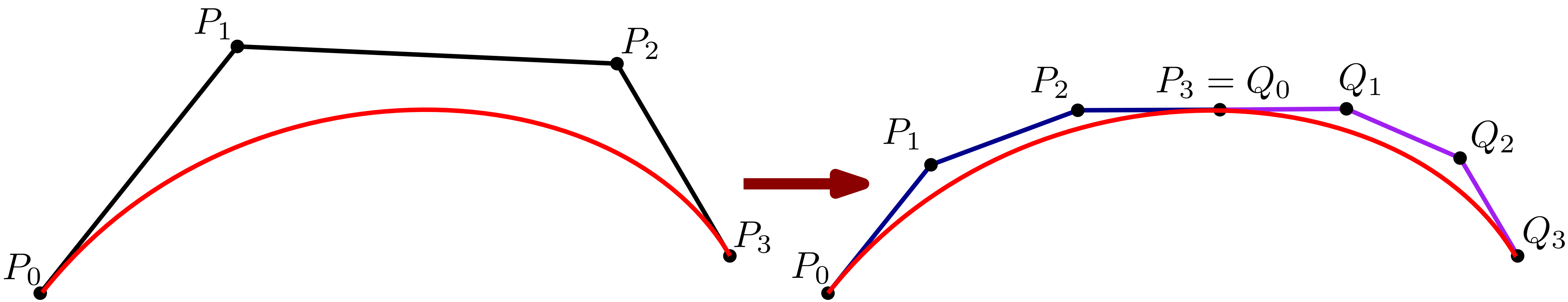
Using De Casteljau's to subdivide a curve

What if you want to add more points to a curve?
(We need this when we need more flexibility to design the curve)

Goal: increase number of control points, without changing the shape of the curve (at all!)

One way to achieve this: subdivision

Subdivide degree-n curve into two curves, each of degree n



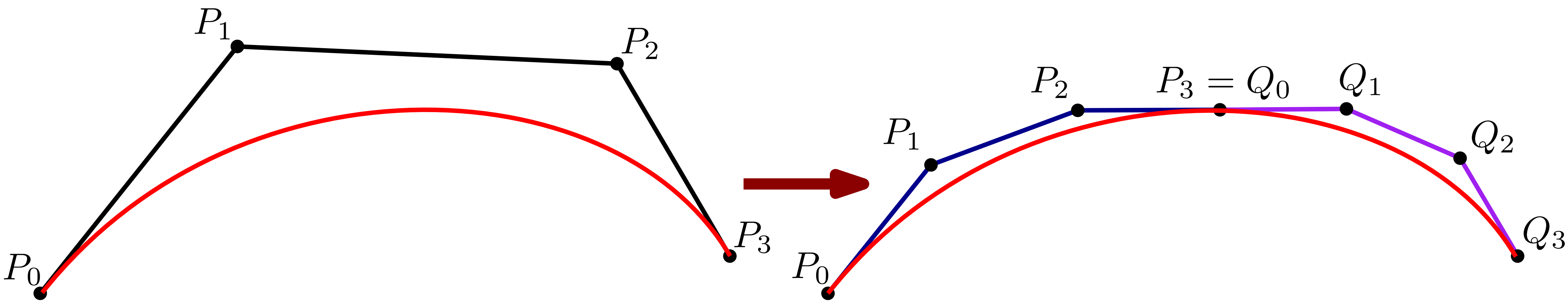
Using De Casteljau's to subdivide a curve

What if you want to add more points to a curve?
(We need this when we need more flexibility to design the curve)

Goal: increase number of control points, without changing the shape of the curve (at all!)

One way to achieve this: subdivision

Subdivide degree-n curve into two curves, each of degree n

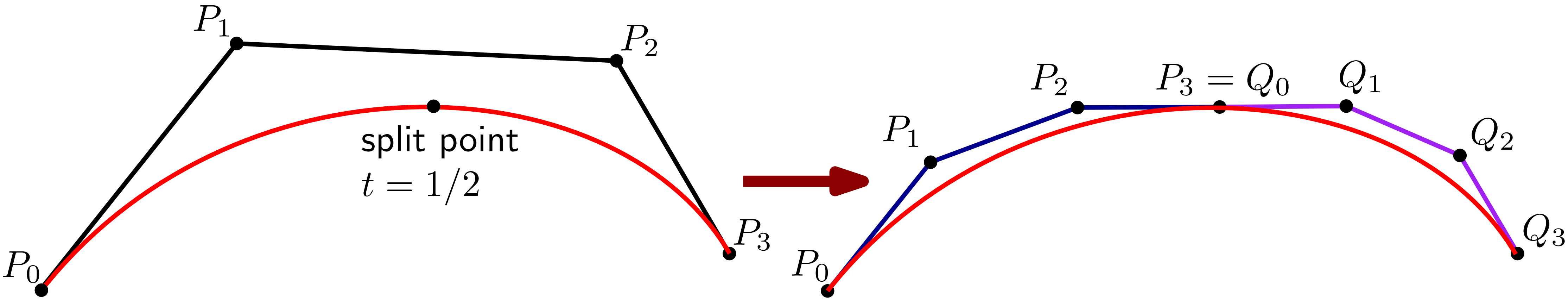


The new points come from the intermediate points of De Casteljau's algorithm!

BÉZIER CURVES AS LINEAR INTERPOLATION

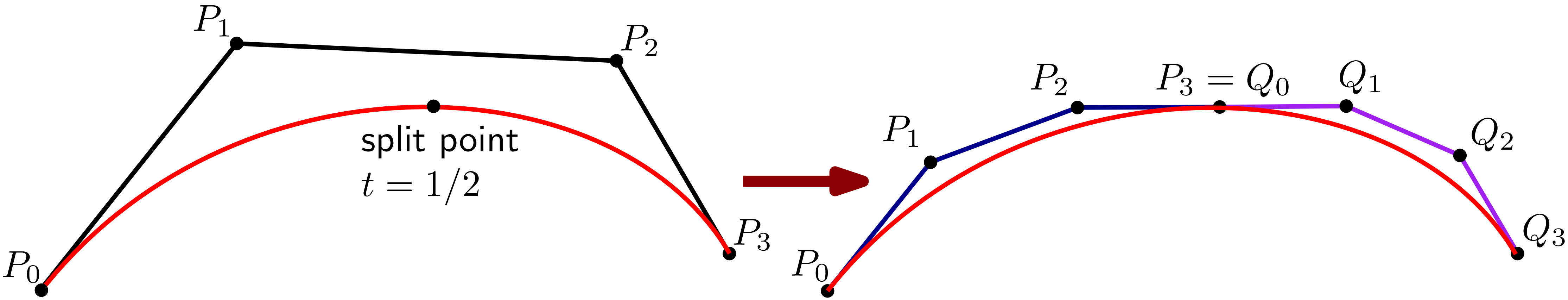
Using De Casteljau's to subdivide a curve

The new points come from the intermediate points of De Casteljau's algorithm!



Using De Casteljau's to subdivide a curve

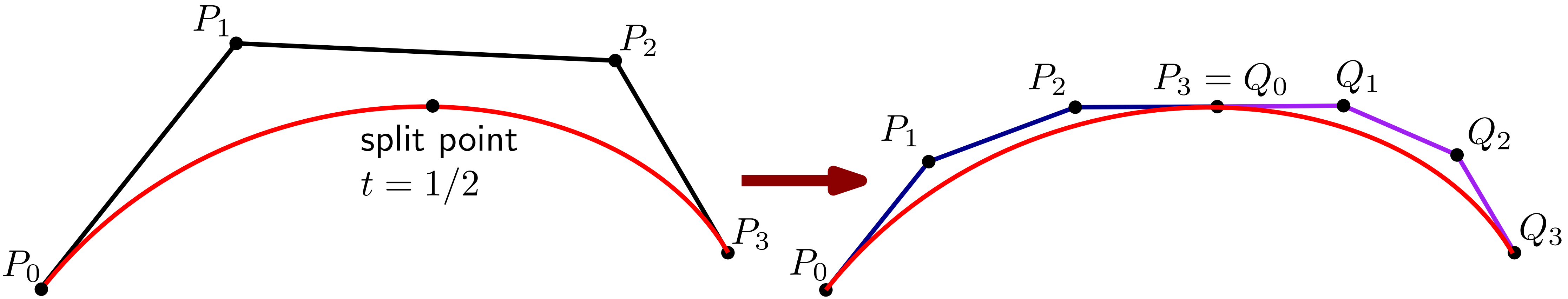
The new points come from the intermediate points of De Casteljau's algorithm!



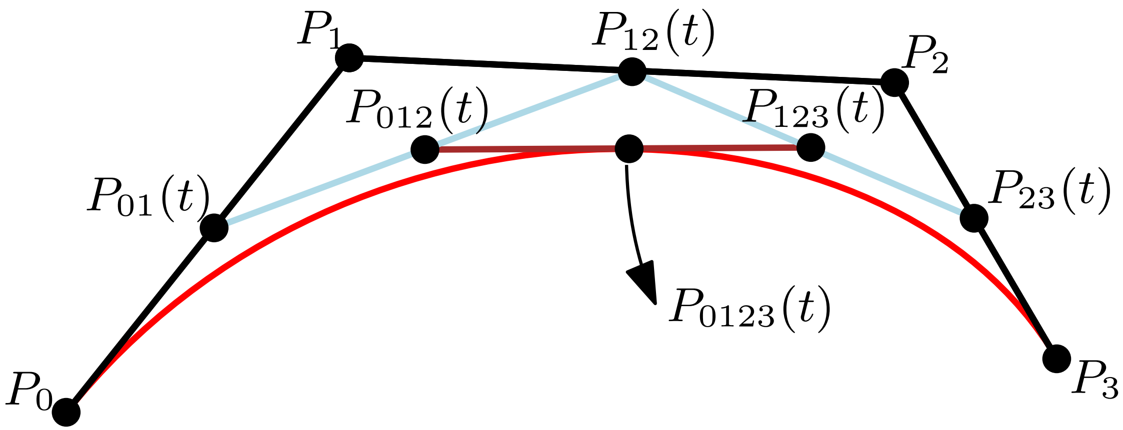
Recall the De Casteljau algorithm ($t = 1/2$):

Using De Casteljau's to subdivide a curve

The new points come from the intermediate points of De Casteljau's algorithm!

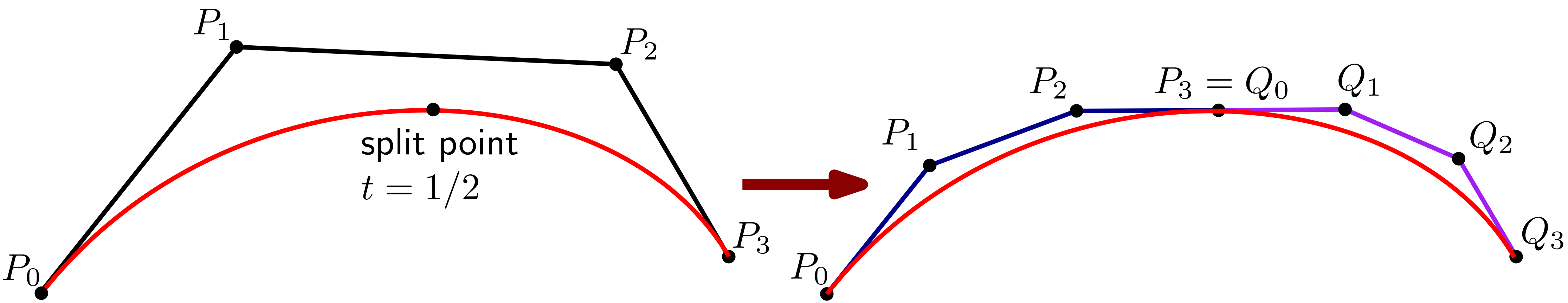


Recall the De Casteljau algorithm ($t = 1/2$):

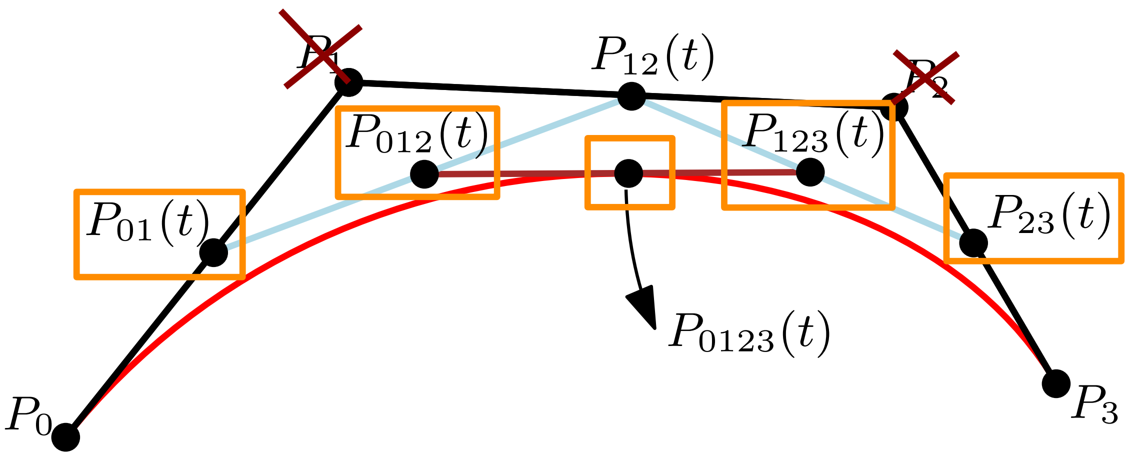


Using De Casteljau's to subdivide a curve

The new points come from the intermediate points of De Casteljau's algorithm!



Recall the De Casteljau algorithm ($t = 1/2$):

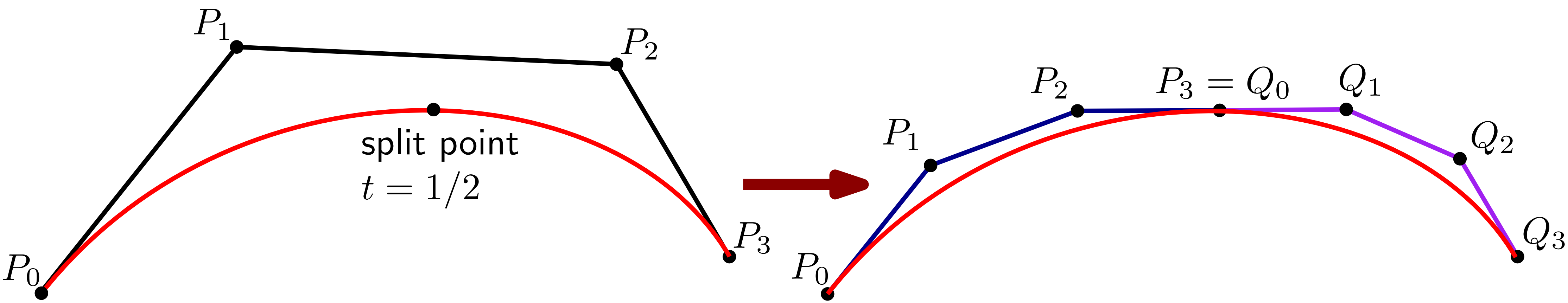


In general, the subdivision is done by:

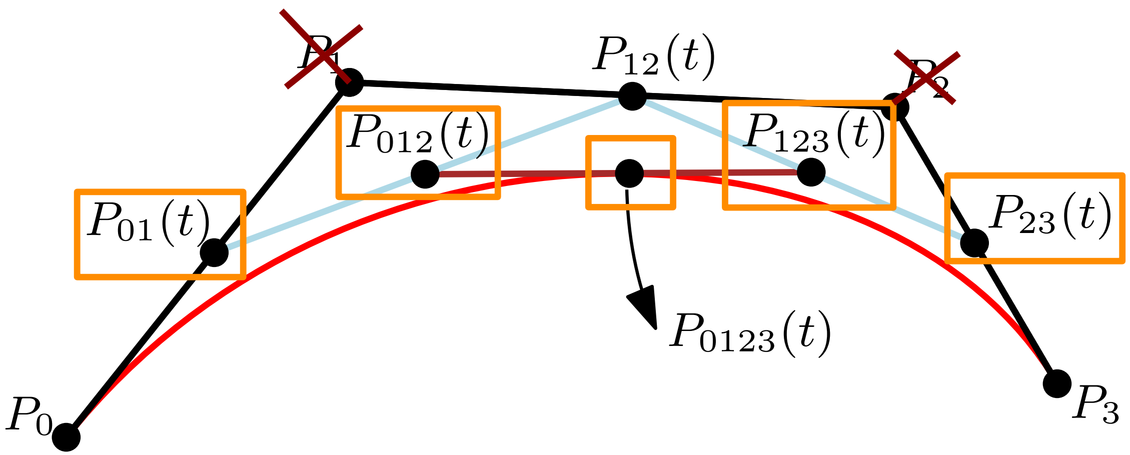
- Discarding interior control points $P_1, \ldots, P_{n-1}$
- Adding $2n-1$ points: the first and last points in each step of De Casteljau's

Using De Casteljau's to subdivide a curve

The new points come from the intermediate points of De Casteljau's algorithm!



Recall the De Casteljau algorithm ($t = 1/2$):



In general, the subdivision is done by:

- Discarding interior control points $P_1, \dots, P_{n-1}$
- Adding $2n-1$ points: the first and last points in each step of De Casteljau's

This method is also useful for **clipping**

Computation of a Bézier curve

Recall definition

$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$

$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$

recall that $0 \leq i \leq n$, $\binom{n}{i} = \frac{n!}{i!(n-i)!}$, and $0! = 1$

Computation of a Bézier curve

Recall definition

$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$

$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$

recall that $0 \leq i \leq n$, $\binom{n}{i} = \frac{n!}{i!(n-i)!}$, and $0! = 1$

Question: how many operations (say, products) needed to compute $P(1/2)$?

Computation of a Bézier curve

Recall definition

$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$

$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$

recall that $0 \leq i \leq n$, $\binom{n}{i} = \frac{n!}{i!(n-i)!}$, and $0! = 1$

Question: how many operations (say, products) needed to compute $P(1/2)$?

Simple speed-up: reuse computed values for different values of t

- Precalculate factorials, store them in table
- Then do the same with binomials
- Precalculate and store values t^i for $i = 0, \ldots, n$ and the necessary t values

Computation of a Bézier curve

Recall definition

$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$

$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$

recall that $0 \le i \le n$, $\binom{n}{i} = \frac{n!}{i!(n-i)!}$, and $0! = 1$

Question: how many operations (say, products) needed to compute $P(1/2)$?

Simple speed-up: reuse computed values for different values of t

- Precalculate factorials, store them in table
- Then do the same with binomials
- Precalculate and store values t^i for $i = 0, \ldots, n$ and the necessary t values

The computation of $P(t)$ becomes
$\sum_{i=0}^{n} \text{Table}_1[i,n] \times \text{Table}_2[t,i] \times \text{Table}_2[1-t, n-i] \times P_i$

Computation of a Bézier curve

Recall definition

$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$

$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$

recall that $0 \leq i \leq n$, $\binom{n}{i} = \frac{n!}{i!(n-i)!}$, and $0! = 1$

Question: how many operations (say, products) needed to compute $P(1/2)$?

Simple speed-up: reuse computed values for different values of t

- Precalculate factorials, store them in table
- Then do the same with binomials
- Precalculate and store values t^i for $i = 0, \ldots, n$ and the necessary t values

The computation of $P(t)$ becomes
$\sum_{i=0}^{n} \text{Table}_1[i,n] \times \text{Table}_2[t,i] \times \text{Table}_2[1-t, n-i] \times P_i$

→ This can also be stored in a table, and reused for other points

Even faster: forward differences

Idea: find a method to "jump" from one point in $P(t)$ to the next one ,$P(t+\Delta)$, using only a few computations for each jump

Even faster: forward differences

step size

Idea: find a method to "jump" from one point in $P(t)$ to the next one ,$P(t+\Delta)$, using only a few computations for each jump

Even faster: forward differences

step size

Idea: find a method to "jump" from one point in $P(t)$ to the next one ,$P(t+\Delta)$, using only a few computations for each jump

Goal: find quantity dP such that $P(t+\Delta) = P(t) + dP$

If dP would exist, then we could do:

$$\begin{aligned} P(0) &= P_0 \\ P(0+\Delta) &= P(0) + dP = P_0 + dP \\ P(2\Delta) &= P(\Delta) + dP = P_0 + 2dP \\ P(i \cdot \Delta) &= P((i-1)\Delta) + dP = P_0 + i \cdot dP \end{aligned}$$

Even faster: forward differences

step size

Idea: find a method to “jump” from one point in $P(t)$ to the next one ,$P(t+\Delta)$, using only a few computations for each jump

Goal: find quantity dP such that $P(t+\Delta) = P(t) + dP$

If dP would exist, then we could do:

$$\begin{aligned} P(0) &= P_0 \\ P(0+\Delta) &= P(0) + dP = P_0 + dP \\ P(2\Delta) &= P(\Delta) + dP = P_0 + 2dP \\ P(i\cdot\Delta) &= P((i-1)\Delta) + dP = P_0 + i\cdot dP \end{aligned}$$

This would be very efficient!

Even faster: forward differences

step size

Idea: find a method to "jump" from one point in $P(t)$ to the next one ,$P(t+\Delta)$, using only a few computations for each jump

Goal: find quantity dP such that $P(t+\Delta) = P(t) + dP$

If dP would exist, then we could do:

$$\begin{aligned} P(0) &= P_0 \\ P(0+\Delta) &= P(0) + dP = P_0 + dP \\ P(2\Delta) &= P(\Delta) + dP = P_0 + 2dP \\ P(i\cdot\Delta) &= P((i-1)\Delta) + dP = P_0 + i\cdot dP \end{aligned}$$

This would be very efficient!

Consider the *Taylor series* representation of $P(t)$:

$$P(t+\Delta) = P(t) + P'(t)\Delta + P''(t)\frac{\Delta^2}{2!} + P'''(t)\frac{\Delta^3}{3!} + P''''(t)\frac{\Delta^4}{4!} + \ldots$$

Even faster: forward differences

step size

Idea: find a method to "jump" from one point in $P(t)$ to the next one ,$P(t+\Delta)$, using only a few computations for each jump

Goal: find quantity dP such that $P(t+\Delta) = P(t) + dP$

If dP would exist, then we could do:

$$\begin{aligned} P(0) &= P_0 \\ P(0+\Delta) &= P(0) + dP = P_0 + dP \\ P(2\Delta) &= P(\Delta) + dP = P_0 + 2dP \\ P(i\cdot\Delta) &= P((i-1)\Delta) + dP = P_0 + i\cdot dP \end{aligned}$$

This would be very efficient!

Consider the *Taylor series* representation of $P(t)$:

$$P(t+\Delta) = P(t) + P'(t)\Delta + P''(t)\frac{\Delta^2}{2!} + P'''(t)\frac{\Delta^3}{3!} + P''''(t)\frac{\Delta^4}{4!} + ...$$

Infinite series

But becomes finite if $P(t)$ has constant degree!

Forward differences for cubic Bézier curve

$P(t+\Delta) = P(t) + P'(t)\Delta + P''(t)\frac{\Delta^2}{2} + P'''(t)\frac{\Delta^3}{6}$

Forward differences for cubic Bézier curve

$P(t+\Delta) = P(t) + P'(t)\Delta + P''(t)\frac{\Delta^2}{2} + P'''(t)\frac{\Delta^3}{6}$

For a cubic Bézier curve, we have

$P(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t)P_2 + t^3 P_3$
or, equivalently,
$P(t) = at^3 + bt^2 + ct + d$
where:
$a = 3(P_1 - P_2) - P_0 + P_3$, $b = 3(P_0 + P_2) - 6P_1$, $c = 3(P_1 - P_0)$, $d = P_0$

Forward differences for cubic Bézier curve

$P(t+\Delta) = P(t) + P'(t)\Delta + P''(t)\frac{\Delta^2}{2} + P'''(t)\frac{\Delta^3}{6}$

For a cubic Bézier curve, we have

$P(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t)P_2 + t^3 P_3$
or, equivalently,
$P(t) = at^3 + bt^2 + ct + d$
where:
$a = 3(P_1 - P_2) - P_0 + P_3$, $b = 3(P_0 + P_2) - 6P_1$, $c = 3(P_1 - P_0)$, $d = P_0$

Now it is easy to differentiate:

$P'(t) = 3at^2 + 2bt + c$ $\quad P''(t) = 6at + 2b$ $\quad P'''(t) = 6a$

Forward differences for cubic Bézier curve

$P(t+\Delta) = P(t) + P'(t)\Delta + P''(t)\frac{\Delta^2}{2} + P'''(t)\frac{\Delta^3}{6}$

For a cubic Bézier curve, we have

$P(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t)P_2 + t^3 P_3$

or, equivalently,

$P(t) = at^3 + bt^2 + ct + d$

where:

$a = 3(P_1 - P_2) - P_0 + P_3$, $b = 3(P_0 + P_2) - 6P_1$, $c = 3(P_1 - P_0)$, $d = P_0$

Now it is easy to differentiate:

$P'(t) = 3at^2 + 2bt + c$ $\qquad P''(t) = 6at + 2b$ $\qquad P'''(t) = 6a$

Recall, goal: find quantity dP such that $P(t+\Delta) = P(t) + dP \iff dP = P(t+\Delta) - P(t)$

Forward differences for cubic Bézier curve

$P(t+\Delta) = P(t) + P'(t)\Delta + P''(t)\frac{\Delta^2}{2} + P'''(t)\frac{\Delta^3}{6}$

we can derive $P(t+\Delta) - P(t)$ from here

For a cubic Bézier curve, we have

$P(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t)P_2 + t^3 P_3$

or, equivalently,

$P(t) = at^3 + bt^2 + ct + d$

where:

$a = 3(P_1 - P_2) - P_0 + P_3$, $b = 3(P_0 + P_2) - 6P_1$, $c = 3(P_1 - P_0)$, $d = P_0$

Now it is easy to differentiate:

$P'(t) = 3at^2 + 2bt + c \qquad P''(t) = 6at + 2b \qquad P'''(t) = 6a$

Recall, goal: find quantity dP such that $P(t+\Delta) = P(t) + dP \iff dP = \boxed{P(t+\Delta) - P(t)}$

Forward differences for cubic Bézier curve

$P(t+\Delta) = P(t) + P'(t)\Delta + P''(t)\frac{\Delta^2}{2} + P'''(t)\frac{\Delta^3}{6}$

we can derive $P(t+\Delta) - P(t)$ from here

For a cubic Bézier curve, we have

$P(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t)P_2 + t^3 P_3$

or, equivalently,

$P(t) = at^3 + bt^2 + ct + d$

where:

$a = 3(P_1 - P_2) - P_0 + P_3$, $b = 3(P_0 + P_2) - 6P_1$, $c = 3(P_1 - P_0)$, $d = P_0$

Now it is easy to differentiate:

$P'(t) = 3at^2 + 2bt + c$ $\quad$ $P''(t) = 6at + 2b$ $\quad$ $P'''(t) = 6a$

Recall, goal: find quantity dP such that $P(t+\Delta) = P(t) + dP \iff dP = P(t+\Delta) - P(t)$

$$dP(t) = P(t+\Delta t) - P(t) = (3at^2\Delta + 2bt\Delta + c\Delta) + (3at\Delta^2 + b\Delta^2) + (a\Delta^3)$$

Forward differences for cubic Bézier curve

$P(t+\Delta) = P(t) + P'(t)\Delta + P''(t)\frac{\Delta^2}{2} + P'''(t)\frac{\Delta^3}{6}$

we can derive $P(t+\Delta) - P(t)$ from here

For a cubic Bézier curve, we have

$P(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t)P_2 + t^3 P_3$

or, equivalently,

$P(t) = at^3 + bt^2 + ct + d$

where:

$a = 3(P_1 - P_2) - P_0 + P_3$, $b = 3(P_0 + P_2) - 6P_1$, $c = 3(P_1 - P_0)$, $d = P_0$

Now it is easy to differentiate:

$P'(t) = 3at^2 + 2bt + c \qquad P''(t) = 6at + 2b \qquad P'''(t) = 6a$

Recall, goal: find quantity dP such that $P(t+\Delta) = P(t) + dP \iff dP = \boxed{P(t+\Delta) - P(t)}$

$$\boxed{dP(t) = P(t+\Delta t) - P(t) = (3at^2\Delta + 2bt\Delta + c\Delta) + (3at\Delta^2 + b\Delta^2) + (a\Delta^3)}$$

Problem: dP depends on t!

Forward differences for cubic Bézier curve

$$dP(t) = 3at^2\Delta + 2bt\Delta + c\Delta + 3at\Delta^2 + b\Delta^2 + a\Delta^3$$

Forward differences for cubic Bézier curve

$dP(t) = 3at^2\Delta + 2bt\Delta + c\Delta + 3at\Delta^2 + b\Delta^2 + a\Delta^3$ degree-2 polynomial on t

Forward differences for cubic Bézier curve

$dP(t) = 3at^2\Delta + 2bt\Delta + c\Delta + 3at\Delta^2 + b\Delta^2 + a\Delta^3$ degree-2 polynomial on t

- Idea: use the same technique again, to get polynomial of degree 1...

  ...then do it again, to obtain polynomial of degree 0, i.e., constant!

Forward differences for cubic Bézier curve

$dP(t) = 3at^2\Delta + 2bt\Delta + c\Delta + 3at\Delta^2 + b\Delta^2 + a\Delta^3$ degree-2 polynomial on t

- Idea: use the same technique again, to get polynomial of degree 1...

  ...then do it again, to obtain polynomial of degree 0, i.e., constant!

$ddP(t) = dP'(t)\Delta + \frac{dP''(t)\Delta^2}{2}$

Forward differences for cubic Bézier curve

$dP(t) = 3at^2\Delta + 2bt\Delta + c\Delta + 3at\Delta^2 + b\Delta^2 + a\Delta^3$ degree-2 polynomial on t

- Idea: use the same technique again, to get polynomial of degree 1...

  ...then do it again, to obtain polynomial of degree 0, i.e., constant!

$ddP(t) = dP'(t)\Delta + \frac{dP''(t)\Delta^2}{2}$

$ddP(t) = (6at\Delta + 2b\Delta + 3a\Delta^2)\Delta + \frac{6a\Delta\Delta^2}{2} = 6at\Delta^2 + 2b\Delta^2 + 6a\Delta^3$

Forward differences for cubic Bézier curve

$dP(t) = 3at^2\Delta + 2bt\Delta + c\Delta + 3at\Delta^2 + b\Delta^2 + a\Delta^3$ degree-2 polynomial on t

- Idea: use the same technique again, to get polynomial of degree 1...

  ...then do it again, to obtain polynomial of degree 0, i.e., constant!

$$ddP(t) = dP'(t)\Delta + \frac{dP''(t)\Delta^2}{2}$$

$$ddP(t) = (6at\Delta + 2b\Delta + 3a\Delta^2)\Delta + \frac{6a\Delta\Delta^2}{2} = 6at\Delta^2 + 2b\Delta^2 + 6a\Delta^3$$

degree-1 polynomial on t

Forward differences for cubic Bézier curve

$dP(t) = 3at^2\Delta + 2bt\Delta + c\Delta + 3at\Delta^2 + b\Delta^2 + a\Delta^3$ degree-2 polynomial on t

- Idea: use the same technique again, to get polynomial of degree 1...

  ...then do it again, to obtain polynomial of degree 0, i.e., constant!

$ddP(t) = dP'(t)\Delta + \frac{dP''(t)\Delta^2}{2}$

$ddP(t) = (6at\Delta + 2b\Delta + 3a\Delta^2)\Delta + \frac{6a\Delta\Delta^2}{2} = 6at\Delta^2 + 2b\Delta^2 + 6a\Delta^3$

degree-1 polynomial on t

- One more time: let's compute $dddP(t)$

$dddP(t) = ddP'(t)\Delta = 6a\Delta^3$

Forward differences for cubic Bézier curve

$dP(t) = 3at^2\Delta + 2bt\Delta + c\Delta + 3at\Delta^2 + b\Delta^2 + a\Delta^3$ degree-2 polynomial on t

- Idea: use the same technique again, to get polynomial of degree 1...

  ...then do it again, to obtain polynomial of degree 0, i.e., constant!

$ddP(t) = dP'(t)\Delta + \frac{dP''(t)\Delta^2}{2}$

$ddP(t) = (6at\Delta + 2b\Delta + 3a\Delta^2)\Delta + \frac{6a\Delta\Delta^2}{2} = 6at\Delta^2 + 2b\Delta^2 + 6a\Delta^3$

degree-1 polynomial on t

- One more time: let's compute $dddP(t)$

$dddP(t) = ddP'(t)\Delta = 6a\Delta^3$ $\rightarrow$ $dddP$ is a constant! (does not depend on t)

We combine the three increments (dP, ddP and $dddP$) in an algorithms as follows:

Forward differences for cubic Bézier curve

$dP(t) = 3at^2\Delta + 2bt\Delta + c\Delta + 3at\Delta^2 + b\Delta^2 + a\Delta^3$ degree-2 polynomial on t

- Idea: use the same technique again, to get polynomial of degree 1...

  ...then do it again, to obtain polynomial of degree 0, i.e., constant!

$$ddP(t) = dP'(t)\Delta + \frac{dP''(t)\Delta^2}{2}$$

$$ddP(t) = (6at\Delta + 2b\Delta + 3a\Delta^2)\Delta + \frac{6a\Delta\Delta^2}{2} = 6at\Delta^2 + 2b\Delta^2 + 6a\Delta^3$$

degree-1 polynomial on t

- One more time: let's compute $dddP(t)$

$dddP(t) = ddP'(t)\Delta = 6a\Delta^3$ $\rightarrow$ $dddP$ is a constant! (does not depend on t)

We combine the three increments (dP, ddP and $dddP$) in an algorithms as follows:

```
procedure FastCubicBézierSketch
    Compute dP, ddP and dddP for t=0
    P ← P_0
    for t = 0 to 1 step Δ do
        Draw point P
        P ← P + dP
        dP ← dP + ddP
        ddP ← ddP + dddP
```

Forward differences for cubic Bézier curve

Final code, trying to reuse computations as much as possible

procedure FastCubicBézier
    $Q_1 \leftarrow 3\Delta$
    $Q_2 \leftarrow Q_1 \cdot \Delta$ $\triangleright\ 3\Delta^2$
    $Q_3 \leftarrow \Delta^3$
    $Q_4 \leftarrow 2Q_2$ $\triangleright\ 6\Delta^2$
    $Q_5 \leftarrow 6Q_3$ $\triangleright\ 6\Delta^3$
    $Q_6 \leftarrow P_0 - 2P_1 + P_2$
    $Q_7 \leftarrow 3(P_1 - P_2) - P_0 + P_3$ $\triangleright\ a$
    $P \leftarrow P_0$
    $dP \leftarrow (P_1 - P_0)Q_1 + Q_6 \cdot Q_2 + Q_7 \cdot Q_3$
    $ddP \leftarrow Q_6 \cdot Q_4 + Q_7 \cdot Q_5$
    $dddP \leftarrow Q_7 \cdot Q_5$
    for $t = 0$ to 1 step Δ **do**
        Draw point P
        $P \leftarrow P + dP$
        $dP \leftarrow dP + ddP$
        $ddP \leftarrow ddP + dddP$

Forward differences for cubic Bézier curve

Final code, trying to reuse computations as much as possible

procedure FASTCUBICBÉZIER

$Q_1 \leftarrow 3\Delta$

$Q_2 \leftarrow Q_1 \cdot \Delta$ $\triangleright$ $3\Delta^2$

$Q_3 \leftarrow \Delta^3$

$Q_4 \leftarrow 2Q_2$ $\triangleright$ $6\Delta^2$

$Q_5 \leftarrow 6Q_3$ $\triangleright$ $6\Delta^3$

$Q_6 \leftarrow P_0 - 2P_1 + P_2$

$Q_7 \leftarrow 3(P_1 - P_2) - P_0 + P_3$ $\triangleright$ a

$P \leftarrow P_0$

$dP \leftarrow (P_1 - P_0)Q_1 + Q_6 \cdot Q_2 + Q_7 \cdot Q_3$

$ddP \leftarrow Q_6 \cdot Q_4 + Q_7 \cdot Q_5$

$dddP \leftarrow Q_7 \cdot Q_5$

for $t = 0$ to 1 step Δ **do**

  Draw point P

  $P \leftarrow P + dP$

  $dP \leftarrow dP + ddP$

  $ddP \leftarrow ddP + dddP$

The reduction in # of operations is huge: Ignoring the initialization, 3 sums for each evaluation of t

Matrix formulation

Bézier curves are often expressed in matrix form

Matrix formulation

Bézier curves are often expressed in matrix form

For $n = 2$, we had $P(t) = (1-t)^2 P_0 + 2t(1-t)P_1 + t^2 P_2$

Matrix formulation

Bézier curves are often expressed in matrix form

$$\text{For } n = 2\text{, we had } P(t) = (1-t)^2 P_0 + 2t(1-t)P_1 + t^2 P_2$$
$$= ((1-t)^2, 2t(1-t), t^2)(P_0, P_1, P_2)^t$$

Matrix formulation

Bézier curves are often expressed in matrix form

For $n = 2$, we had $P(t) = (1-t)^2 P_0 + 2t(1-t)P_1 + t^2 P_2$

$$= ((1-t)^2, 2t(1-t), t^2)(P_0, P_1, P_2)^t$$

$$= (t^2, t, 1) \begin{pmatrix} & & \\ & ? & \\ & & \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \end{pmatrix}$$

Question: what is the matrix?

Matrix formulation

Bézier curves are often expressed in matrix form

For $n = 2$, we had $P(t) = (1-t)^2 P_0 + 2t(1-t)P_1 + t^2 P_2$

$$= ((1-t)^2, 2t(1-t), t^2)(P_0, P_1, P_2)^t$$

$$= (t^2, t, 1) \begin{pmatrix} & & \\ & ? & \\ & & \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \end{pmatrix}$$

Question: what is the matrix?

For n=3, we had $P(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t)P_2 + t^3 P_3$

Matrix formulation

Bézier curves are often expressed in matrix form

For $n = 2$, we had $P(t) = (1-t)^2 P_0 + 2t(1-t)P_1 + t^2 P_2$

$$= ((1-t)^2, 2t(1-t), t^2)(P_0, P_1, P_2)^t$$

$$= (t^2, t, 1) \begin{pmatrix} & & \\ & ? & \\ & & \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \end{pmatrix}$$

Question: what is the matrix?

For n=3, we had $P(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t)P_2 + t^3 P_3$

$$= (t^3, t^2, t, 1) \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix}$$

Matrix formulation

Bézier curves are often expressed in matrix form

For $n = 2$, we had $P(t) = (1-t)^2 P_0 + 2t(1-t)P_1 + t^2 P_2$

$$= ((1-t)^2, 2t(1-t), t^2)(P_0, P_1, P_2)^t$$

$$= (t^2, t, 1) \begin{pmatrix} & & \\ & ? & \\ & & \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \end{pmatrix}$$

Question: what is the matrix?

For n=3, we had $P(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t)P_2 + t^3 P_3$

$$= (t^3, t^2, t, 1) \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix}$$

Note: matrix notation is very practical, but not necessarily very efficient, or numerically stable

Matrix formulation

Bézier curves are often expressed in matrix form

For $n = 2$, we had $P(t) = (1-t)^2 P_0 + 2t(1-t)P_1 + t^2 P_2$

$$= ((1-t)^2, 2t(1-t), t^2)(P_0, P_1, P_2)^t$$

$$= (t^2, t, 1)\begin{pmatrix} & & \\ & ? & \\ & & \end{pmatrix}\begin{pmatrix} P_0 \\ P_1 \\ P_2 \end{pmatrix}$$

Question: what is the matrix?

For n=3, we had $P(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t)P_2 + t^3 P_3$

$$= (t^3, t^2, t, 1)\begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}\begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix}$$

Note: matrix notation is very practical, but not necessarily very efficient, or numerically stable

Question: how many sums/products to evaluate $P(t)$?

Another way to increase number of points

- Recall: curve subdivision took a degree-n curve and produced two curves of degree-n ($2n + 1$ control points in total)
- Alternative: add points (increase degree) while preserving curve

Another way to increase number of points

- Recall: curve subdivision took a degree-n curve and produced two curves of degree-n ($2n + 1$ control points in total)
- Alternative: add points (increase degree) while preserving curve

Why? More control points make it easier to edit, so its shape can be better adjusted

Another way to increase number of points

- Recall: curve subdivision took a degree-n curve and produced two curves of degree-n ($2n + 1$ control points in total)
- Alternative: add points (increase degree) while preserving curve

Why? More control points make it easier to edit, so its shape can be better adjusted

Degree elevation: given a degree-n curve $P_n(t)$, **add one control point** to produce an **identical** curve $P_{n+1}(t)$

Another way to increase number of points

- Recall: curve subdivision took a degree-n curve and produced two curves of degree-n ($2n+1$ control points in total)
- Alternative: add points (increase degree) while preserving curve

Why? More control points make it easier to edit, so its shape can be better adjusted

Degree elevation: given a degree-n curve $P_n(t)$, **add one control point** to produce an **identical** curve $P_{n+1}(t)$

original curve		new curve
$P_n(t)$	→	$P_{n+1}(t)$
P: $n+1$ control points		Q: $n+2$ new control points

Another way to increase number of points

- Recall: curve subdivision took a degree-n curve and produced two curves of degree-n ($2n+1$ control points in total)
- Alternative: add points (increase degree) while preserving curve

Why? More control points make it easier to edit, so its shape can be better adjusted

Degree elevation: given a degree-n curve $P_n(t)$, **add one control point** to produce an **identical** curve $P_{n+1}(t)$

original curve

$P_n(t)$

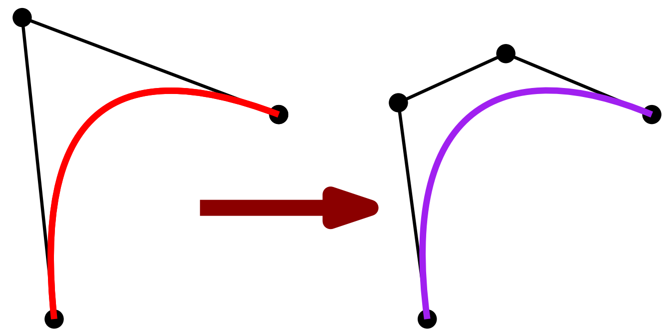
P: $n+1$ control points

→

new curve

$P_{n+1}(t)$

Q: $n+2$ new control points



Adding one more point

original curve

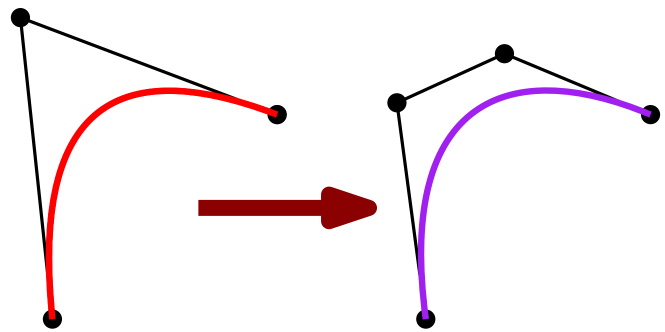
$P_n(t)$

P: $n+1$ control points

new curve

$P_{n+1}(t)$

Q: $n+2$ new control points



Adding one more point

original curve

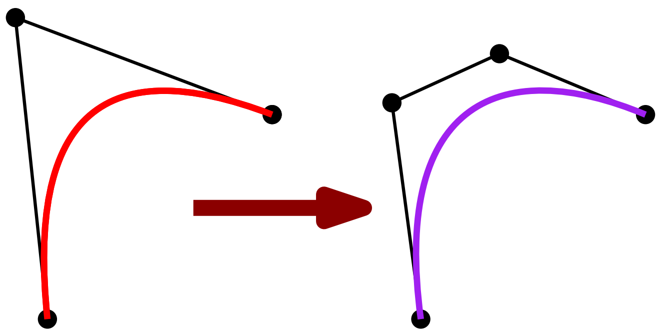
$P_n(t)$

P: $n+1$ control points

new curve

$P_{n+1}(t)$

Q: $n+2$ new control points



Producing control points for $P_{n+1}(t)$

Adding one more point

original curve

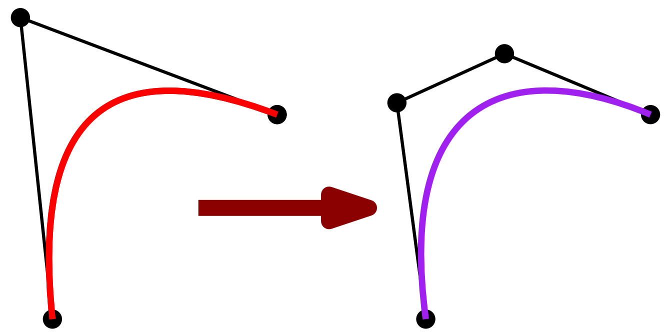
$P_n(t)$

P: $n+1$ control points

new curve

$P_{n+1}(t)$

Q: $n+2$ new control points



Producing control points for $P_{n+1}(t)$

With some basic algebraic tricks one can write $P(t)$ as an $(n+1)$-degree Bézier curve

DEGREE ELEVATION

Adding one more point

original curve

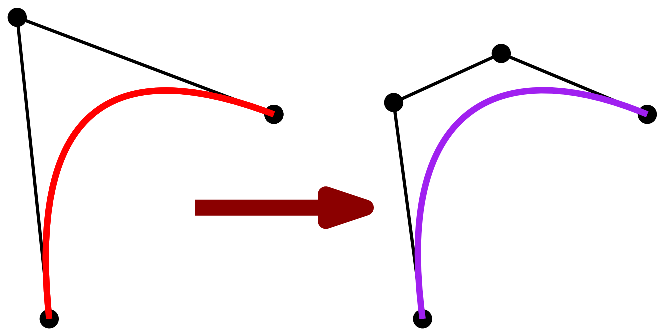
$P_n(t)$

P: $n+1$ control points

new curve

$P_{n+1}(t)$

Q: $n+2$ new control points



Producing control points for $P_{n+1}(t)$

> With some basic algebraic tricks one can
> write $P(t)$ as an $(n+1)$-degree Bézier curve

Start from trivial identity $P(t) = (t + (1-t))P(t) = tP(t) + (1-t)P(t)$

Use that $P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t) = \sum_{i=0}^{n} \binom{n}{i} t^i (1-t)^{n-i} P_i$, and extract coefficientes of new $(n+2)$ control points

Adding one more point

original curve

$P_n(t)$

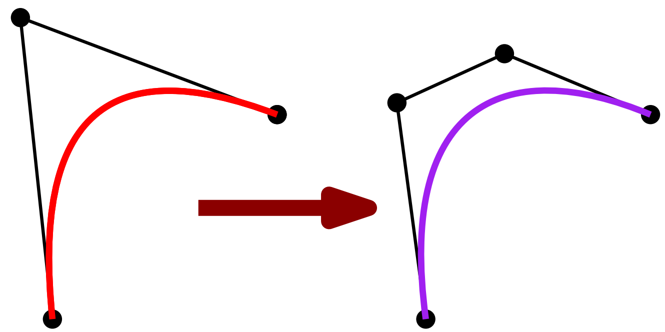
P: $n+1$ control points

→

new curve

$P_{n+1}(t)$

Q: $n+2$ new control points



Producing control points for $P_{n+1}(t)$

With some basic algebraic tricks one can write $P(t)$ as an $(n+1)$-degree Bézier curve

Start from trivial identity $P(t) = (t + (1-t))P(t) = tP(t) + (1-t)P(t)$

Use that $P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t) = \sum_{i=0}^{n} \binom{n}{i} t^i (1-t)^{n-i} P_i$, and extract coefficientes of new $(n+2)$ control points

Result:

$$P(t) = tP(t) + (1-t)P(t) = \sum_{i=0}^{n+1} \binom{n+1}{i} t^i (1-t)^{n-i+1} \left(\frac{i}{n+1} P_{i-1} + \left(1 - \frac{i}{n+1}\right) P_i\right)$$

DEGREE ELEVATION

Adding one more point

original curve

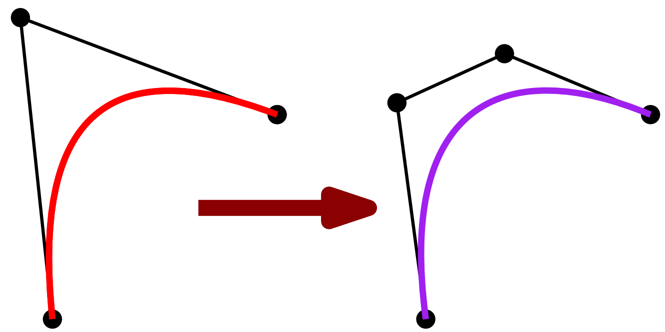
$P_n(t)$

P: $n+1$ control points

new curve

$P_{n+1}(t)$

Q: $n+2$ new control points



Producing control points for $P_{n+1}(t)$

With some basic algebraic tricks one can write $P(t)$ as an $(n+1)$-degree Bézier curve

Start from trivial identity $P(t) = (t + (1-t))P(t) = tP(t) + (1-t)P(t)$

Use that $P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t) = \sum_{i=0}^{n} \binom{n}{i} t^i (1-t)^{n-i} P_i$, and extract coefficientes of new $(n+2)$ control points

Result:

$$P(t) = tP(t) + (1-t)P(t) = \sum_{i=0}^{n+1} \binom{n+1}{i} t^i (1-t)^{n-i+1} \left(\frac{i}{n+1} P_{i-1} + \left(1 - \frac{i}{n+1}\right) P_i \right)$$

Bézier curve of degree $(n+1)$!

$Q_i = \alpha_i P_{i-1} + (1-\alpha_i) P_i$

new control points

Note: here we assume $P_{-1} = 0$ and $P_{n+1} = 0$

Summary

The expression obtained for $P_{n+1}(t)$ is:

$$P_{n+1}(t) = \sum_{i=0}^{n+1} \binom{n+1}{i} t^i (1-t)^{n+1-i} \left(\frac{i}{n+1} P_{i-1} + \left(1 - \frac{i}{n+1}\right) P_i \right)$$

$$P_{n+1}(t) = \sum_{i=0}^{n+1} B_{n+1,i}(t) Q_i$$

where $Q_i = \alpha_i P_{i-1} + (1-\alpha_i) P_i$, $\alpha_i = \frac{i}{n+1}$
and $Q_0 = P_0$, $Q_{n+1} = P_n$

Summary

The expression obtained for $P_{n+1}(t)$ is:

$$P_{n+1}(t) = \sum_{i=0}^{n+1} \binom{n+1}{i} t^i (1-t)^{n+1-i} \left(\frac{i}{n+1} P_{i-1} + \left(1 - \frac{i}{n+1}\right) P_i \right)$$

$$P_{n+1}(t) = \sum_{i=0}^{n+1} B_{n+1,i}(t) Q_i$$

where $Q_i = \alpha_i P_{i-1} + (1-\alpha_i) P_i$, $\alpha_i = \frac{i}{n+1}$
and $Q_0 = P_0$, $Q_{n+1} = P_n$

Example

Summary

The expression obtained for $P_{n+1}(t)$ is:

$$P_{n+1}(t) = \sum_{i=0}^{n+1} \binom{n+1}{i} t^i (1-t)^{n+1-i} \left(\frac{i}{n+1} P_{i-1} + \left(1 - \frac{i}{n+1}\right) P_i\right)$$

$$P_{n+1}(t) = \sum_{i=0}^{n+1} B_{n+1,i}(t) Q_i$$

where $Q_i = \alpha_i P_{i-1} + (1-\alpha_i) P_i$, $\alpha_i = \frac{i}{n+1}$
and $Q_0 = P_0$, $Q_{n+1} = P_n$

Example

degree-4 curve (5 control points)

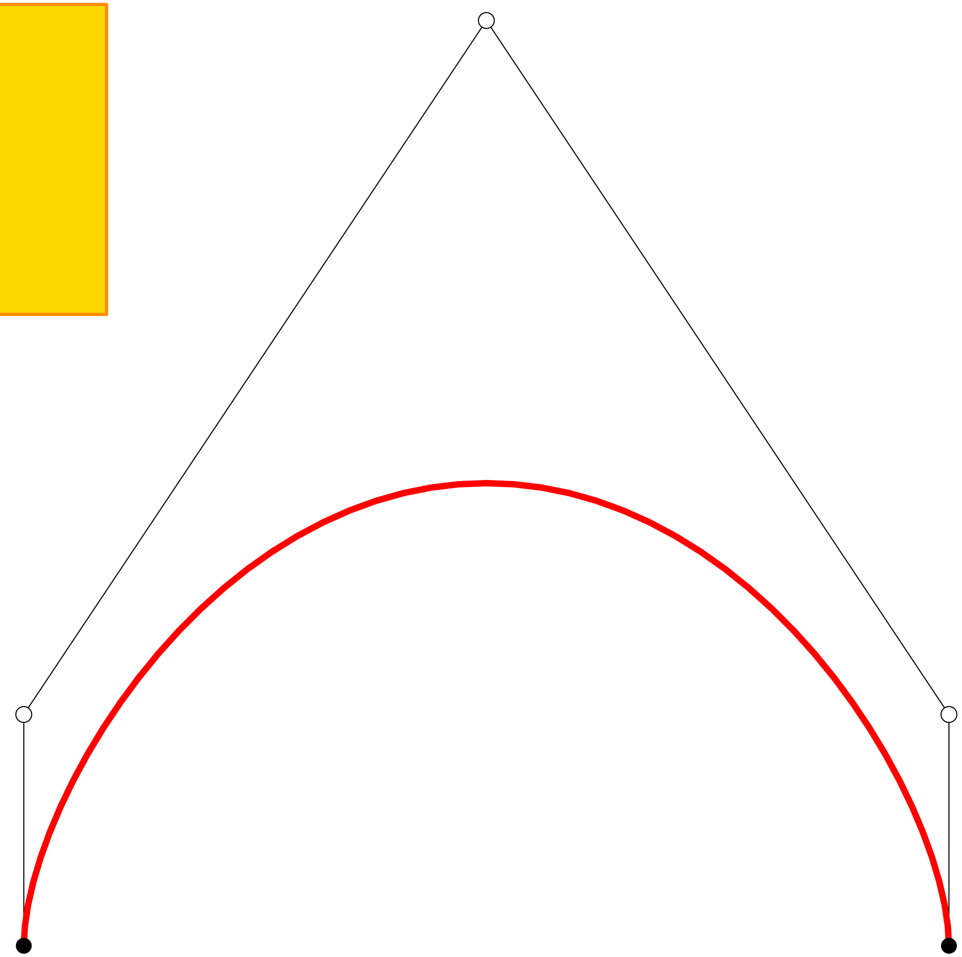


Figure from [G. Farin and D. Hansford, *The Essentials of CAGD*, AK Peters, 2000].

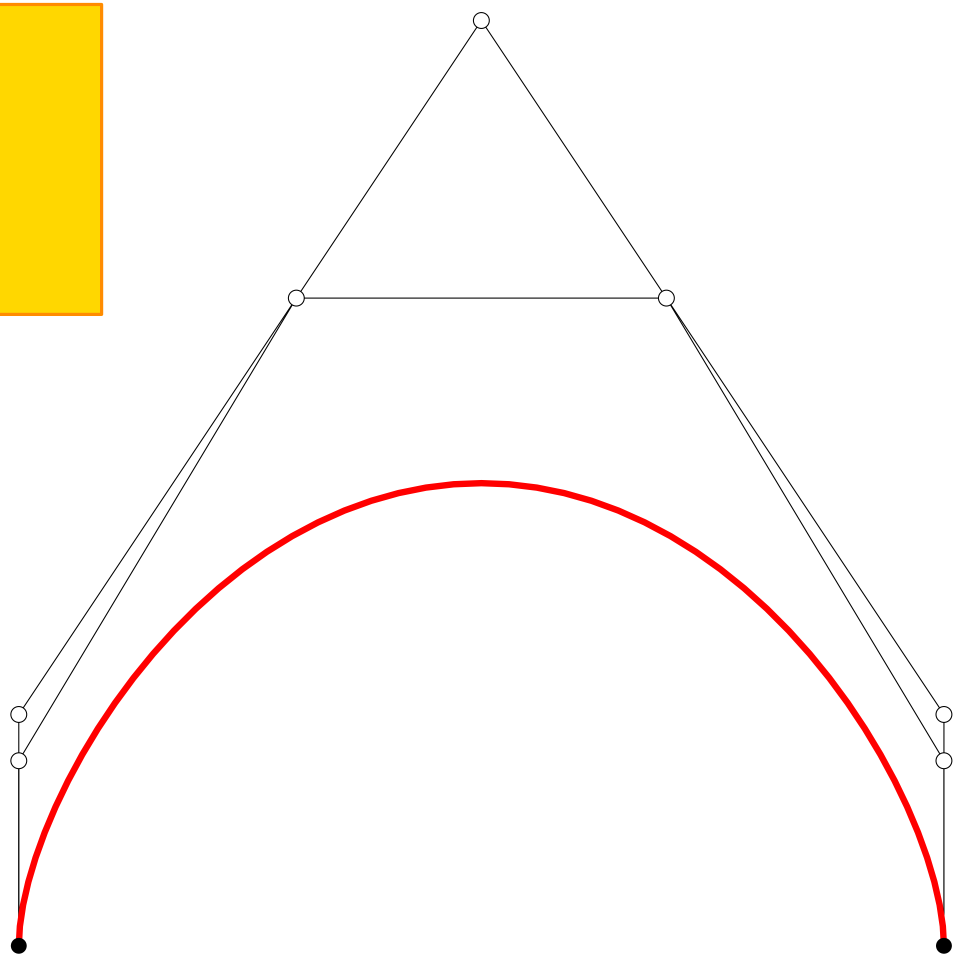
Summary

The expression obtained for $P_{n+1}(t)$ is:

$$P_{n+1}(t) = \sum_{i=0}^{n+1} \binom{n+1}{i} t^i (1-t)^{n+1-i} \left(\frac{i}{n+1} P_{i-1} + \left(1 - \frac{i}{n+1}\right) P_i\right)$$

$$P_{n+1}(t) = \sum_{i=0}^{n+1} B_{n+1,i}(t) Q_i$$

where $Q_i = \alpha_i P_{i-1} + (1-\alpha_i) P_i$, $\alpha_i = \frac{i}{n+1}$
and $Q_0 = P_0$, $Q_{n+1} = P_n$



Example

degree-4 curve (5 control points)

degree-5 curve (6 control points)

Figure from [G. Farin and D. Hansford, *The Essentials of CAGD*, AK Peters, 2000].

DEGREE ELEVATION

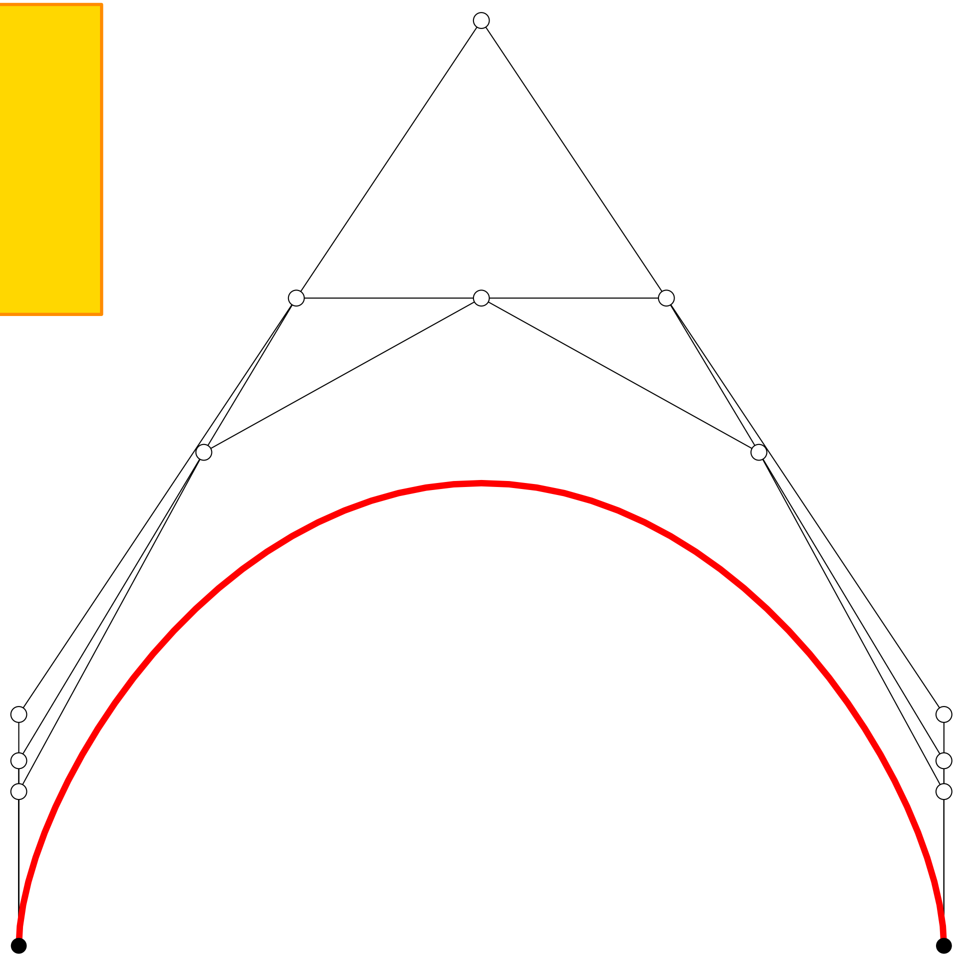
Summary

The expression obtained for $P_{n+1}(t)$ is:

$$P_{n+1}(t) = \sum_{i=0}^{n+1} \binom{n+1}{i} t^i (1-t)^{n+1-i} \left(\frac{i}{n+1} P_{i-1} + \left(1 - \frac{i}{n+1}\right) P_i \right)$$

$$P_{n+1}(t) = \sum_{i=0}^{n+1} B_{n+1,i}(t) Q_i$$

where $Q_i = \alpha_i P_{i-1} + (1-\alpha_i) P_i$, $\alpha_i = \frac{i}{n+1}$
and $Q_0 = P_0$, $Q_{n+1} = P_n$



Example

degree-4 curve (5 control points)

degree-5 curve (6 control points)

degree-6 curve (7 control points)

Figure from [G. Farin and D. Hansford, *The Essentials of CAGD*, AK Peters, 2000].

Summary

The expression obtained for $P_{n+1}(t)$ is:

$$P_{n+1}(t) = \sum_{i=0}^{n+1} \binom{n+1}{i} t^i (1-t)^{n+1-i} \left(\frac{i}{n+1} P_{i-1} + \left(1 - \frac{i}{n+1}\right) P_i\right)$$

$$P_{n+1}(t) = \sum_{i=0}^{n+1} B_{n+1,i}(t) Q_i$$

where $Q_i = \alpha_i P_{i-1} + (1-\alpha_i) P_i$, $\alpha_i = \frac{i}{n+1}$
and $Q_0 = P_0$, $Q_{n+1} = P_n$

Example

degree-4 curve (5 control points)

degree-5 curve (6 control points)

degree-6 curve (7 control points)

degree-n curve ($n+1$ control points)

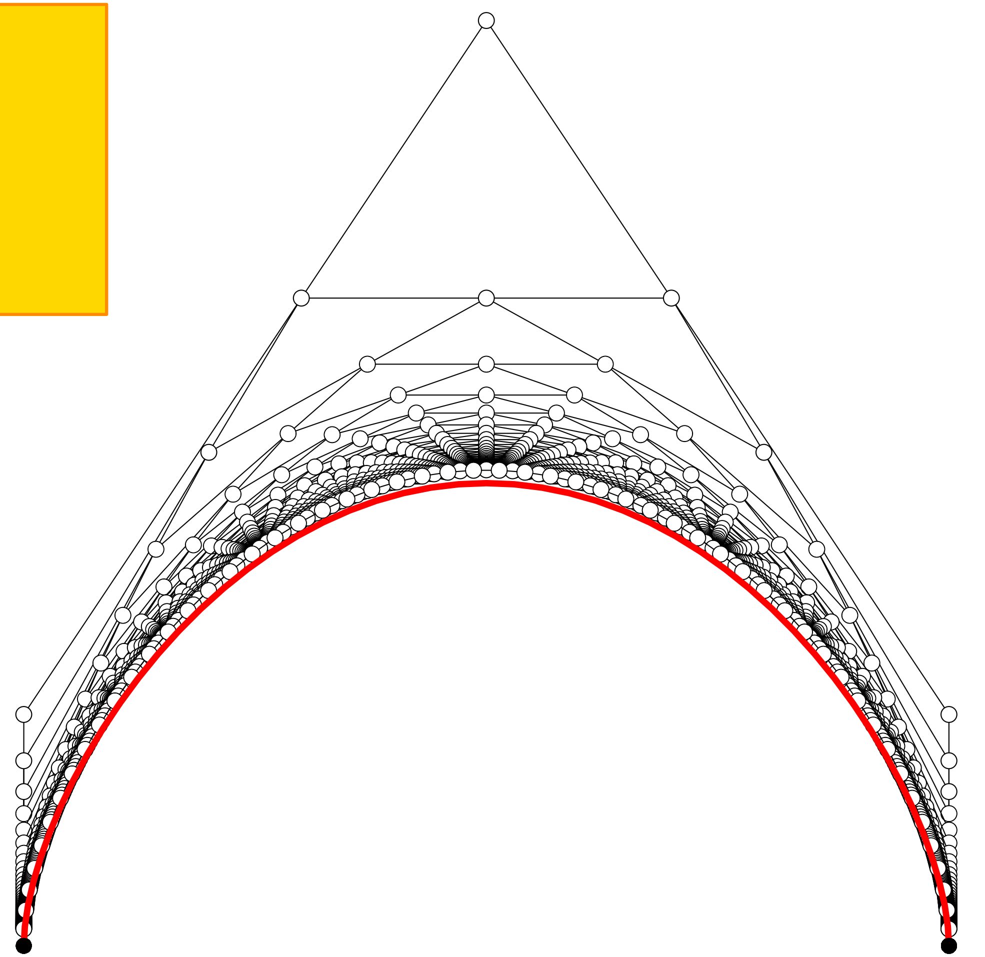


Figure from [G. Farin and D. Hansford, *The Essentials of CAGD*, AK Peters, 2000].

DEGREE ELEVATION

Summary

Question: Can you do degree reduction?

The expression obtained for $P_{n+1}(t)$ is:

$$P_{n+1}(t) = \sum_{i=0}^{n+1} \binom{n+1}{i} t^i (1-t)^{n+1-i} \left(\frac{i}{n+1} P_{i-1} + \left(1 - \frac{i}{n+1}\right) P_i \right)$$

$$P_{n+1}(t) = \sum_{i=0}^{n+1} B_{n+1,i}(t) Q_i$$

where $Q_i = \alpha_i P_{i-1} + (1-\alpha_i) P_i$, $\alpha_i = \frac{i}{n+1}$
and $Q_0 = P_0$, $Q_{n+1} = P_n$

Example

degree-4 curve (5 control points)

degree-5 curve (6 control points)

degree-6 curve (7 control points)

degree-n curve ($n+1$ control points)

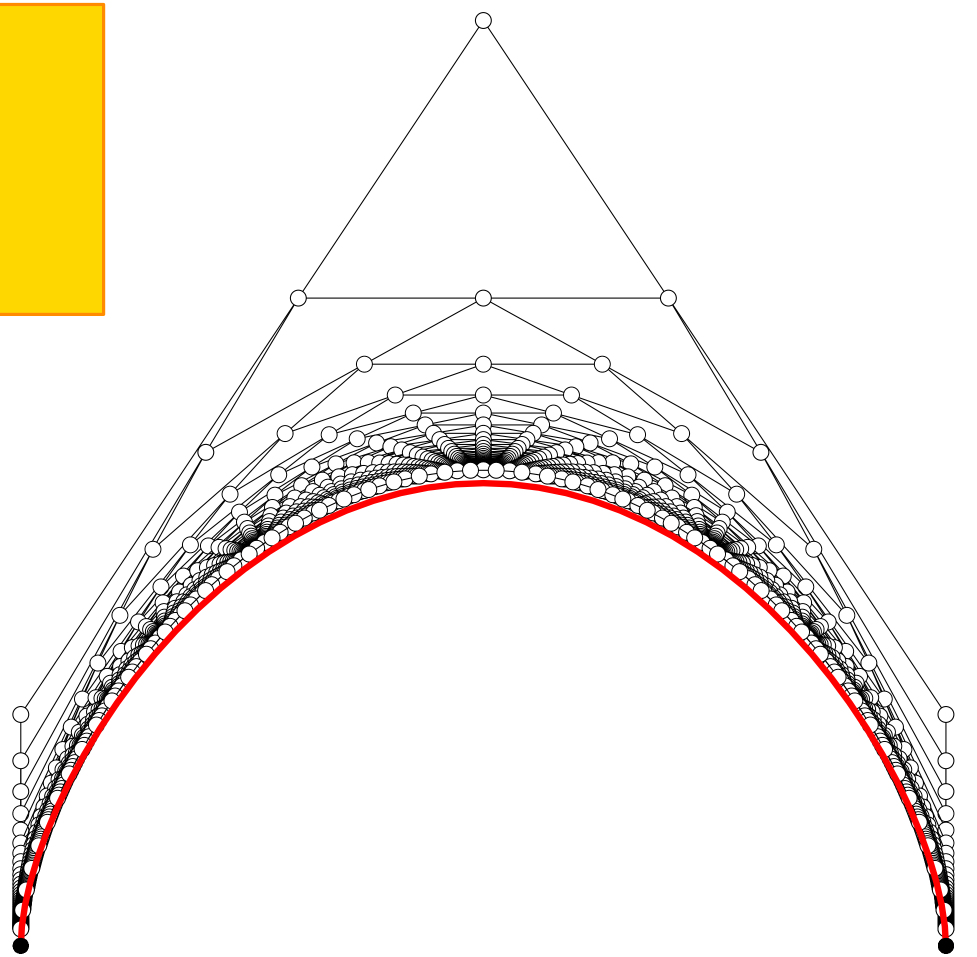
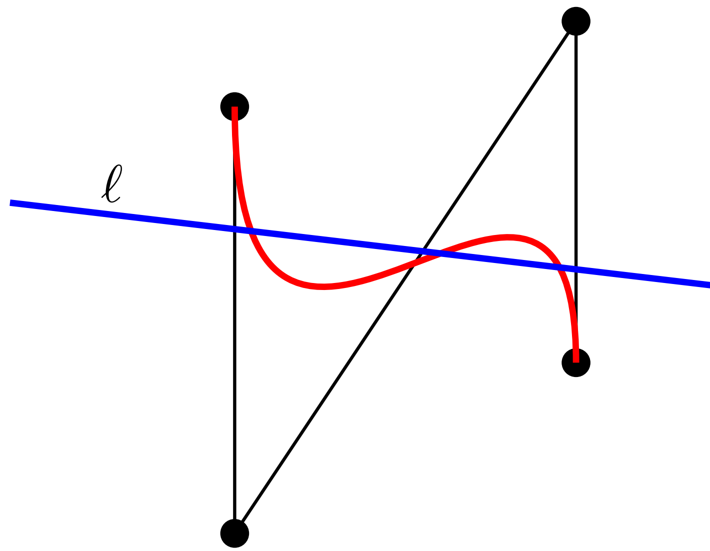


Figure from [G. Farin and D. Hansford, *The Essentials of CAGD*, AK Peters, 2000].

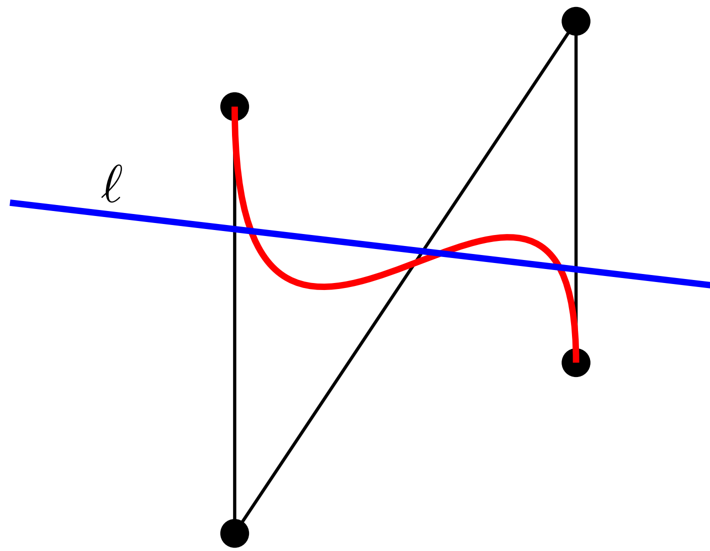
Back to the variation diminishing property

Recall the property: The number of intersections of any line with a Bézier curve is at most the number of intersections of the line with the control polygon



Back to the variation diminishing property

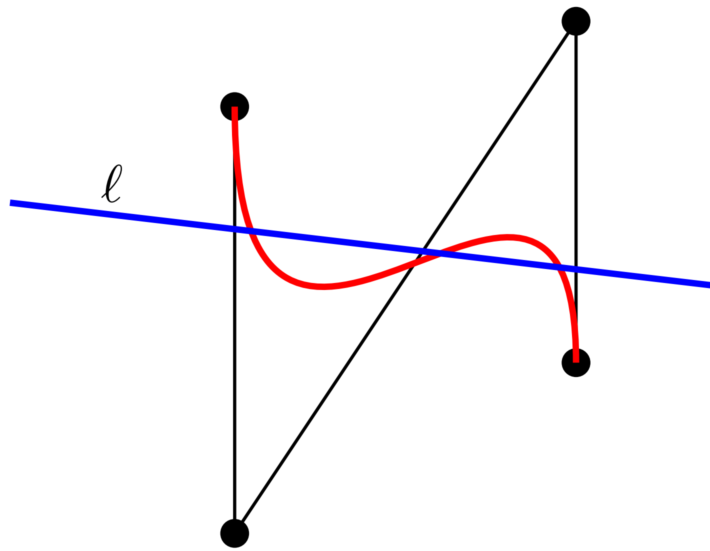
Recall the property: The number of intersections of any line with a Bézier curve is at most the number of intersections of the line with the control polygon



Proof sketch using degree elevation

Back to the variation diminishing property

Recall the property: The number of intersections of any line with a Bézier curve is at most the number of intersections of the line with the control polygon

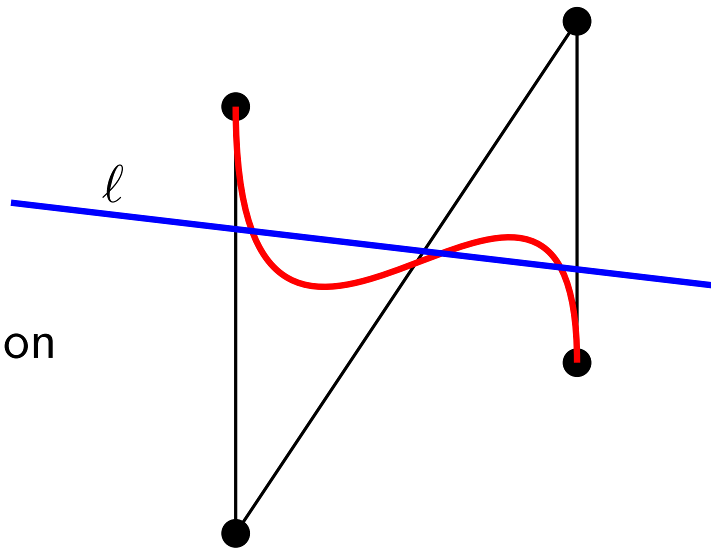


Proof sketch using degree elevation

Recall: linear interpolation is variation diminishing

Back to the variation diminishing property

Recall the property: The number of intersections of any line with a Bézier curve is at most the number of intersections of the line with the control polygon



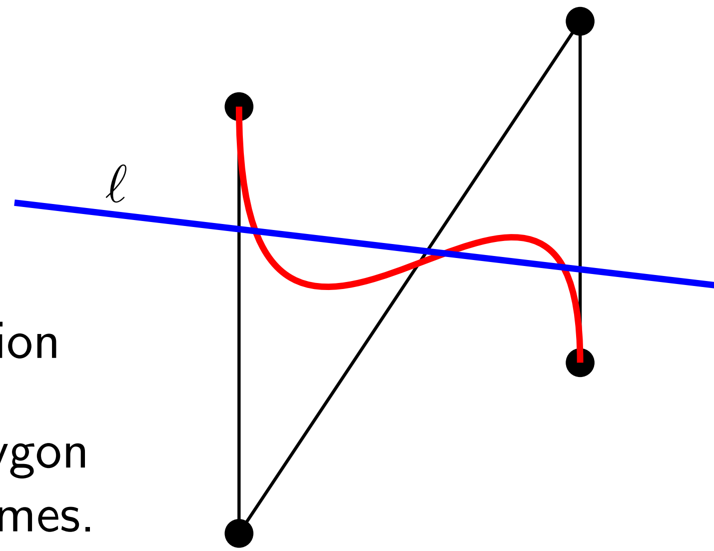
Proof sketch using degree elevation

Recall: linear interpolation is variation diminishing

Observation 2: degree elevation is an instance of linear interpolation

Back to the variation diminishing property

Recall the property: The number of intersections of any line with a Bézier curve is at most the number of intersections of the line with the control polygon



Proof sketch using degree elevation

Recall: linear interpolation is variation diminishing

Observation 2: degree elevation is an instance of linear interpolation

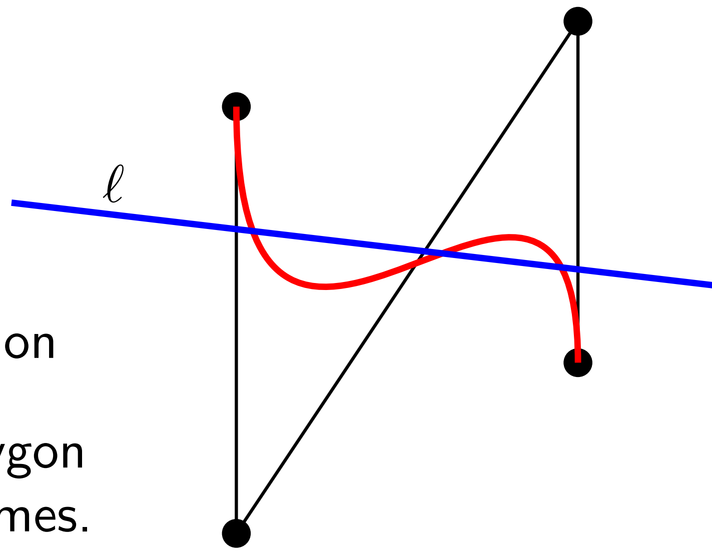
Let R_0 be the control polygon of $P(t)$, let R_1 be the control polygon after increasing the degree by one, and R_k after increasing it k times.

Let ℓ be a given line. Then R_k has no more intersections with ℓ than R_0.

Back to the variation diminishing property

Recall the property: The number of intersections of any line with a Bézier curve is at most the number of intersections of the line with the control polygon

Proof sketch using degree elevation



Recall: linear interpolation is variation diminishing

Observation 2: degree elevation is an instance of linear interpolation

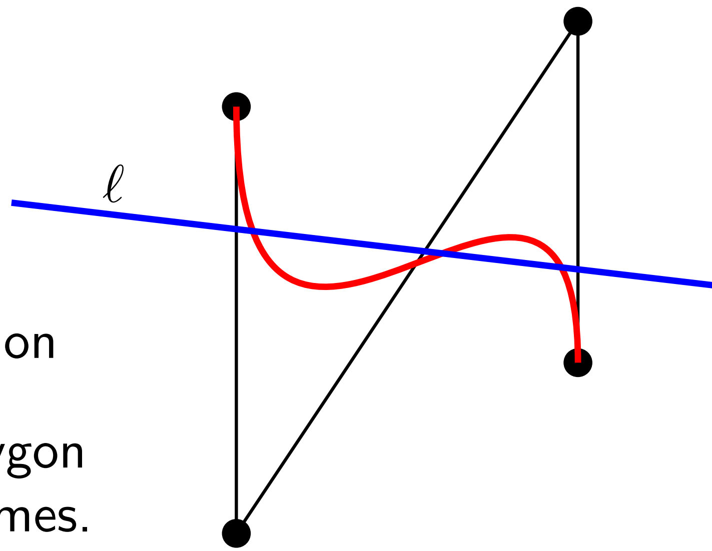
Let R_0 be the control polygon of $P(t)$, let R_1 be the control polygon after increasing the degree by one, and R_k after increasing it k times.

Let ℓ be a given line. Then R_k has no more intersections with ℓ than R_0.

Lemma: in the limit, as $k \to \infty$, R_k approaches the Bézier curve $P(t)$

Back to the variation diminishing property

Recall the property: The number of intersections of any line with a Bézier curve is at most the number of intersections of the line with the control polygon



Proof sketch using degree elevation

Recall: linear interpolation is variation diminishing

Observation 2: degree elevation is an instance of linear interpolation

Let R_0 be the control polygon of $P(t)$, let R_1 be the control polygon after increasing the degree by one, and R_k after increasing it k times.

Let ℓ be a given line. Then R_k has no more intersections with ℓ than R_0.

Lemma: in the limit, as $k \to \infty$, R_k approaches the Bézier curve $P(t)$

Corollary: the Bézier curve $P(t)$ has no more intersections with ℓ than its control polygon

Goal: find Bézier curve that interpolates given points

- In general, the Bézier curve does not interpolate its control points
- There are situations in which the user may want to force the curve through some points

Goal: find Bézier curve that interpolates given points

- In general, the Bézier curve does not interpolate its control points
- There are situations in which the user may want to force the curve through some points

One way to interpolate

Goal: given **data points** $Q_0, \ldots, Q_n$, find **control points** $P_0, \ldots, P_n$ and values $t_0, \ldots, t_n$ such that for each i, $P(t_i) = Q_i$

Goal: find Bézier curve that interpolates given points

- In general, the Bézier curve does not interpolate its control points
- There are situations in which the user may want to force the curve through some points

One way to interpolate

Goal: given **data points** $Q_0, \ldots, Q_n$, find **control points** $P_0, \ldots, P_n$ and values $t_0, \ldots, t_n$ such that for each i, $P(t_i) = Q_i$

This can be expressed as a system of equations, for example for $n = 3$:

Goal: find Bézier curve that interpolates given points

- In general, the Bézier curve does not interpolate its control points
- There are situations in which the user may want to force the curve through some points

One way to interpolate

Goal: given **data points** $Q_0, \ldots, Q_n$, find **control points** $P_0, \ldots, P_n$ and values $t_0, \ldots, t_n$ such that for each i, $P(t_i) = Q_i$

This can be expressed as a system of equations, for example for $n = 3$:

$$P(t_i) = (1-t_i)^3 P_0 + 3t_i(1-t_i)^2 P_1 + 3t_i^2(1-t_i) P_2 + t_i^3 P_3 = Q_i \quad i = 0, 1, 2, 3$$

Goal: find Bézier curve that interpolates given points

- In general, the Bézier curve does not interpolate its control points
- There are situations in which the user may want to force the curve through some points

One way to interpolate

Goal: given **data points** $Q_0, \ldots, Q_n$, find **control points** $P_0, \ldots, P_n$ and values $t_0, \ldots, t_n$ such that for each i, $P(t_i) = Q_i$

This can be expressed as a system of equations, for example for $n = 3$:

$$P(t_i) = (1-t_i)^3 P_0 + 3t_i(1-t_i)^2 P_1 + 3t_i^2(1-t_i)P_2 + t_i^3 P_3 = Q_i \quad i = 0, 1, 2, 3$$

$$\begin{pmatrix} (1-t_0)^3 & 3t_0(1-t_0)^2 & 3t_0^2(1-t_0) & t_0^3 \\ (1-t_1)^3 & 3t_1(1-t_1)^2 & 3t_1^2(1-t_1) & t_1^3 \\ (1-t_2)^3 & 3t_2(1-t_2)^2 & 3t_2^2(1-t_2) & t_2^3 \\ (1-t_3)^3 & 3t_3(1-t_3)^2 & 3t_3^2(1-t_3) & t_3^3 \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix}$$

INTERPOLATION WITH BÉZIER CURVES

Goal: find Bézier curve that interpolates given points

- In general, the Bézier curve does not interpolate its control points
- There are situations in which the user may want to force the curve through some points

One way to interpolate

Goal: given **data points** $Q_0, \ldots, Q_n$, find **control points** $P_0, \ldots, P_n$ and values $t_0, \ldots, t_n$ such that for each i, $P(t_i) = Q_i$

This can be expressed as a system of equations, for example for $n = 3$:

$$P(t_i) = (1-t_i)^3 P_0 + 3t_i(1-t_i)^2 P_1 + 3t_i^2(1-t_i)P_2 + t_i^3 P_3 = Q_i \quad i = 0,1,2,3$$

$$\begin{pmatrix} (1-t_0)^3 & 3t_0(1-t_0)^2 & 3t_0^2(1-t_0) & t_0^3 \\ (1-t_1)^3 & 3t_1(1-t_1)^2 & 3t_1^2(1-t_1) & t_1^3 \\ (1-t_2)^3 & 3t_2(1-t_2)^2 & 3t_2^2(1-t_2) & t_2^3 \\ (1-t_3)^3 & 3t_3(1-t_3)^2 & 3t_3^2(1-t_3) & t_3^3 \end{pmatrix} \boxed{\begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix}} = \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix}$$

unknowns

We are free to choose the values of the t_is!

Example for $n = 3$

For example, take $t_i = i/n = i/3$

Example for $n = 3$

For example, take $t_i = i/n = i/3$

unknowns

$$\begin{pmatrix} (1-0)^3 & 3\cdot 0(1-0)^2 & 3\cdot 0^2(1-0) & 0^3 \\ (1-1/3)^3 & 3\cdot 1/3(1-1/3)^2 & 3\cdot 1/3^2(1-1/3) & 1/3^3 \\ (1-2/3)^3 & 3\cdot 2/3(1-2/3)^2 & 3\cdot (2/3)^2(1-2/3) & (2/3)^3 \\ (1-1)^3 & 3\cdot 1(1-1)^2 & 3\cdot 1^2(1-1) & 1^3 \end{pmatrix} \boxed{\begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix}} = \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix}$$

Example for $n = 3$

For example, take $t_i = i/n = i/3$

unknowns

$$\begin{pmatrix} (1-0)^3 & 3\cdot 0(1-0)^2 & 3\cdot 0^2(1-0) & 0^3 \\ (1-1/3)^3 & 3\cdot 1/3(1-1/3)^2 & 3\cdot 1/3^2(1-1/3) & 1/3^3 \\ (1-2/3)^3 & 3\cdot 2/3(1-2/3)^2 & 3\cdot (2/3)^2(1-2/3) & (2/3)^3 \\ (1-1)^3 & 3\cdot 1(1-1)^2 & 3\cdot 1^2(1-1) & 1^3 \end{pmatrix} \boxed{\begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix}} = \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ \frac{8}{27} & \frac{4}{9} & \frac{2}{9} & \frac{1}{27} \\ \frac{1}{27} & \frac{2}{9} & \frac{4}{9} & \frac{8}{27} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix}$$

INTERPOLATION WITH BÉZIER CURVES

Example for $n = 3$

For example, take $t_i = i/n = i/3$

unknowns

$$\begin{pmatrix} (1-0)^3 & 3\cdot 0(1-0)^2 & 3\cdot 0^2(1-0) & 0^3 \\ (1-1/3)^3 & 3\cdot 1/3(1-1/3)^2 & 3\cdot 1/3^2(1-1/3) & 1/3^3 \\ (1-2/3)^3 & 3\cdot 2/3(1-2/3)^2 & 3\cdot (2/3)^2(1-2/3) & (2/3)^3 \\ (1-1)^3 & 3\cdot 1(1-1)^2 & 3\cdot 1^2(1-1) & 1^3 \end{pmatrix} \boxed{\begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix}} = \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ \frac{8}{27} & \frac{4}{9} & \frac{2}{9} & \frac{1}{27} \\ \frac{1}{27} & \frac{2}{9} & \frac{4}{9} & \frac{8}{27} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix} \longrightarrow \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -\frac{5}{6} & 3 & -\frac{3}{2} & \frac{1}{3} \\ \frac{1}{3} & -\frac{3}{2} & 3 & -\frac{5}{6} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix}$$

Example for $n = 3$

For example, take $t_i = i/n = i/3$

unknowns

$$\begin{pmatrix} (1-0)^3 & 3\cdot 0(1-0)^2 & 3\cdot 0^2(1-0) & 0^3 \\ (1-1/3)^3 & 3\cdot 1/3(1-1/3)^2 & 3\cdot 1/3^2(1-1/3) & 1/3^3 \\ (1-2/3)^3 & 3\cdot 2/3(1-2/3)^2 & 3\cdot (2/3)^2(1-2/3) & (2/3)^3 \\ (1-1)^3 & 3\cdot 1(1-1)^2 & 3\cdot 1^2(1-1) & 1^3 \end{pmatrix} \boxed{\begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix}} = \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ \frac{8}{27} & \frac{4}{9} & \frac{2}{9} & \frac{1}{27} \\ \frac{1}{27} & \frac{2}{9} & \frac{4}{9} & \frac{8}{27} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix} \longrightarrow \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -\frac{5}{6} & 3 & -\frac{3}{2} & \frac{1}{3} \\ \frac{1}{3} & -\frac{3}{2} & 3 & -\frac{5}{6} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix}$$

Concrete example, take $Q_0 = (0,0)$, $Q_1 = (1,1)$, $Q_2 = (2,1)$, $Q_3 = (3,0)$

INTERPOLATION WITH BÉZIER CURVES

Example for $n = 3$

For example, take $t_i = i/n = i/3$

unknowns

$$\begin{pmatrix} (1-0)^3 & 3\cdot 0(1-0)^2 & 3\cdot 0^2(1-0) & 0^3 \\ (1-1/3)^3 & 3\cdot 1/3(1-1/3)^2 & 3\cdot 1/3^2(1-1/3) & 1/3^3 \\ (1-2/3)^3 & 3\cdot 2/3(1-2/3)^2 & 3\cdot (2/3)^2(1-2/3) & (2/3)^3 \\ (1-1)^3 & 3\cdot 1(1-1)^2 & 3\cdot 1^2(1-1) & 1^3 \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ \frac{8}{27} & \frac{4}{9} & \frac{2}{9} & \frac{1}{27} \\ \frac{1}{27} & \frac{2}{9} & \frac{4}{9} & \frac{8}{27} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix} \longrightarrow \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -\frac{5}{6} & 3 & -\frac{3}{2} & \frac{1}{3} \\ \frac{1}{3} & -\frac{3}{2} & 3 & -\frac{5}{6} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix}$$

Concrete example, take $Q_0 = (0,0)$, $Q_1 = (1,1)$, $Q_2 = (2,1)$, $Q_3 = (3,0)$

$$\begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -\frac{5}{6} & 3 & -\frac{3}{2} & \frac{1}{3} \\ \frac{1}{3} & -\frac{3}{2} & 3 & -\frac{5}{6} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 2 & 1 \\ 3 & 0 \end{pmatrix}$$

Example for $n = 3$

For example, take $t_i = i/n = i/3$

unknowns

$$\begin{pmatrix} (1-0)^3 & 3\cdot 0(1-0)^2 & 3\cdot 0^2(1-0) & 0^3 \\ (1-1/3)^3 & 3\cdot 1/3(1-1/3)^2 & 3\cdot 1/3^2(1-1/3) & 1/3^3 \\ (1-2/3)^3 & 3\cdot 2/3(1-2/3)^2 & 3\cdot (2/3)^2(1-2/3) & (2/3)^3 \\ (1-1)^3 & 3\cdot 1(1-1)^2 & 3\cdot 1^2(1-1) & 1^3 \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ \frac{8}{27} & \frac{4}{9} & \frac{2}{9} & \frac{1}{27} \\ \frac{1}{27} & \frac{2}{9} & \frac{4}{9} & \frac{8}{27} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix} \longrightarrow \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -\frac{5}{6} & 3 & -\frac{3}{2} & \frac{1}{3} \\ \frac{1}{3} & -\frac{3}{2} & 3 & -\frac{5}{6} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \end{pmatrix}$$

Concrete example, take $Q_0 = (0,0)$, $Q_1 = (1,1)$, $Q_2 = (2,1)$, $Q_3 = (3,0)$

$$\begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -\frac{5}{6} & 3 & -\frac{3}{2} & \frac{1}{3} \\ \frac{1}{3} & -\frac{3}{2} & 3 & -\frac{5}{6} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 2 & 1 \\ 3 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 3/2 \\ 2 & 3/2 \\ 3 & 0 \end{pmatrix}$$

Example for $n = 3$

Concrete example, take $Q_0 = (0,0)$, $Q_1 = (1,1)$, $Q_2 = (2,1)$, $Q_3 = (3,0)$

Result:

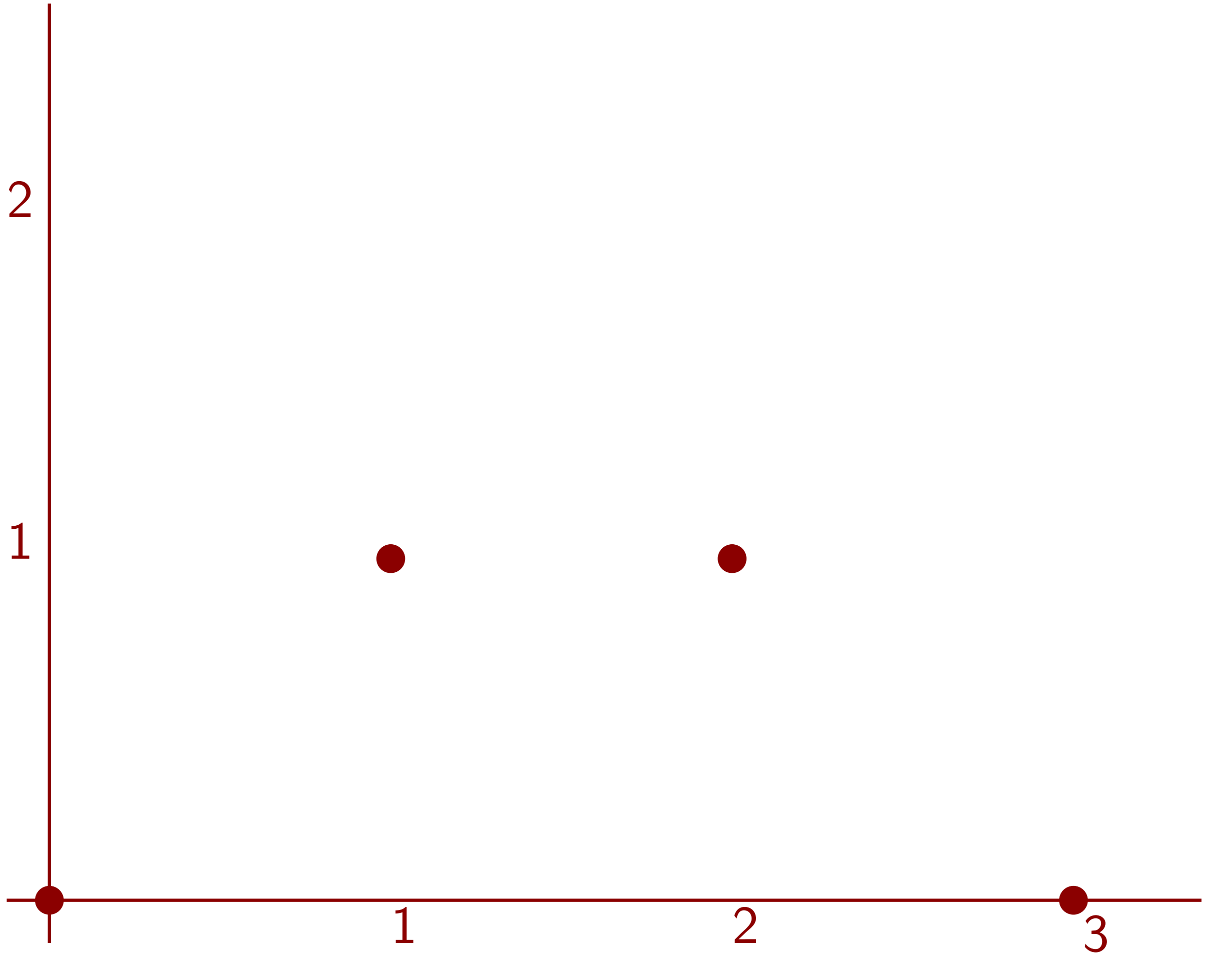
$$\begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 3/2 \\ 2 & 3/2 \\ 3 & 0 \end{pmatrix}$$

Example for $n = 3$

Concrete example, take $Q_0 = (0,0)$, $Q_1 = (1,1)$, $Q_2 = (2,1)$, $Q_3 = (3,0)$

Result:

$$\begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 3/2 \\ 2 & 3/2 \\ 3 & 0 \end{pmatrix}$$

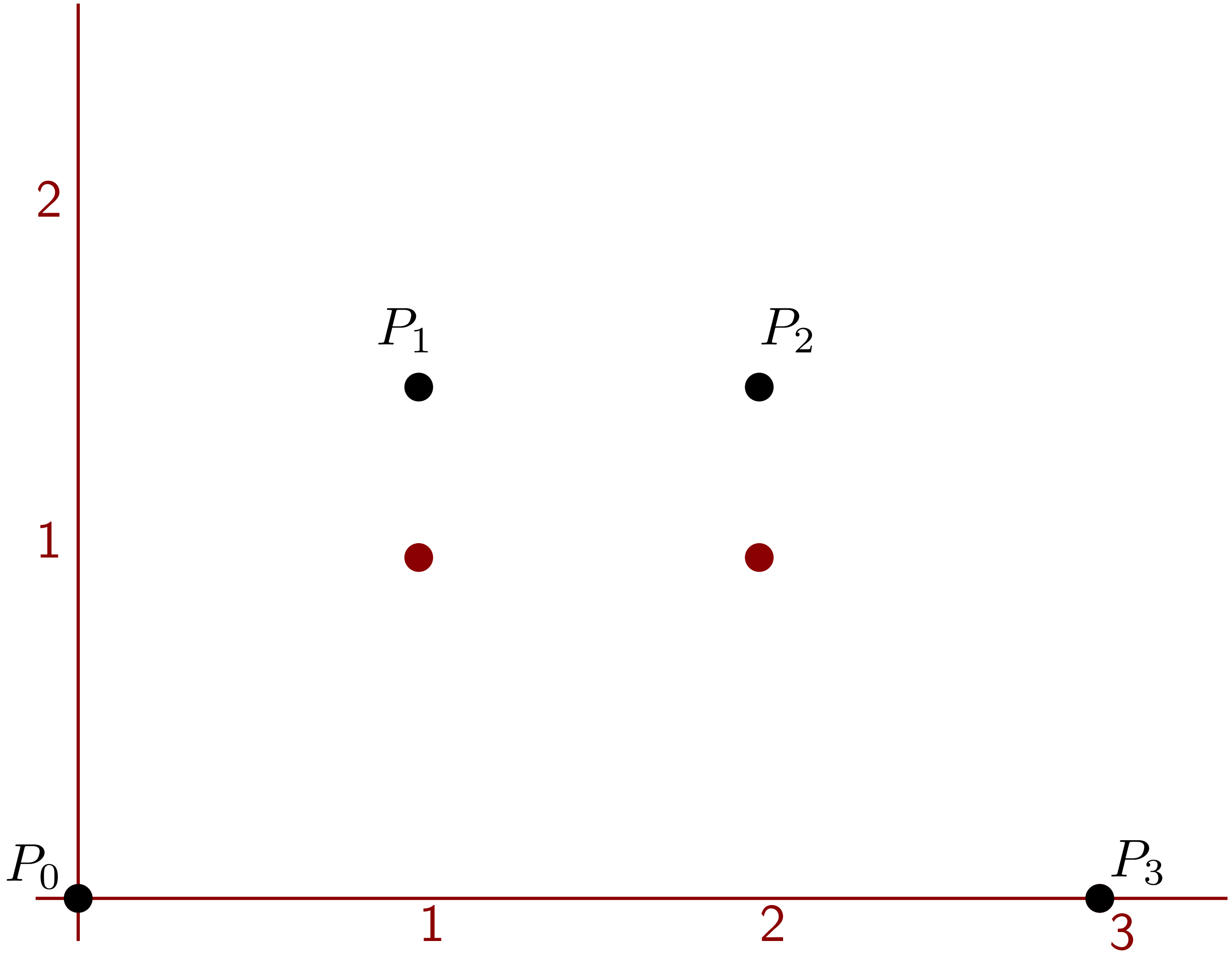


Example for $n = 3$

Concrete example, take $Q_0 = (0,0)$, $Q_1 = (1,1)$, $Q_2 = (2,1)$, $Q_3 = (3,0)$

Result:

$$\begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 3/2 \\ 2 & 3/2 \\ 3 & 0 \end{pmatrix}$$

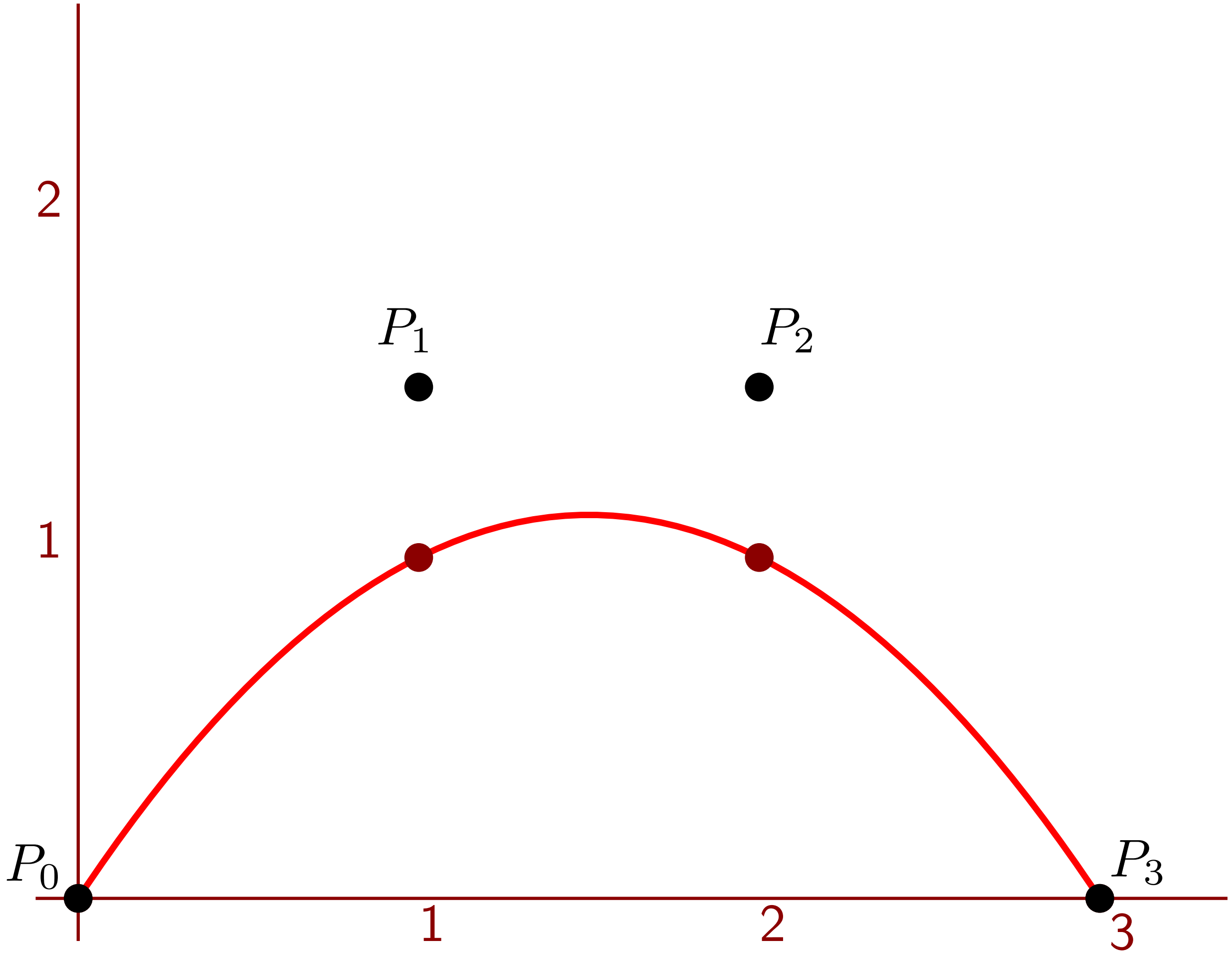


Example for $n = 3$

Concrete example, take $Q_0 = (0, 0)$, $Q_1 = (1, 1)$, $Q_2 = (2, 1)$, $Q_3 = (3, 0)$

Result:

$$\begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 3/2 \\ 2 & 3/2 \\ 3 & 0 \end{pmatrix}$$



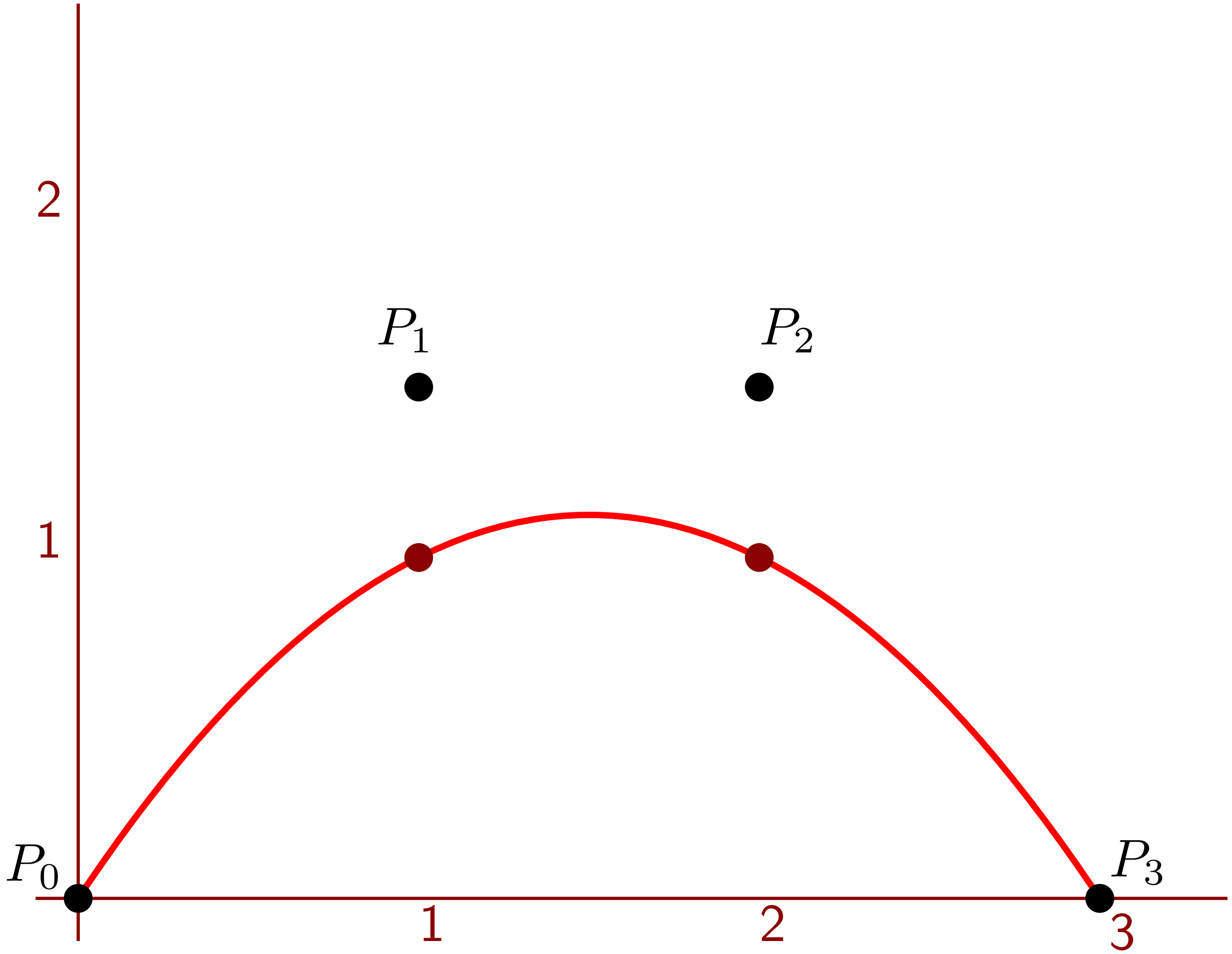
Example for $n = 3$

Concrete example, take $Q_0 = (0, 0)$, $Q_1 = (1, 1)$, $Q_2 = (2, 1)$, $Q_3 = (3, 0)$

Result:

$$\begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 3/2 \\ 2 & 3/2 \\ 3 & 0 \end{pmatrix}$$

This is only one way to interpolate with Bézier curves, others are possible



Rational Bézier curves

Rational Bézier curves

Each control point has a weight, giving more flexibility to shape the curve

Rational Bézier curves

Each control point has a weight, giving more flexibility to shape the curve

$$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$$

Bézier curve

$$P(t) = \frac{\sum_{i=0}^{n} w_i P_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} = \sum_{i=0}^{n} P_i \left(\frac{w_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} \right)$$

Rational Bézier curve

Rational Bézier curves

Each control point has a weight, giving more flexibility to shape the curve

rational weights

$$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$$

Bézier curve

$$P(t) = \frac{\sum_{i=0}^{n} w_i P_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} = \sum_{i=0}^{n} P_i \left(\frac{w_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} \right)$$

Rational Bézier curve

Rational Bézier curves

Each control point has a weight, giving more flexibility to shape the curve

rational weights

$$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t) \qquad P(t) = \frac{\sum_{i=0}^{n} w_i P_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} = \sum_{i=0}^{n} P_i \left(\frac{w_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} \right)$$

Bézier curve　　　　Rational Bézier curve

- Weights are usually non-negative (otherwise denominator could be zero)

Rational Bézier curves

Each control point has a weight, giving more flexibility to shape the curve

rational weights

$$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$$

Bézier curve

$$P(t) = \frac{\sum_{i=0}^{n} w_i P_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} = \sum_{i=0}^{n} P_i \left(\frac{w_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} \right)$$

Rational Bézier curve

- Weights are usually non-negative (otherwise denominator could be zero)

Advantages: why complicate things so much?

Rational Bézier curves

Each control point has a weight, giving more flexibility to shape the curve

rational weights

$$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$$

Bézier curve

$$P(t) = \frac{\sum_{i=0}^{n} w_i P_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} = \sum_{i=0}^{n} P_i \left(\frac{w_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} \right)$$

Rational Bézier curve

- Weights are usually non-negative (otherwise denominator could be zero)

Advantages: why complicate things so much?

- Invariant under projections

Rational Bézier curves

Each control point has a weight, giving more flexibility to shape the curve

rational weights

$$P(t) = \sum_{i=0}^{n} P_i B_{n,i}(t)$$

Bézier curve

$$P(t) = \frac{\sum_{i=0}^{n} w_i P_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} = \sum_{i=0}^{n} P_i \left(\frac{w_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} \right)$$

Rational Bézier curve

- Weights are usually non-negative (otherwise denominator could be zero)

Advantages: why complicate things so much?

- Invariant under projections
- It can represent conic curves (impossible with Hermite or Bézier curves) (e.g., segments of circles, ellipses, hyperbolas and parabolas)

Understanding rational Bézier curves

Effect of the weights

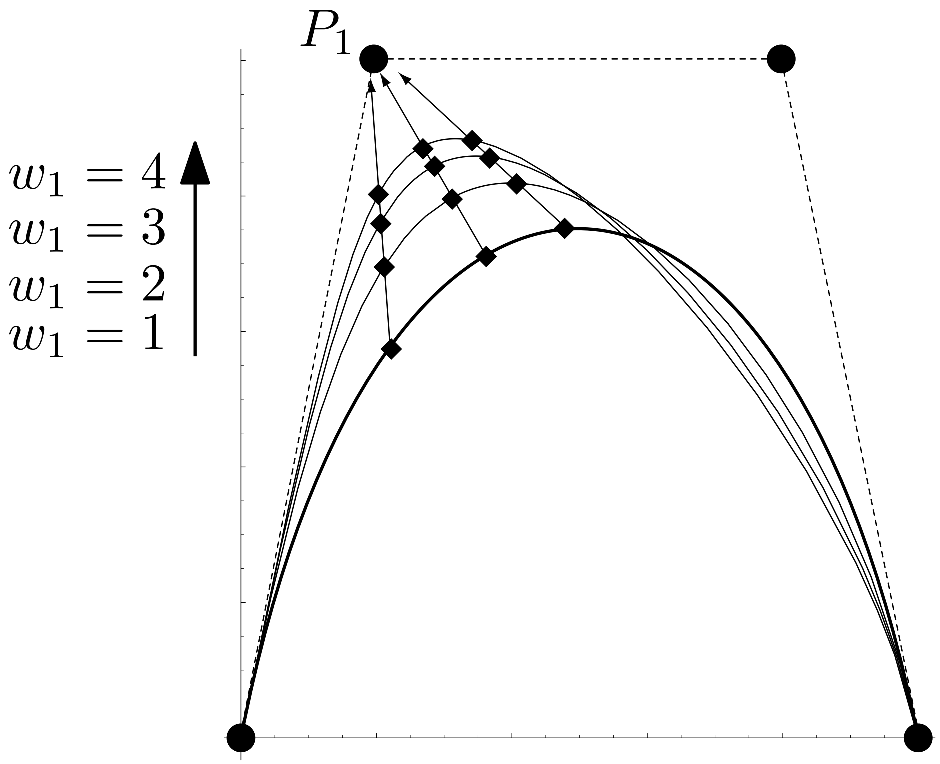
$$P(t) = \frac{\sum_{i=0}^{n} w_i P_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} = \sum_{i=0}^{n} P_i \left(\frac{w_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} \right)$$

Understanding rational Bézier curves

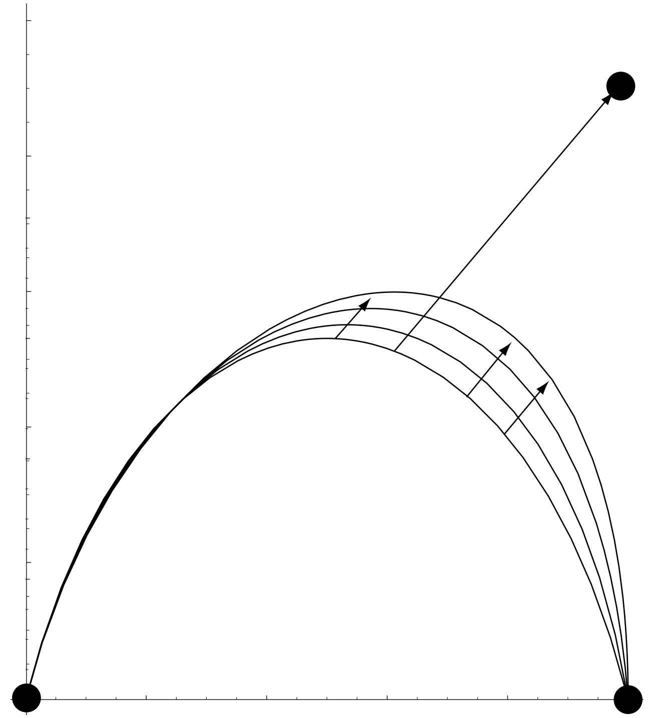
Effect of the weights

$$P(t) = \frac{\sum_{i=0}^{n} w_i P_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} = \sum_{i=0}^{n} P_i \left(\frac{w_i B_{n,i}(t)}{\sum_{j=0}^{n} w_j B_{n,j}(t)} \right)$$

- If $w_i > 1$, the curve gets closer to P_i
- If $w_i < 1$, the curve moves away from P_i



changing weights



moving control point (weight fixed at 1)

Figure from book by Salomon (page 219)

Rational Bézier curves as curves in projective space

2D rational Bézier curve =projection of 3D nonrational Bézier curve onto 2D space!

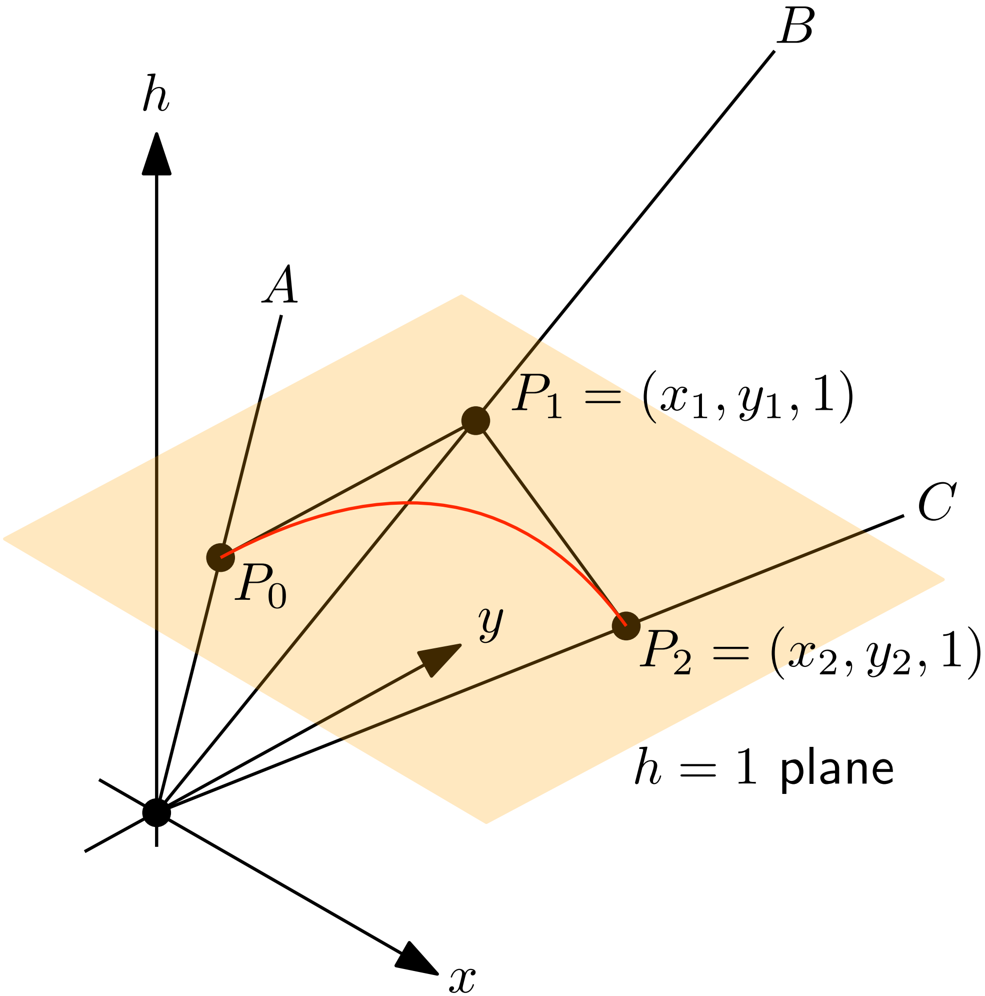


Figure adapted from book by Mortenson

Rational Bézier curves as curves in projective space

2D rational Bézier curve =projection of 3D nonrational Bézier curve onto 2D space!

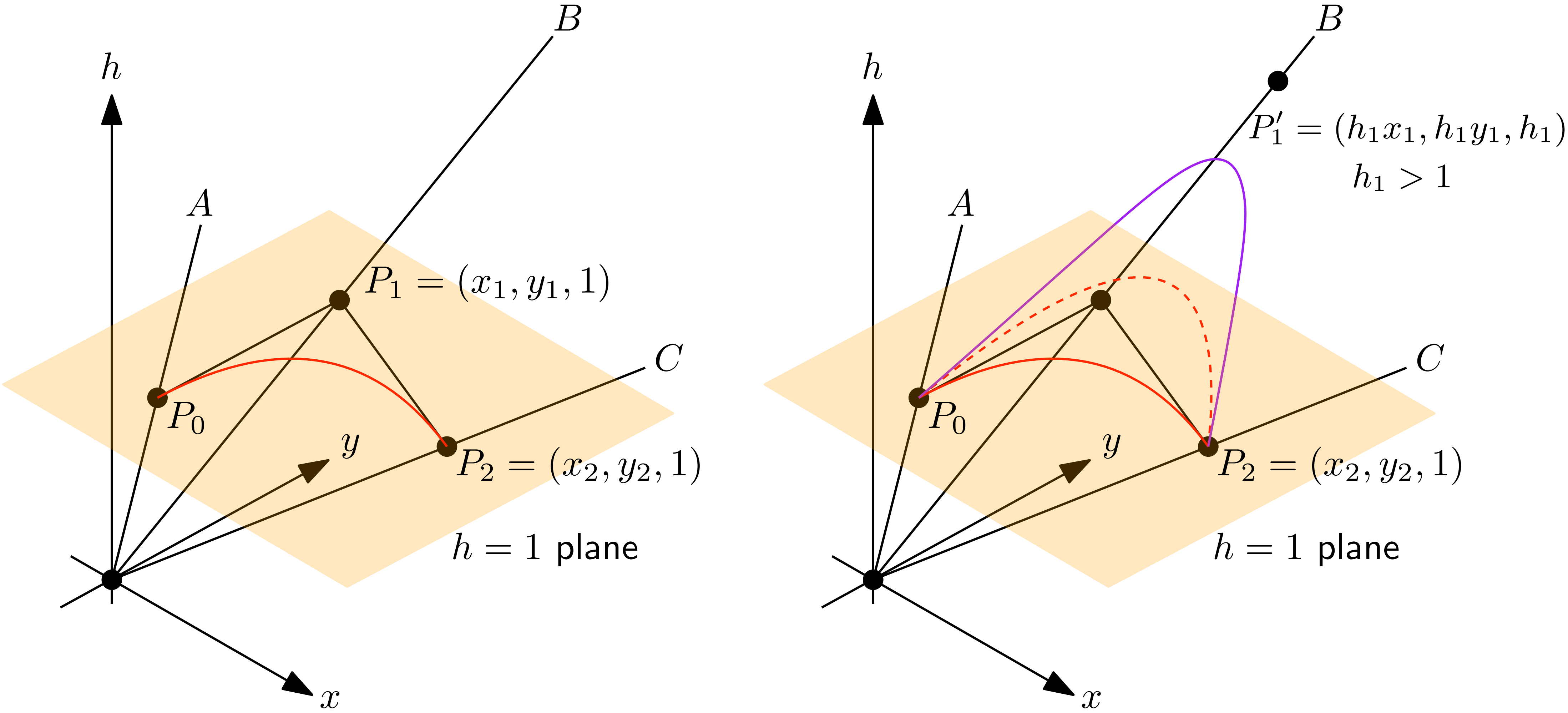


Figure adapted from book by Mortenson

Representing conics with rational Bézier curves

We can represent a conic curve *exactly* with a quadratic rational Bézier curve:

Representing conics with rational Bézier curves

We can represent a conic curve *exactly* with a quadratic rational Bézier curve:

Theorem Consider a conic curve $C(t)$. Then there exist weights w_0, w_1, w_2 and control points P_0, P_1, P_2 such that

$$C(t) = \frac{w_0 P_0 B_{2,0}(t) + w_1 P_1 B_{2,1}(t) + w_2 P_2 B_{2,2}(t)}{w_0 B_{2,0}(t) + w_1 B_{2,1}(t) + w_2 B_{2,2}(t)}$$

Representing conics with rational Bézier curves

We can represent a conic curve *exactly* with a quadratic rational Bézier curve:

Theorem Consider a conic curve $C(t)$. Then there exist weights w_0, w_1, w_2 and control points P_0, P_1, P_2 such that

$$C(t) = \frac{w_0 P_0 B_{2,0}(t) + w_1 P_1 B_{2,1}(t) + w_2 P_2 B_{2,2}(t)}{w_0 B_{2,0}(t) + w_1 B_{2,1}(t) + w_2 B_{2,2}(t)}$$

Example

Take $w_0 = w_2 = 1$ and let $s = \frac{w_1}{1+w_1}$

- $s = 1/2$ produces a **parabolic** arc
- $s < 1/2$ produces an **elliptic** arc
- $s > 1/2$ produces a **hyperbolic** arc

for any three non-colinear control points