COMPUTER AIDED GRAPHIC DESIGN COURSE EXERCISES

MASSIMO NOCENTINI

January 16, 2015

Abstract

This document contains some exercises and collects my work done during the CAGD course given by Prof. Alessandra Sestini and Prof. Costanza Conti at University of Florence.

In particular, here we collect exercises requested by Prof. Costanza Conti about Bezier and BSplines curves. We implement numerical methods using Julia language [1] and everything (code for solving exercises and TEXsources of this document) is under version control, available as open source Git repository [1], under *MIT License*.

CONTENTS

[1] Hosted on `http://github.com/massimo-nocentini/cagd`

1.1 *Curve from simple set of control points*

In Figure 1 we report the very first Bezier curve obtained using our implementation. The curve is obtained using control points $(1,1),(3,4),(5,6),(7,8),(10,2),(1,1)$ in the given order. This plot was the first test for our implementation of code reported in Exercise 1 and requested in Exercise 2: it contains a segmented curve in green which is the control polygon, and a Bezier curve in red built using the given control points.

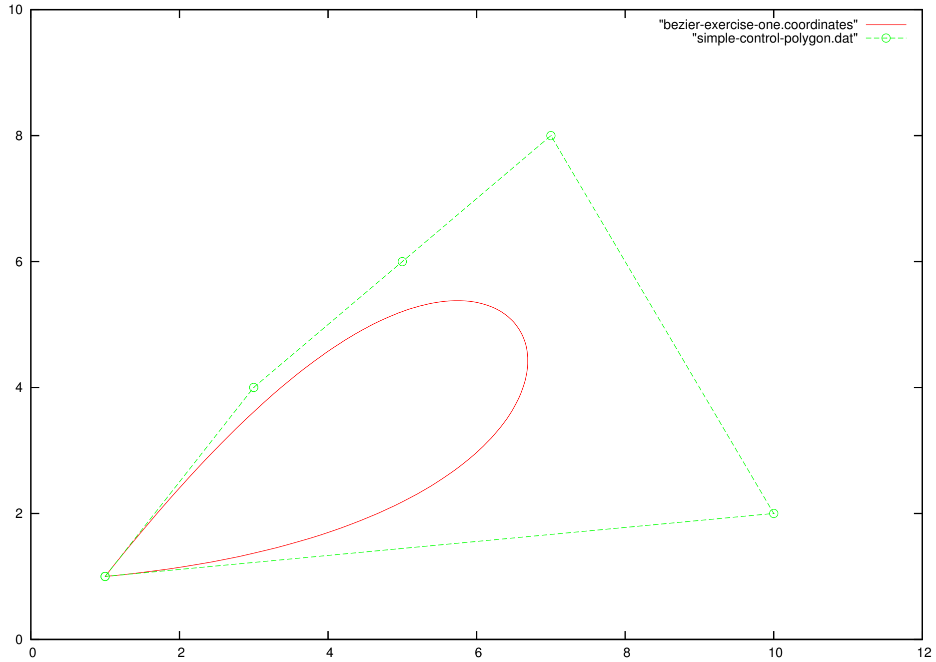

Figure 1: Curve from simple set of control points

1.2 *Curve from parametric specification*

As required in Exercise 3, in Figure 2 we report a Bezier curve matching the following parametric specification:

$$\begin{bmatrix} x(u) \\ y(u) \end{bmatrix} = \begin{bmatrix} 1+u+u^2 \\ u^3 \end{bmatrix}$$

with $u \in [0,1]$. In order to find the control polygon we do simple reductions with *Maxima*:

```
a:v0*(1-t)^3 + v1*3*t*(1-t)^2 + v2*3*(t^2)*(1-t) + v3*t^3;
```

(%o1) $t^3\, v3 + 3\,(1-t)\, t^2\, v2 + 3\,(1-t)^2\, t\, v1 + (1-t)^3\, v0$

```
b:ratsimp(a,t);
```

(%o2) $t^3\,(v3 - 3\,v2 + 3\,v1 - v0) + t^2\,(3\,v2 - 6\,v1 + 3\,v0) + t\,(3\,v1 - 3\,v0) + v0$

```
b = 1 + t + t^2 + 0*t^3;
```

(%o3) $t^3\,(v3 - 3\,v2 + 3\,v1 - v0) + t^2\,(3\,v2 - 6\,v1 + 3\,v0) + t\,(3\,v1 - 3\,v0) + v0 = t^2 + t + 1$

```
solve([coeff(b, t,3) = 0, coeff(b, t,2) = 1, coeff(b, t,1) = 1,
        coeff(b, t,0) = 1], [v0,v1,v2,v3]);
```

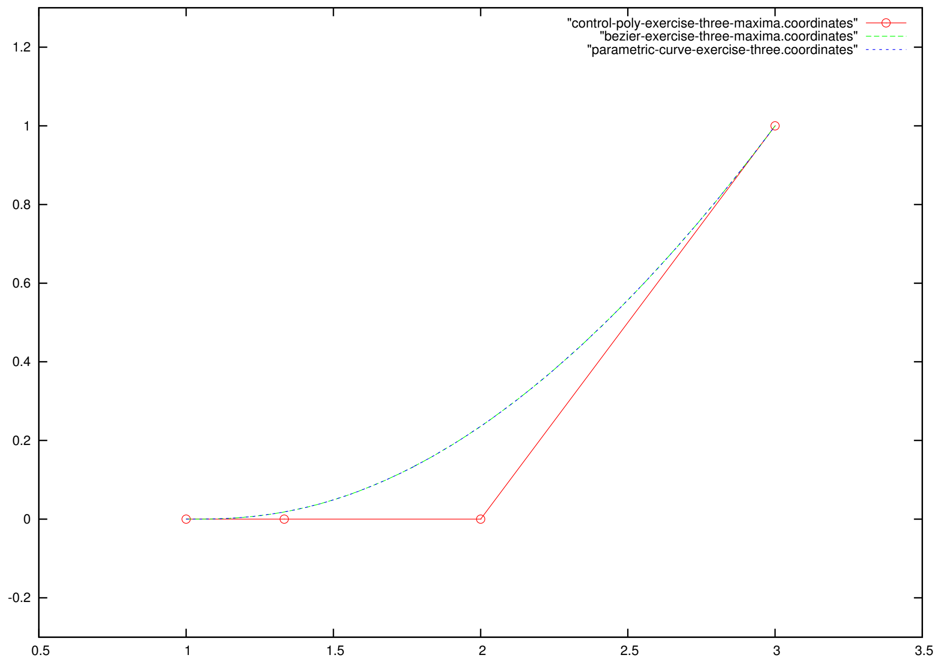

Figure 2: Curve from parametric specification

(%o4) $[[v0 = 1, v1 = \frac{4}{3}, v2 = 2, v3 = 3]]$

```
a:y0*(1-t)^3 + y1*3*t*(1-t)^2 + y2*3*(t^2)*(1-t) + y3*t^3;
```

(%o5) $t^3\, y3 + 3\,(1-t)\,t^2\, y2 + 3\,(1-t)^2\, t\, y1 + (1-t)^3\, y0$

```
b:ratsimp(a,t);
```

(%o6) $t^3\,(y3 - 3\,y2 + 3\,y1 - y0) + t^2\,(3\,y2 - 6\,y1 + 3\,y0) + t\,(3\,y1 - 3\,y0) + y0$

```
b = t^3;
```

(%o7) $t^3\,(y3 - 3\,y2 + 3\,y1 - y0) + t^2\,(3\,y2 - 6\,y1 + 3\,y0) + t\,(3\,y1 - 3\,y0) + y0 = t^3$

```
solve([coeff(b, t,3) = 1, coeff(b, t,2) = 0, coeff(b, t,1) = 0,
        coeff(b, t,0) = 0], [y0,y1,y2,y3]);
```

(%o8) $[[y0 = 0, y1 = 0, y2 = 0, y3 = 1]]$
Hence four control points are $\{(1,0), (\frac{3}{4}, 0), (2,0), (3,1)\}$.

1.3 *Splitted curve on a given parameter $\hat{t}$*

As required in Exercise 4, in Figure 3 we report the control polygon for the original curve in red, and two sets of control points with the same cardinality that, used together, build a Bezier that is the same as the original one. Those sets of points are obtained via subdivision algorithm (which is a clever implementation of classic *de Casteljau algorithm*) fixing parameter $\hat{t} = \frac{1}{4}$: they are colored in green (the relative Bezier in magenta) and in blue (the relative Bezier in cyan), respectively.

1.4 *Repeating the same control point more times*

As required in Exercise 5, in Figure 4 we report two Bezier curves: the red one relative to control points $\{(2,4), (6,12), (10,1), (12,12)\}$ the green one relative to control points $\{(2,4), (6,12), (10,1), (10,1), (10,1), (10,1), (12,12)\}$, ie. with the point $(10,1)$ repeated three more times. We see that curve relative

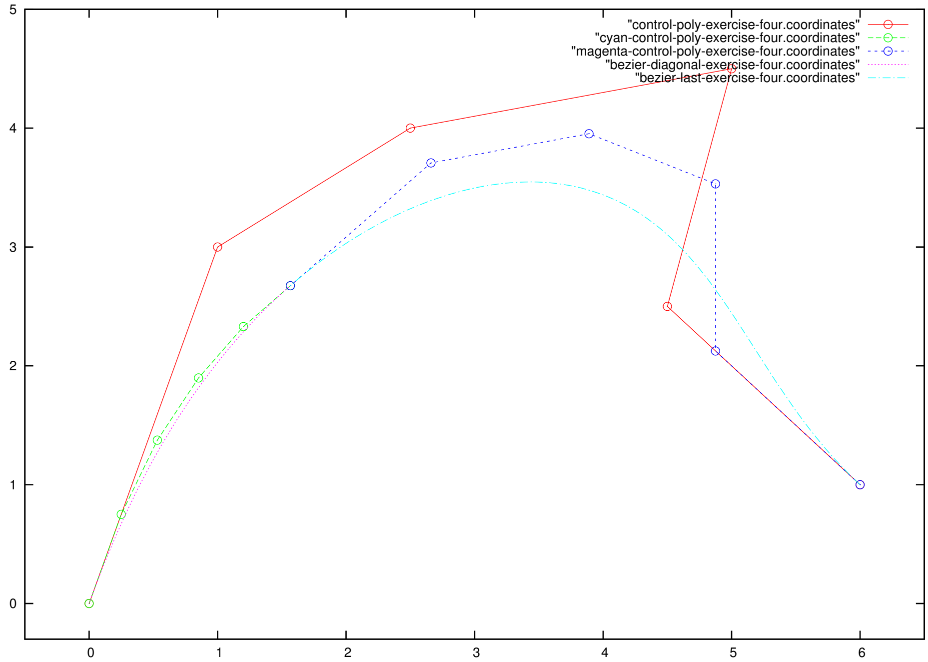

Figure 3: Splitted curve

to the augmented control polygon goes down toward $(10, 1)$ more than the other curve: this can be explained from a probabilistic point of view, since a Bezier curve can be thought as a *mean* of the control polygon, hence repeating point $(10, 1)$ increases its weight.

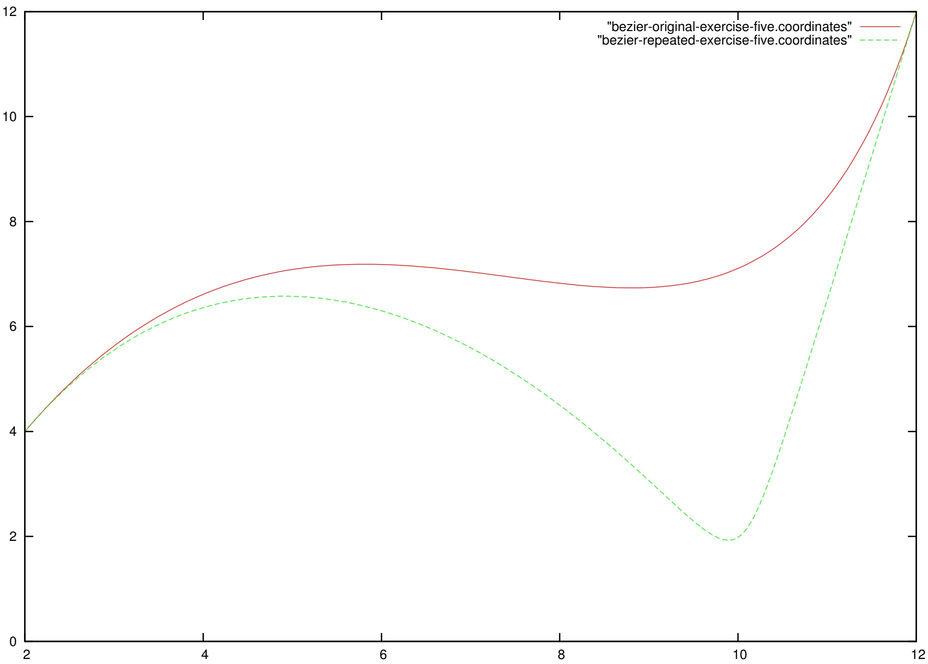

Figure 4: Repeating the control point $(10, 1)$ three more times

1.5 *Increasing degree*

As required in Exercise 6, we start from an original set of control points, plotted in Figure 5, and we proceed by increasing degrees of successive Bezier curves three times. We obtains three augmented set of control points, plotted in Figure 6, respectively. It is possible to check the slow convergence for the sequence of polygons to the Bezier curve and, in Figure 7, the Bezier curves relative to each augmented polygons doesn't change in shape, ie their set of points equal the original one (we simply plot those Bezier curves in the same plot and each one is over the others).

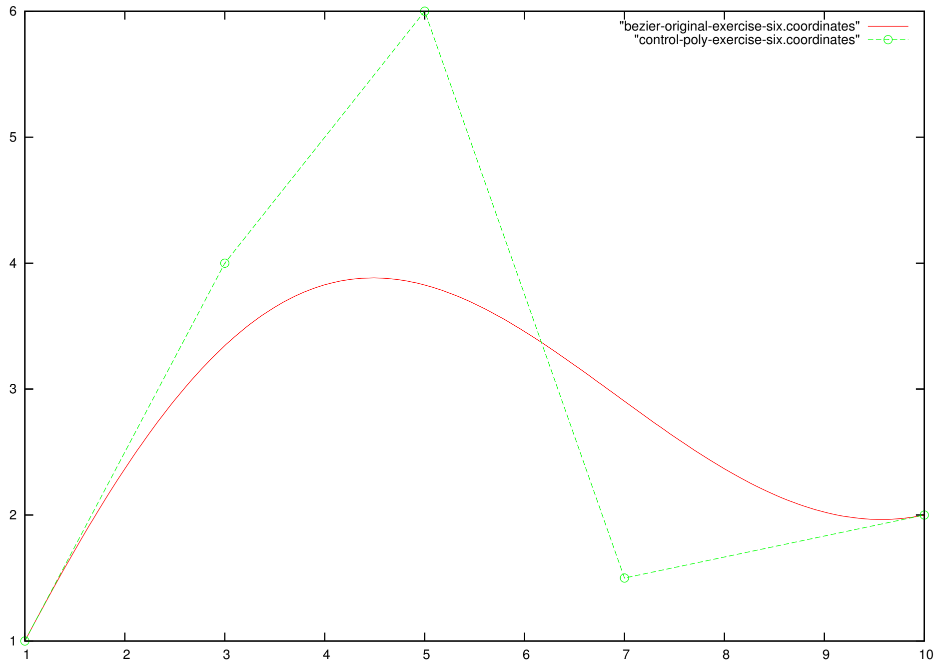

Figure 5: Original curve before increasing degree

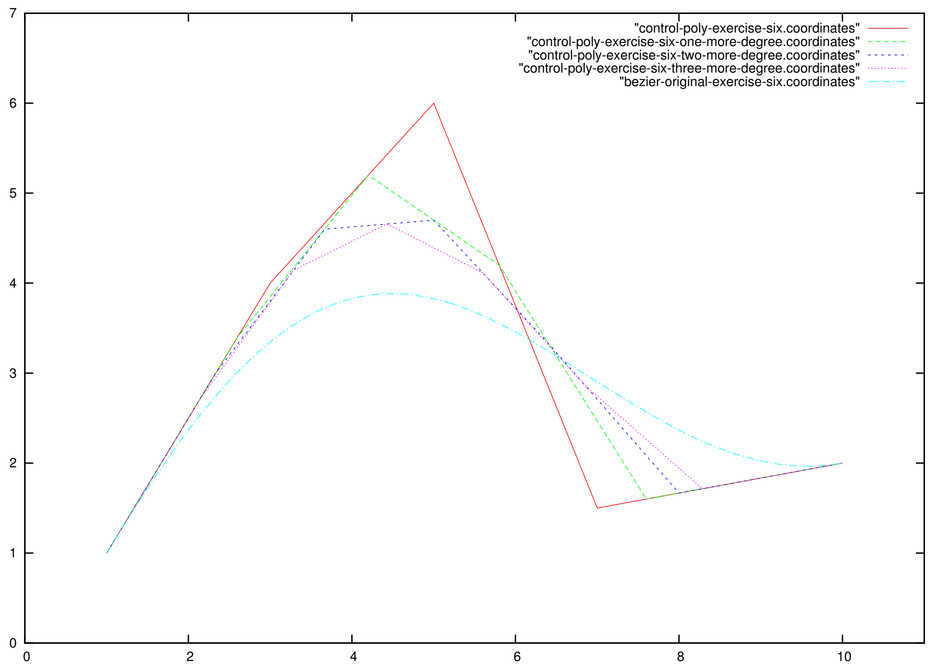

Figure 6: Some control polygons, each one with one more degree

1.6 *Joining curve requiring $\mathcal{C}^0, \mathcal{C}^1, \mathcal{C}^2$*

In this section we report plots about Bezier spline curves, extending a fixed control net, requiring $\mathcal{C}^0, \mathcal{C}^1, \mathcal{C}^2$ for each extension, both on the right (see Figure 8, Figure 9, Figure 10, respectively) both on the left (see Figure 11, Figure 12, Figure 13, respectively).

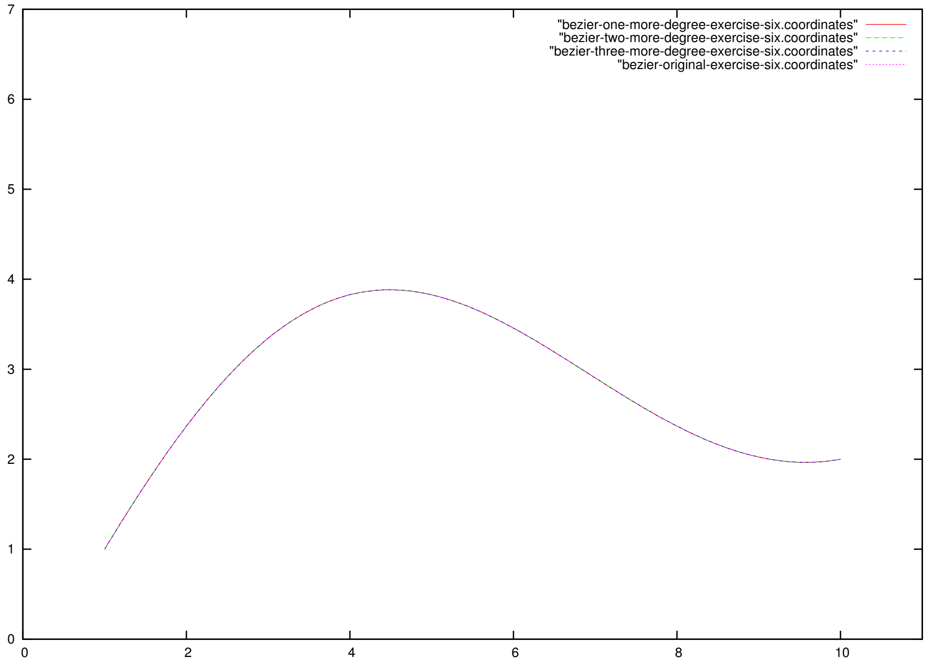

Figure 7: Increasing degree doesn't change the Bezier shape

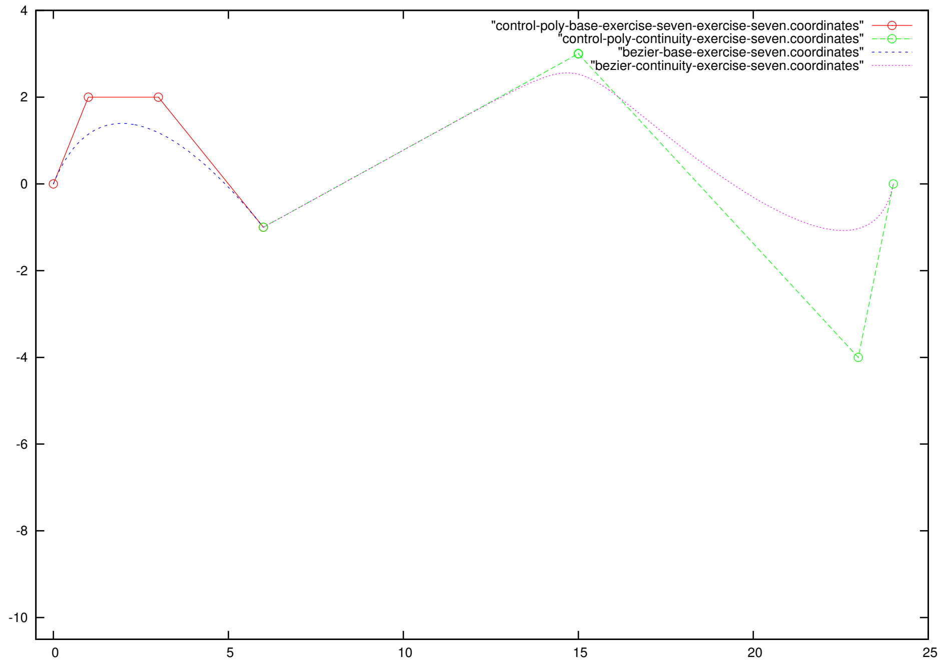

Figure 8: Bezier spline curve, requiring $\mathcal{C}^0$, extending on the right

1.7 *First Polar of a Bezier curve*

In this section we elaborate an extra exercise relative to first derivative of a Bezier curve and the first step of de Casteljau algorithm.

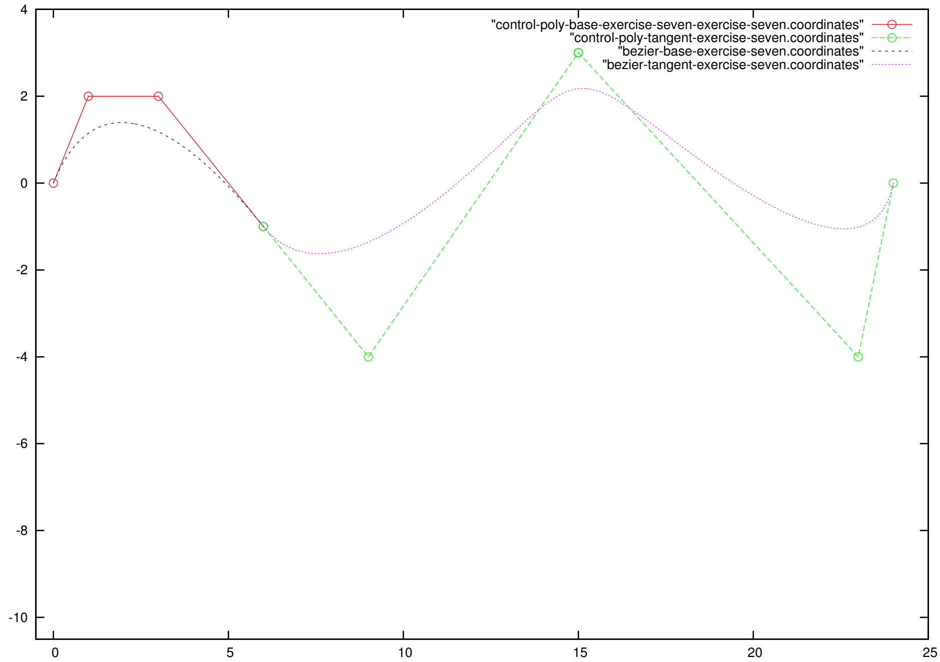

Figure 9: Bezier spline curve, requiring $\mathcal{C}^1$, extending on the right

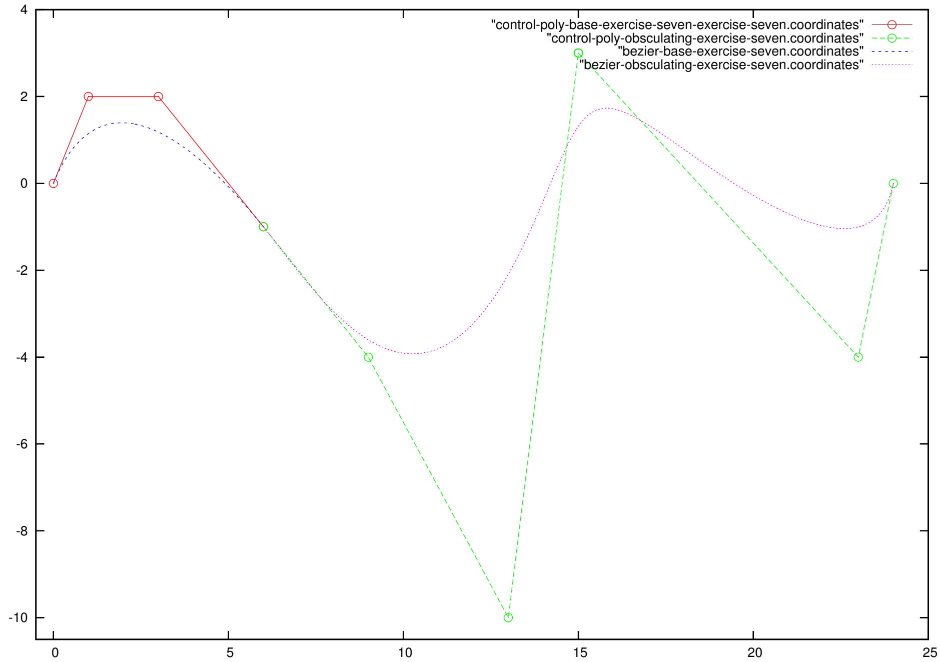

Figure 10: Bezier spline curve, requiring $\mathcal{C}^2$, extending on the right

Let $\mathbf{p}_1(t)$ a Bezier curve built over a control net with n points $\mathbf{b}_0^{(1)}, \ldots, \mathbf{b}_{n-1}^{(1)}$, respect a fixed parameter $\hat{t}$ after one step of de Casteljau algorithm. Just use the formal definition:

$$\begin{aligned}
\mathbf{p}_{1,\hat{t}}(t) &= \sum_{i=0}^{n-1} \mathbf{b}_i^{(1)}(\hat{t}) B_i^{n-1}(t) \\
&= \sum_{i=0}^{n-1} \left((1-\hat{t})\mathbf{b}_i^{(0)} + \hat{t}\mathbf{b}_{i+1}^{(0)} \right) B_i^{n-1}(t) \\
&= \sum_{i=0}^{n-1} \left((1-\hat{t})\mathbf{b}_i^{(0)} + \hat{t}\mathbf{b}_{i+1}^{(0)} - \left((1-t)\mathbf{b}_i^{(0)} + t\mathbf{b}_{i+1}^{(0)} \right) \right) B_i^{n-1}(t) + \sum_{i=0}^{n-1} \mathbf{b}_i^{(1)}(t) B_i^{n-1}(t) \\
&= \sum_{i=0}^{n-1} \left((t-\hat{t})\mathbf{b}_i^{(0)} + (\hat{t}-t)\mathbf{b}_{i+1}^{(0)} \right) B_i^{n-1}(t) + \sum_{i=0}^{n-1} \mathbf{b}_i^{(1)}(t) B_i^{n-1}(t) \\
&= (\hat{t}-t) \sum_{i=0}^{n-1} \left(\mathbf{b}_{i+1}^{(0)} - \mathbf{b}_i^{(0)} \right) B_i^{n-1}(t) + \sum_{i=0}^{n-1} \mathbf{b}_i^{(1)}(t) B_i^{n-1}(t)
\end{aligned}$$

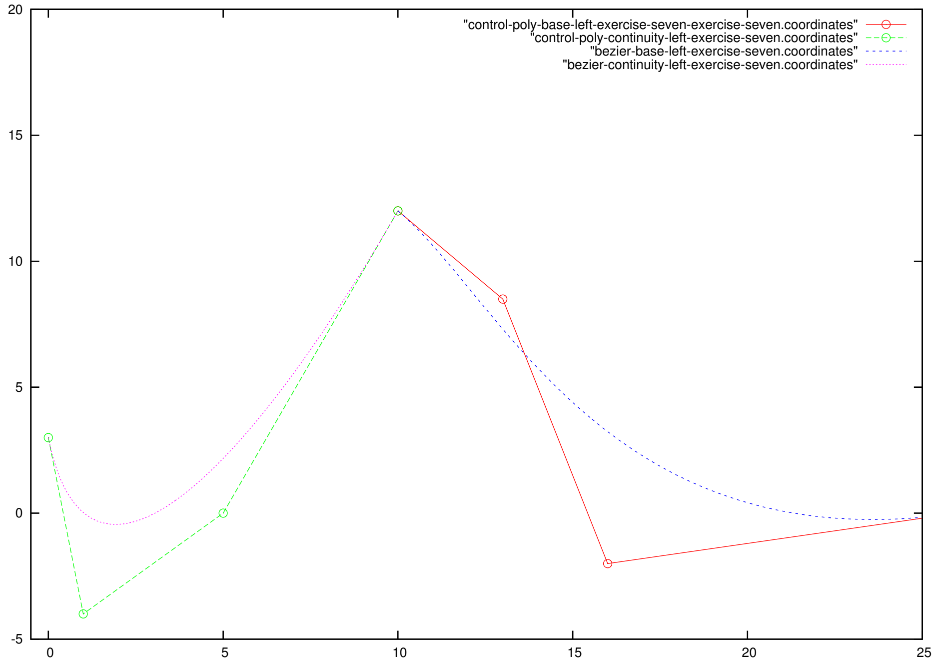

Figure 11: Bezier spline curve, requiring $\mathcal{C}^0$, extending on the left

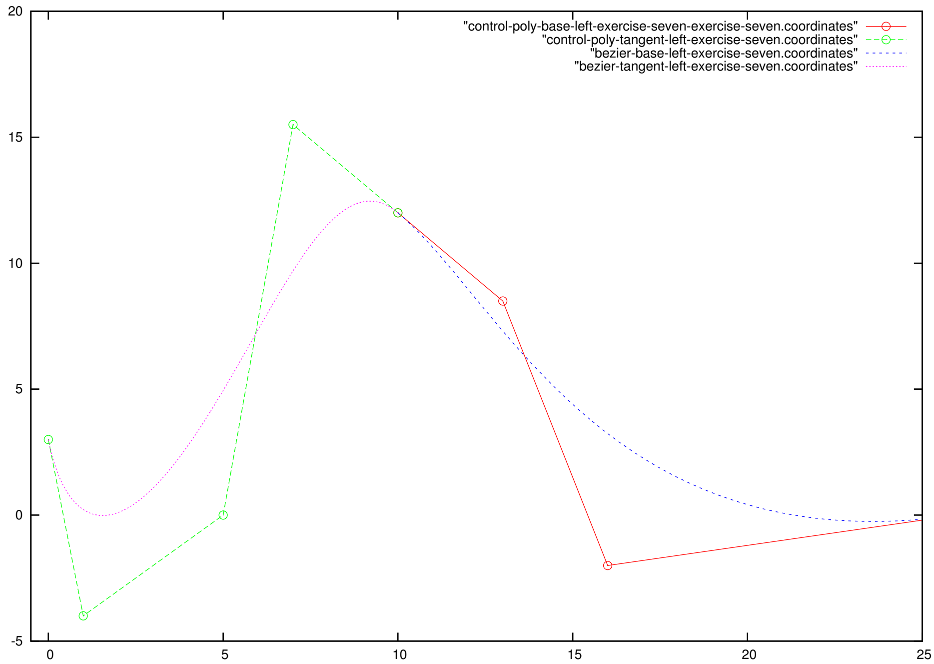

Figure 12: Bezier spline curve, requiring $\mathcal{C}^1$, extending on the left

where, with little abuse of notation, $\mathbf{b}_i^{(1)}(t) = \left((1-t)\mathbf{b}_i^{(0)} + t\mathbf{b}_{i+1}^{(0)}\right)$. Recall that $n\sum_{i=0}^{n-1}\left(\mathbf{b}_{i+1}^{(0)} - \mathbf{b}_i^{(0)}\right)B_i^{n-1}(t)$ is the first derivative of a Bezier curve, hence:

$$\mathbf{p}_{1,\hat{t}}(t) = \mathbf{b}_{1,t}(t) + \frac{\hat{t}-t}{n}\frac{\partial \mathbf{b}_{0,\hat{t}}(t)}{\partial t}$$

So, this is called the "first polar form" of Bezier curve $\mathbf{b}_{0,\hat{t}}(t)$ respect parameter $\hat{t}$. Geometrically, the term $\mathbf{b}_{1,t}(t)$ is an affine combination of points $\mathbf{b}_0^{(0)}, \ldots, \mathbf{b}_n^{(0)}$ respect parameter t (not $\hat{t}$), hence it is a point also; the term $\frac{\hat{t}-t}{n}\frac{\partial \mathbf{b}_{0,\hat{t}}(t)}{\partial t}$ is a vector, so $\mathbf{p}_{1,\hat{t}}(t)$ is a vector applied to a point, yielding a point as required. It is quite interesting how the polar form combine a point produced using t as parameter with the derivative vector produced using

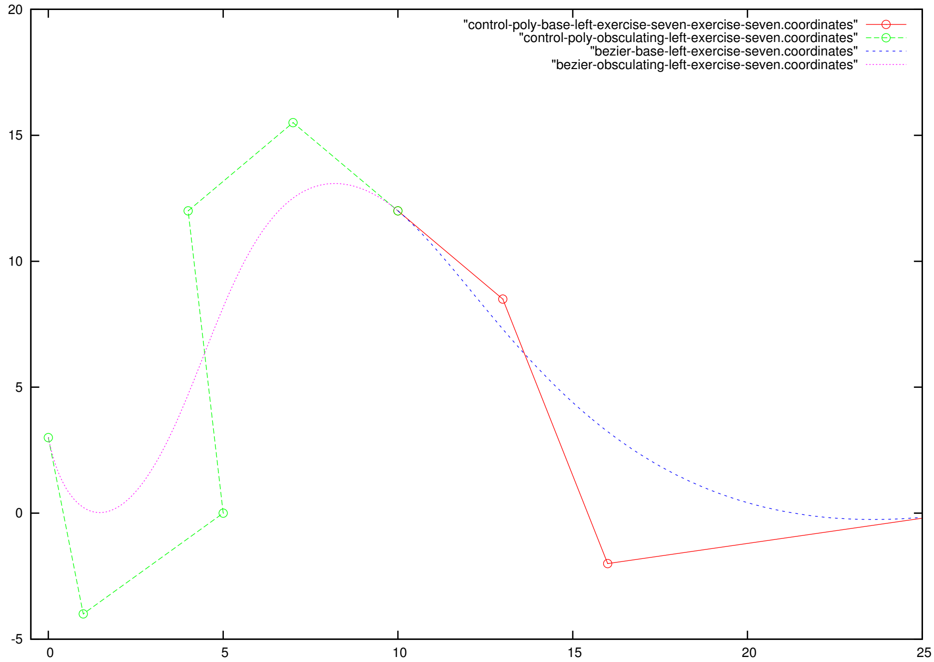

Figure 13: Bezier spline curve, requiring $\mathcal{C}^2$, extending on the left

parameter $\hat{t}$. This derivation comes from [2], page 73, and we report the output of our implementation in Figure 14.

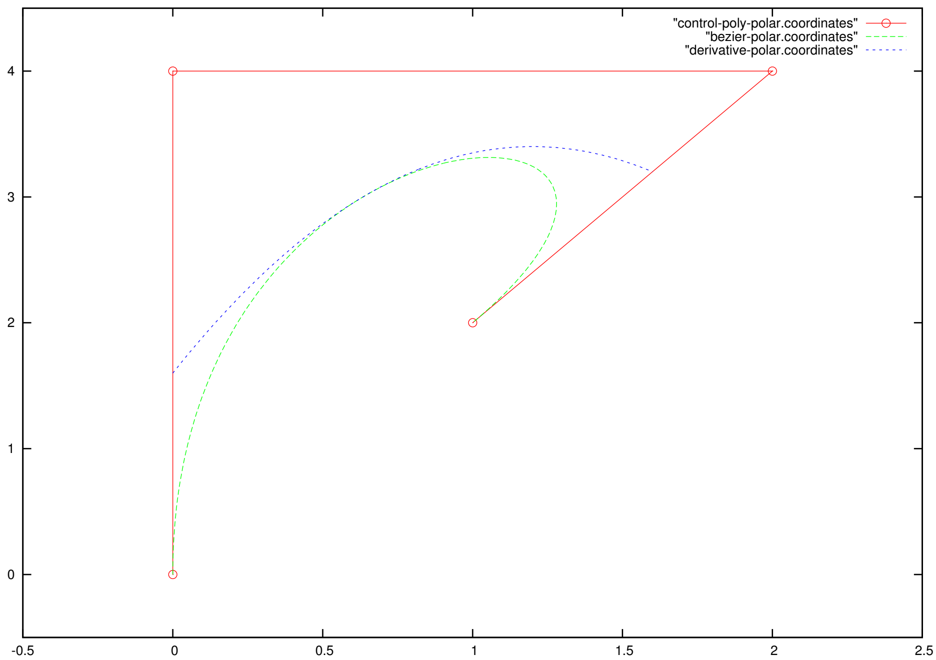

Figure 14: First Polar form of a Bezier curve

2.1 *Mushrooms from clumped, uniformed and closed partitions*

In this section we report three simple BSpline curves, using three different knots partitions: *clumped*, *uniform* and *cyclic* (see Figure 15, Figure 16 and Figure 17 respectively). The control polygon doesn't change and aims to produce a mushroom in cartoon style, each knot has multiplicity 1 (so maximum continuity is required in each junction) and no control point is repeated.

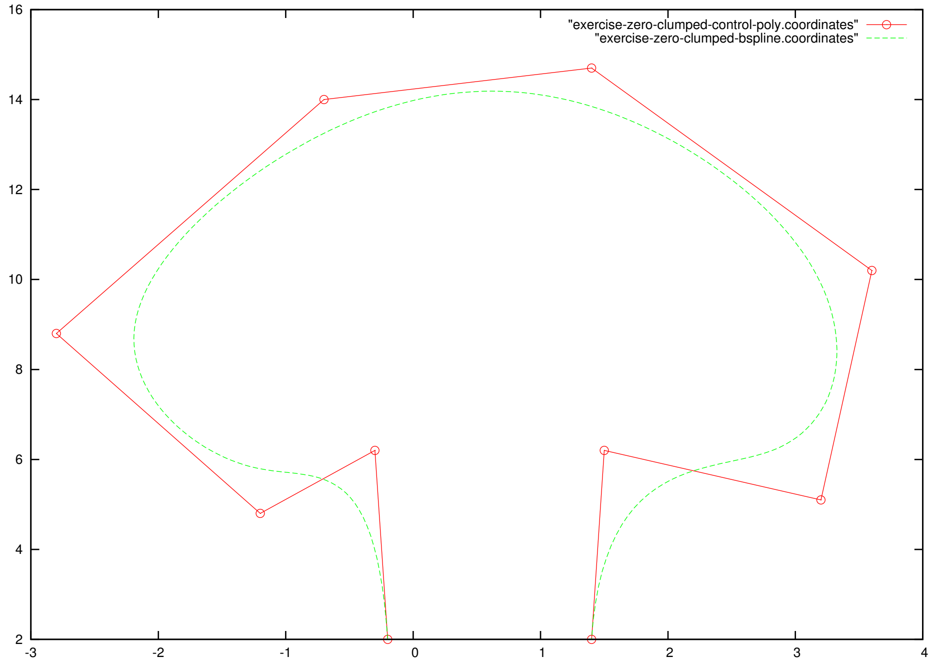

Figure 15: Mushroom from clumped knots partition

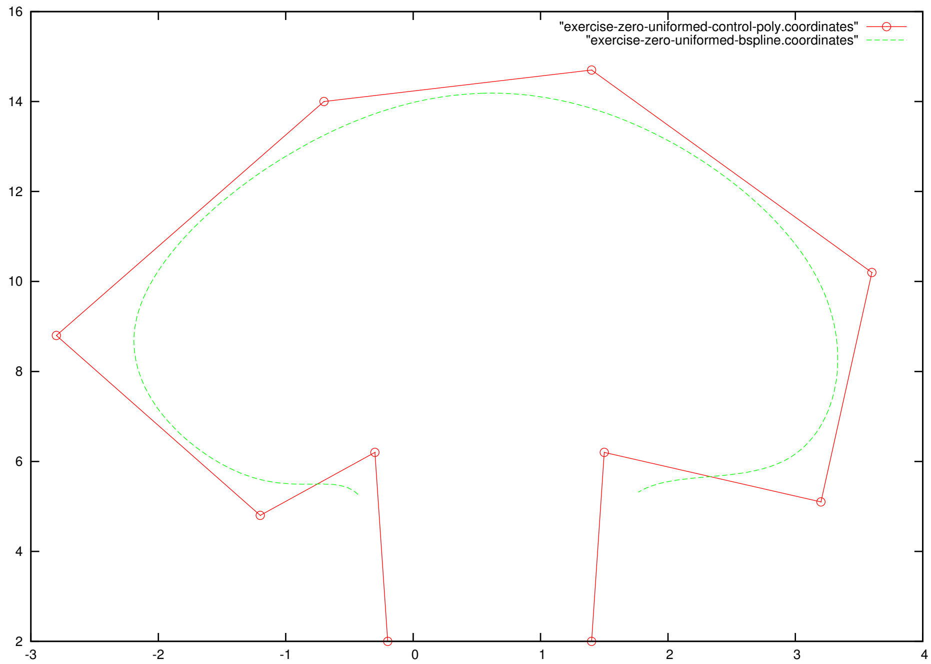

Figure 16: Mushroom from uniformed knots partition

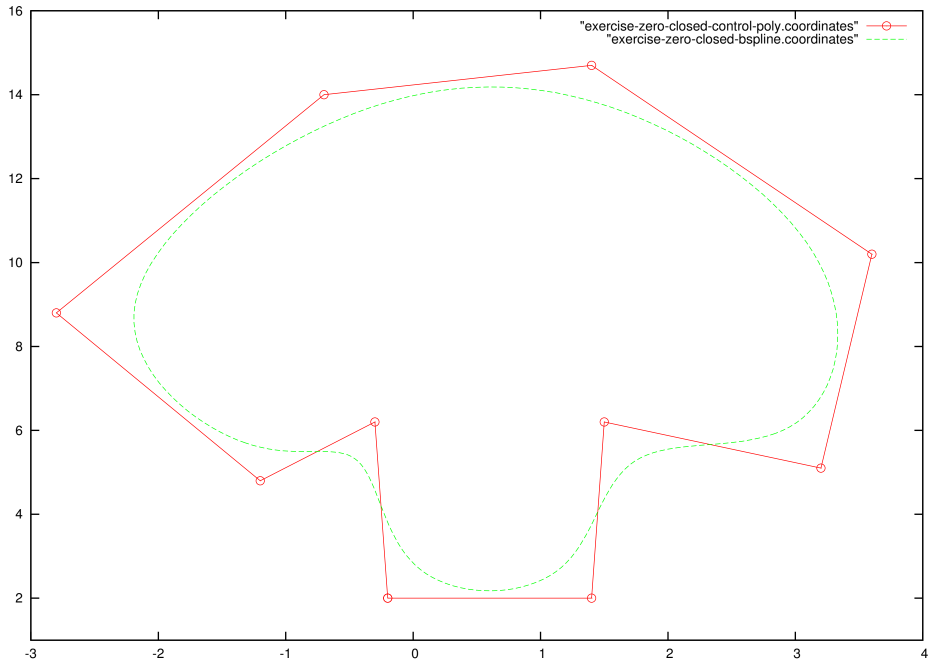

Figure 17: Mushroom from cyclic knots partition

2.2 *Increasing order k while decreasing* continuity *vector*

As Exercise 1 requires, in Figure 18 we report five BSpline curves, each one of them drawn against the same control net. We modify for each curve its knots partition, formally for curves $\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3, \mathbf{c}_4$ and $\mathbf{c}_5$ the following extended partitions are used:

$$\begin{aligned}
\Delta_1 &= \{0, 0, \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}, 1, 1\} \\
\Delta_2 &= \{0, 0, 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1, 1, 1\} \\
\Delta_3 &= \{0, 0, 0, 0, \frac{1}{3}, \frac{2}{3}, 1, 1, 1, 1\} \\
\Delta_4 &= \{0, 0, 0, 0, 0, \frac{1}{2}, 1, 1, 1, 1, 1\} \\
\Delta_5 &= \{0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1\}
\end{aligned}$$

respectively, and each curve $\mathbf{c}_i$ has *degree* i. As we can see, each knots partition is clumped, so the first and last control points are interpolated and the sequence of curves uses partitions with knots 0 and 1 repeated one more time in successive partitions. For lower degree curves, such as $\mathbf{c}_1$, they are quite next to control net, in particular $\mathbf{c}_1$ is the control net itself, while curve $\mathbf{c}_2$ is tangent to control net in two segments since it has *order* 3 and for $t \in [\frac{1}{4}, \frac{1}{2}) = [t_4, t_5)$ curve $\mathbf{c}_2$ has support $[t_2, t_3, t_4]$ where knot 0 has multiplicity 2, hence $\frac{\partial \mathbf{c}_2(t)}{\partial t}$ lies on direction given by $\mathbf{V}_3 - \mathbf{V}_2$. The same reasoning can be done for knot 1. Finally, the very last partition allow to draw a *Bezier* curve (the yellow one).

2.3 *Knots' multiplicities against the same control polygon*

In Figure 19 we report plots required by Exercise 2. This composite picture contains 5 curves according the following specifications, from top to bottom:

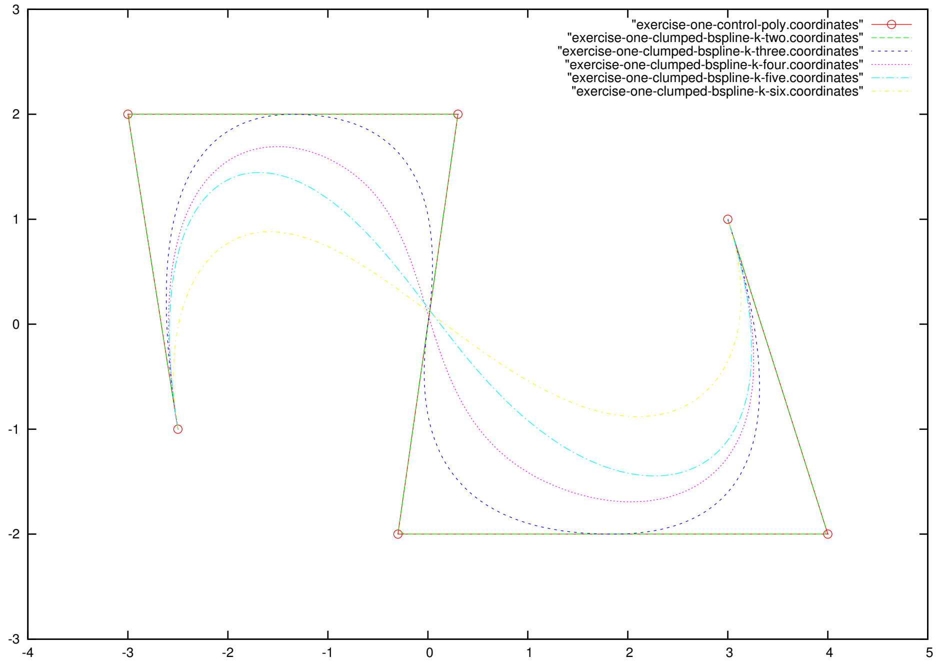

Figure 18: Decreasing the *continuity* vector collapsing in a Bezier

order	knots partition
4	$\{0^4, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$
4	$\{0^4, 1^2, 2^2, 3^2, 4^4\}$
6	$\{0^6, 1, 2, 3, 4, 5^6\}$
6	$\{0^6, 1, 2, 3^2, 4^6\}$
8	$\{0^8, 1, 2, 3^8\}$

The curve relative to the first row interpolate the first control point since 0 has multiplicity 4 which is the curve's order, while the last control point isn't interpolated since knots partition ends uniformly.

The curve relative to the second row both interpolate the first control point due to the multiplicity k of 0 and is tangent to control net on the direction $\mathbf{V}_4 - \mathbf{V}_3$ since for $t \in [t_5, t_6]$ the support is $[t_2, t_3, t_4, t_5]$, where 0 has multiplicity $3 = order - 1$.

Curves relative to third and forth rows both interpolate first and last control points because of clumped partition, while the last one reduce to only two knots $1, 2$, toward Bezier representation.

2.4 *Linearity near doubled control point with a clumped partition*

In Figure 20 we plot curve required for Exercise 3: a quadratic curve ($order = 3$) with a doubled control point $\mathbf{V}_3 = \mathbf{V}_4 = (1, 1)$ over an extended partition $\Delta = \{0, 0, 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1, 1, 1\}$, a clumped one with no repeated knots.

Lets consider the subcurve $\mathbf{c}_5(t)$ for $t \in [t_5, t_6]$:

$$\mathbf{c}_5(t) = \sum_{i=3}^{5} \mathbf{V}_i N_{i,3}(t)$$

Since $\mathbf{V}_4 = \mathbf{V}_3$, for $t \in [t_5, t_6]$, $\mathbf{c}_5(t)$ must lie on the convex hull $\mathbf{V}_3, \mathbf{V}_4, \mathbf{V}_5$ and, for $t \in [t_4, t_5]$, $\mathbf{c}_5(t)$ must lie on the convex hull $\mathbf{V}_2, \mathbf{V}_3, \mathbf{V}_4$, it does follow that $\mathbf{c}_5(t_5)$ has to lie in their intersection, therefore vertex $\mathbf{V}_4$, interpolating it.

Moreover, $\mathbf{c}_5(t)$ has two linear segment between $\mathbf{V}_2, \mathbf{V}_3$ and $\mathbf{V}_4, \mathbf{V}_5$, both of them join in t_4, t_6 with continuity $\mathcal{C}^1$.

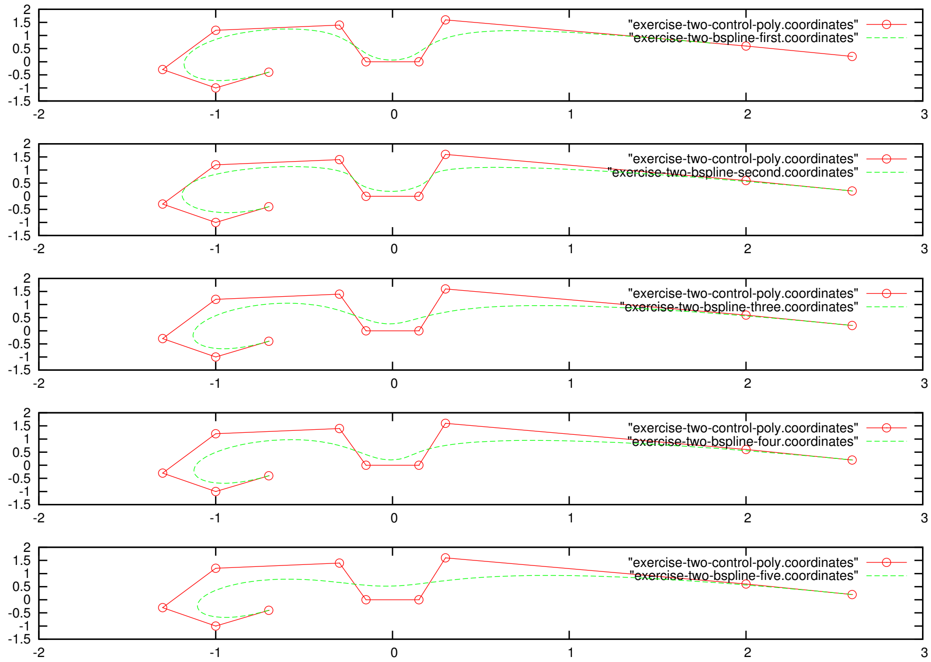

Figure 19: Knots's multiplicities increased against the same control polygon

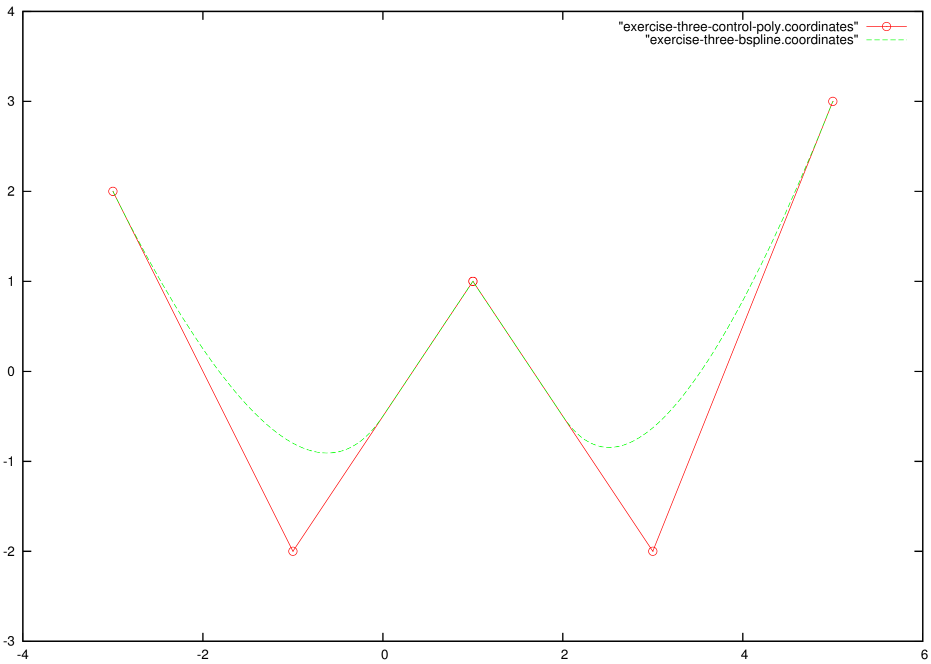

Figure 20: Linearity toward $(1, 1)$ due to "linearized" convex hull

2.5 *Two clumped partitions, different* continuity *vectors and same control polygon*

In Figure 21 we report two curves required by Exercise 4. Both of them have *order* $= 4$ and a control net with a doubled point $\mathbf{V}_3 = \mathbf{V}_4$, while the former (colored green) is over a knots partition $\Delta_1 = \{0, 0, 0, 0, \frac{1}{4}, \frac{3}{4}, 1, 1, 1, 1\}$ (having simple knots, no repeated one), the latter (colored blue) is over a knots partition $\Delta_2 = \{0,0,0,0,\frac{1}{2},\frac{1}{2},1,1,1,1\}$ (having knot $\frac{1}{2}$ doubled). Let define the following subcurves of the former:

$$\mathbf{c}_4(t) = \sum_{i=1}^{4} \mathbf{V}_i N_{i,4}(t), \quad t \in [t_4, t_5]$$
$$\mathbf{c}_5(t) = \sum_{i=2}^{5} \mathbf{V}_i N_{i,4}(t), \quad t \in [t_5, t_6]$$
$$\mathbf{c}_6(t) = \sum_{i=3}^{6} \mathbf{V}_i N_{i,4}(t), \quad t \in [t_6, t_7]$$

As we saw during classes [3], $\mathbf{c}_5'(t_5) = \mathbf{c}_4'(t_5)$ holds, so the curve interpolate a point on the segment with extrema $\mathbf{V}_2, \mathbf{V}_3$ and it is tangent to the control net in t_5. On the other hand, $\mathbf{c}_6'(t_6) = \mathbf{c}_5'(t_6)$, so the curve interpolate a point on the segment with extrema $\mathbf{V}_4, \mathbf{V}_5$ and it is tangent to the control net in t_6. Finally, observe that the curve doesn't interpolate the doubled control point $\mathbf{V}_3 = \mathbf{V}_4$.

The previous argument doesn't apply to the second curve because its knots partition is not simple, it contains a double knot.

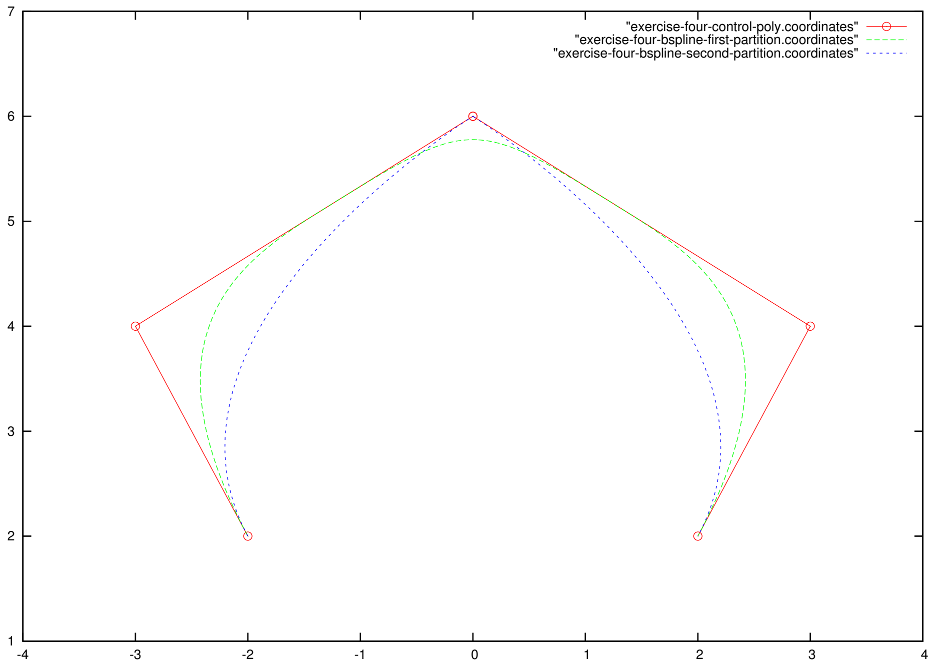

Figure 21: Same control polygon with doubled $(0,6)$ againts two clumped partitions

2.6 *Increasing clumped partitions for increasing occurrences of a control point*

In Exercise 5 it is required to provide three extended knots partitions in order to draw cubic curves reported in Figure 22. Each curve is defined against the "same" control net, having the middle vertex of the top row repeated one, two, three times respectively. Our choice of knots partitions, each clumped since interpolation of first and last control points is requested:

$$\Delta_1 = \{0,0,0,0,\frac{1}{2},1,1,1,1\}$$
$$\Delta_2 = \{0,0,0,0,\frac{1}{3},\frac{2}{3},1,1,1,1\}$$
$$\Delta_3 = \{0,0,0,0,\frac{1}{4},\frac{1}{2},\frac{3}{4},1,1,1,1\}$$

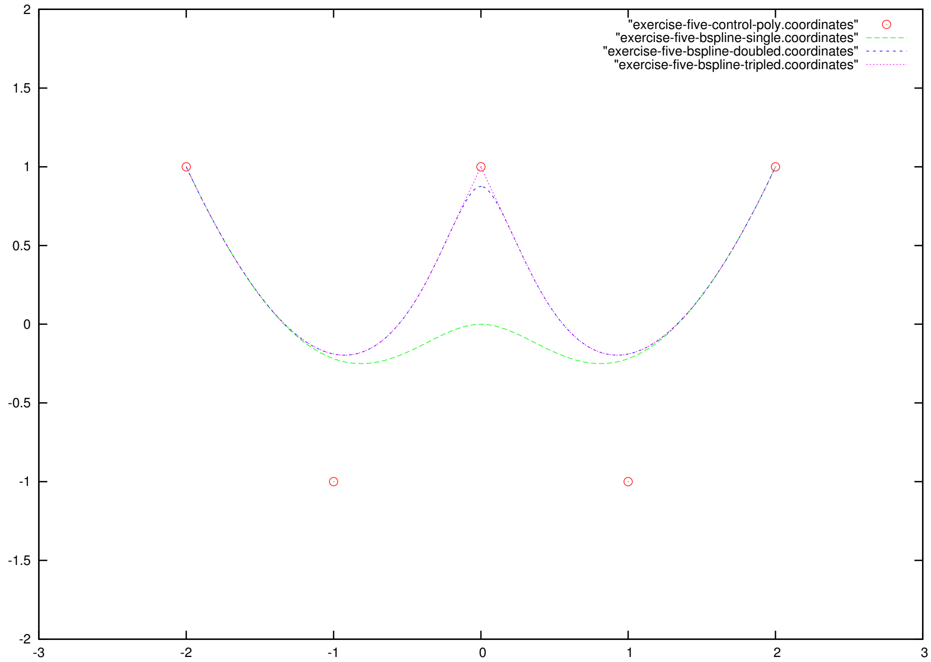

Figure 22: Three B-Splines, each for $1, 2, 3$ occurrences of $(0, 1)$ respectively

2.7 *Two B-Splines from two closed partitions*

In this last section we report two closed curves having $order = 4$ over two cyclic knots partitions, in Figure 23 and Figure 24, respectively.

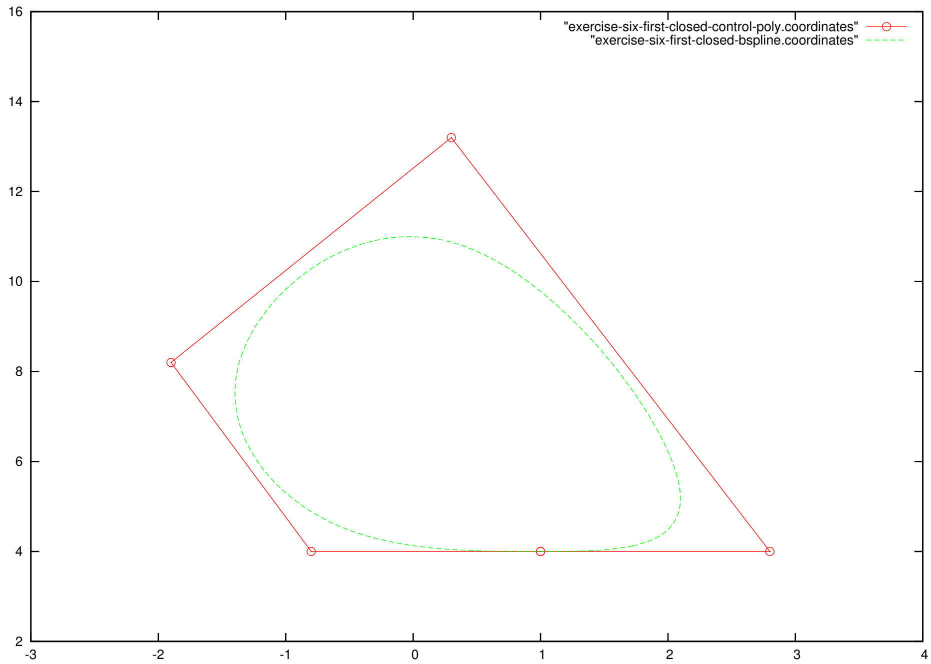

Figure 23: A first B-Spline from a closed partition

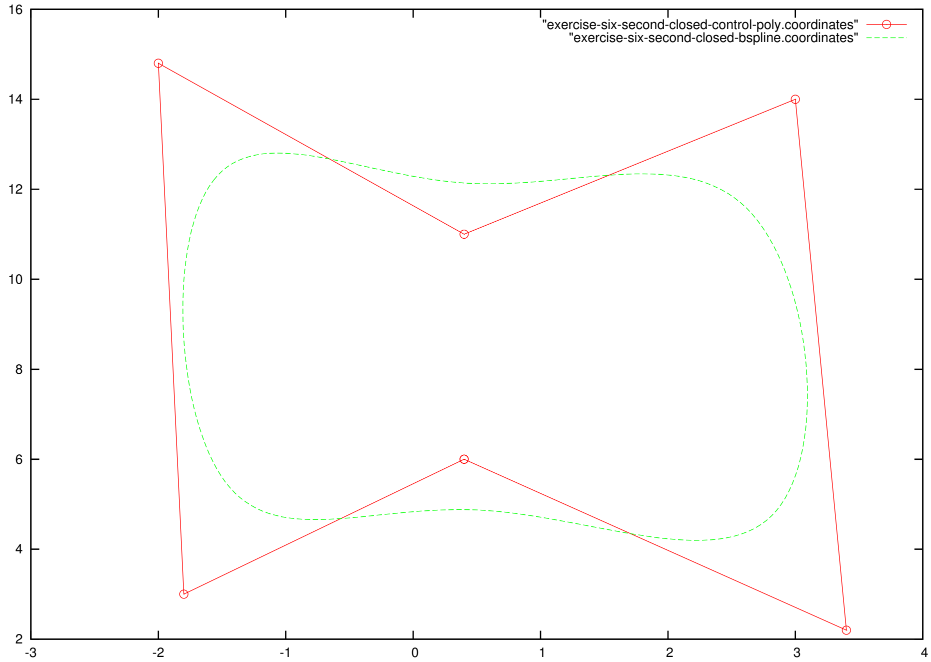

Figure 24: A second B-Spline from a closed partition

REFERENCES

[1] Open Source Project, *Julia language*, `http://julialang.org/`

[2] Gerald Farin, *Curves and surfaces for CAGD, Fifth Edition*

[3] Costanza Conti, *Lecture notes*, distributed during classes