Projet $n°2$ - 3 séances

Courbes de Bézier et typographie

1 Un peu d'histoire

Les courbes de Bézier sont des courbes polynomiales paramétriques décrites pour la première fois en 1962 par Pierre Bézier (ingénieur Arts et Métiers et Supélec à la régie Renault dans les années 1950) qui les utilisait pour concevoir des pièces d'automobiles.
Depuis les années 1980, les courbes de Bézier sont aussi utilisées pour l'affichage numériques des textes sur les écrans. Les fontes Postscript Type 1 élaborées par John Warnock [1] utilisent des courbes de Bézier d'ordre 3 (ou des Bézigons, voir plus loin), les polices Truetype [2] des courbes de Bézier d'ordre 2. La vidéo #Automaths 14 vous en apprendra plus sur ce sujet. Le grand avantage de cette méthode par rapport à celle classique de stockage des données de chaque pixels est de pouvoir créer des formes qui restent lisses même en zoomant beaucoup, et ce sans stocker beaucoup de données. Sur la figure 1 on peut par exemple observer la différence entre des lettres e dessinées à partir de la méthode classique de stockage de données et un chiffre cinq dessiné en utilisant des Bézigons.
La littérature dans ce domaine est très vaste, nous citerons par exemple les livres [DP98] ou [HubHub06], ou encore le livre [Don86].

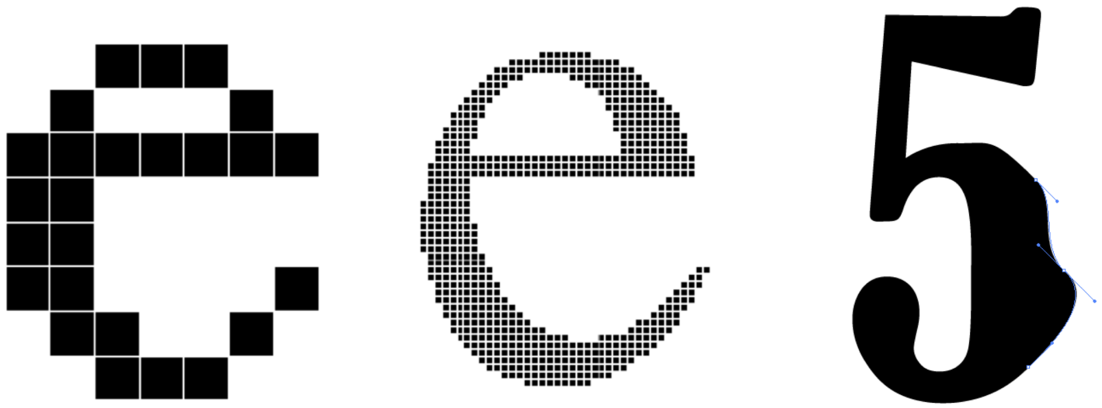

FIGURE 1 – Deux "e" classiques avec différents niveaux de résolution. Un 5 déformé avec AdobeIllustrator. D'après Frédéric Vivien.

2 Courbes de Bézier

2.1 Définition et interprétation géométrique

Les courbes de Bézier sont définies à partir d'une famille de $n+1$ points $\mathbf{c}_0, \cdots, \mathbf{c}_n \in \mathbb{R}^2$, appelés points de contrôle (avec $n \geq 1$). A partir de cette famille et des polynômes de Bernstein (les B_k^n ci-dessous), on construit la courbe de Bézier associée de la manière suivante.

1. qui fondera l'entreprise Adobe.
2. crées plus tard par Apple.

Définition 1

On appelle courbe de Bézier de degré $n \geq 1$, associée aux points de contrôle $\mathbf{c}_0, \cdots, \mathbf{c}_n \in \mathbb{R}^2$, la courbe paramétrée $\gamma_{\mathbf{c}}^n : [0,1] \to \mathbb{R}^2$ vérifiant

$$\forall t \in [0,1] \quad \gamma_{\mathbf{c}}^n(t) = \sum_{k=0}^{n} B_k^n(t)\, \mathbf{c}_k \text{ où } B_k^n(t) = \binom{n}{k} t^k (1-t)^{n-k}.$$

Les points de contrôles suivants permettent respectivement d'afficher une lettre U et une lettre e :

i	0	1	2	3	4	5
$c_i[0]$	1	1	1.1	3.9	4	4
$c_i[1]$	8	0	-0.5	-0.5	0	8

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$c_i[0]$	1	2	2	3	3	3	2	1	1	1	1	1	2	3	3	3
$c_i[1]$	2	2	2	2	3	3	3	3	3	2	1	1	1	1	1	1.5

Les lettres obtenues sont représentées sur la figure 2 suivante.

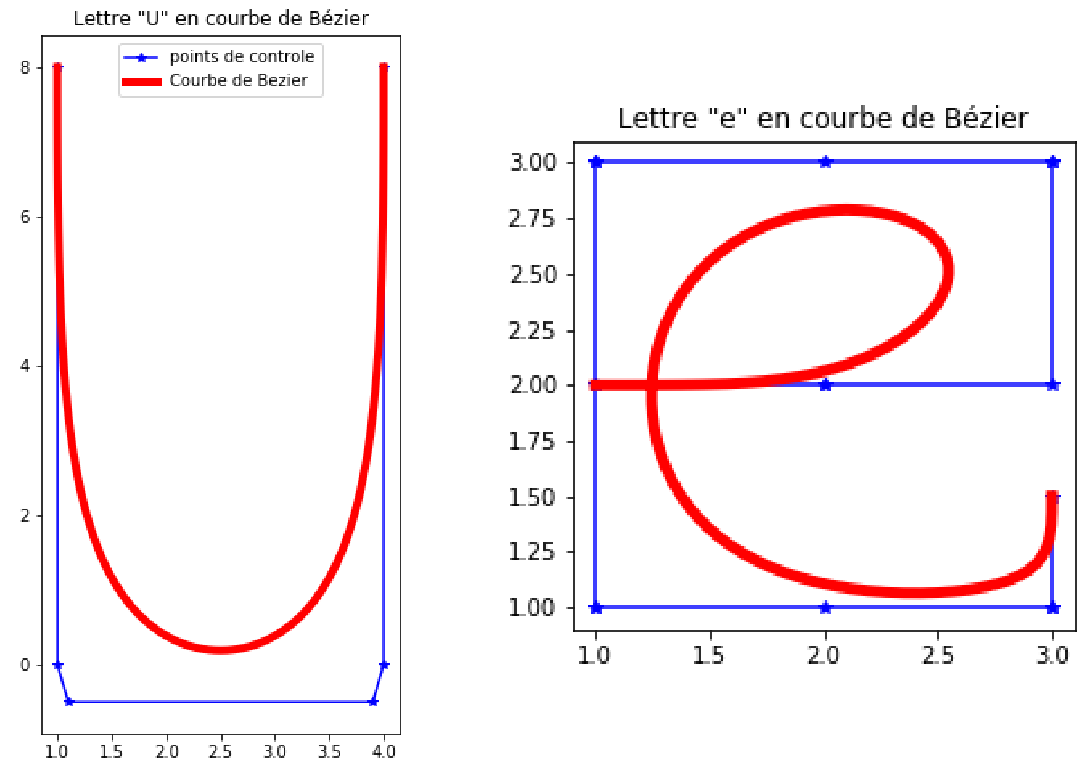

FIGURE 2 – Tentatives de tracés de lettres avec des courbes de Bézier

Question 1: Avec Python

a. Créer une fonction Python qui étant donné n points de contrôle et un vecteur de discrétisation de l'intervalle $[0,1]$, renvoie un tableau ("array") contenant abscisses et ordonnées de la courbe de Bézier aux "temps" du vecteur de discrétisation donné.

b. Pour les listes de points de contrôle ci-dessus, afficher la courbe de Bézier et le polygone de contrôle (union des segments $[c_j, c_{j+1}]$) associés de manière à reproduire les lettres ci-dessus.

2.2 Propriétés des courbes de Bézier

Il est utile d'étudier les propriétés mathématiques de ces courbes avant de passer à leur utilisation.

Question 2: Avec Python

Vérifier les propriétés suivantes sur un (ou des) ensemble(s) de points de contrôle de votre choix (par exemple ceux donnant les lettres "U" et "e" ci-dessus).

a. La courbe de Bézier $\gamma^n_{\mathbf{c}}$ associée aux points $\mathbf{c}_0, \dots, \mathbf{c}_n$ passe par les points $\mathbf{c}_0$ et $\mathbf{c}_n$, mais pas, en général, par les points $\mathbf{c}_1, \dots, \mathbf{c}_{n-1}$.

b. Le segment $[\mathbf{c}_0, \mathbf{c}_1]$ est tangent à la courbe $\gamma^n_{\mathbf{c}}$ en $\mathbf{c}_0$, et le segment $[\mathbf{c}_{n-1}, \mathbf{c}_n]$ est tangent à la courbe en $\mathbf{c}_n$. Plus précisément, les vecteurs vitesse $(\gamma^n_{\mathbf{c}})'$ en 0 et en 1 sont donnés par $(\gamma^n_{\mathbf{c}})'(0) = n(\mathbf{c}_1 - \mathbf{c}_0)$ et $(\gamma^n_{\mathbf{c}})'(1) = n(\mathbf{c}_n - \mathbf{c}_{n-1})$.

c. L'ordre dans lequel on considère les points $\mathbf{c}_i$ est très important. Si l'on échange deux points, on modifie complètement l'allure de la courbe de Bézier.

d. Si on modifie un des points de contrôle, toute la courbe de Bézier est modifiée.

Question 3: À l'écrit

a. Montrer qu'une courbe paramétrée de Bézier est de classe C^∞.

b. Démontrer la première partie de la propriété *a.* de la question précédente : la courbe de Bézier $\gamma^n_{\mathbf{c}}$ passe par les points $\mathbf{c}_0$ et $\mathbf{c}_n$

c. * Démontrer la propriété *b.* de la question précédente.

2.3 Théorème de Bernstein

Ce théorème traduit l'idée que si l'on est prêt à prendre beaucoup de points de contrôle, on peut toujours approcher une courbe continue par une courbe de Bézier, aussi bien que souhaité. Si on considère une courbe continue $g : [0,1] \to \mathbb{R}^2$, alors la courbe de Bézier associée aux points de contrôle équirépartis $\left(g\left(\frac{k}{n}\right)\right)_{0 \le k \le n}$ converge vers g lorsque n tends vers l'infini.

Pour le montrer rigoureusement, il faut écrire les composantes de $g : t \mapsto \big(g_1(t), g_2(t)\big)$ et appliquer le théorème suivant à g_1 et g_2 :

Théorème 1 (*Théorème de Bernstein*)

Soit f une fonction continue sur $[0,1]$ à valeur dans $\mathbb{R}$. La suite de fonctions polynomiales $(B_n(f))_{n \ge 1}$ définie par :

$$\forall n \ge 1,\ \forall t \in [0,1], \quad B_n(f)(t) = \sum_{k=0}^{n} B^n_k(t) f\left(\frac{k}{n}\right),$$

converge uniformément vers f sur $[0,1]$ quand $n \to \infty$, c'est-à-dire que

$$\sup_{t \in [0,1]} |f(t) - B_n(f)(t)| \xrightarrow[n \to \infty]{} 0.$$

Avant de donner le schéma de la preuve, on va étudier un peu plus en détail les polynômes de Bernstein de manière à mieux comprendre la preuve.

Question 4: À l'écrit

a. Montrer pour tout $t \in [0,1]$ l'identité $\displaystyle\sum_{k=0}^{n} B_k^n(t) = 1$. Pourquoi cela permet pour tout t de dire que $B_n(f)(t)$ est une moyenne pondérée [3] des points de contrôle ?

b. Il y a un lien entre les polynômes de Bernstein et les lois binomiales. Préciser ce lien.

c. En utilisant l'espérance et la variance des lois binomiales, montrer que

$$\sum_{k=0}^{n} \frac{k}{n} B_k^n(t) = t, \quad \sum_{k=0}^{n} \left(\frac{k}{n} - t\right)^2 B_k^n(t) = \frac{t(1-t)}{n}. \tag{1}$$

d. On fixe maintenant $t = 0,3$ et $n = 100$. Le lien `https://www.geogebra.org/m/FcTaeADw` permet de visualiser les probabilités associées à la loi binomiale de paramètres n, t, à savoir les $\left(B_k^n(t)\right)_{0 \le k \le n}$. Pour quelle valeur de k obtient-on un coefficient maximal ? Faire lien avec la première égalité de (1). Pour quelles valeurs de k/n obtient-on des coefficients $B_k^n(t)$ supérieurs à un centième du coefficient maximal ? Reprendre cette question avec la nouvelle valeur $n = 500$.

Eléments de démonstration du théorème de Bernstein sous l'hypothèse $f \in \mathcal{C}^1([0,1])$. Cette hypothèse plus restrictive simplifie la preuve.

a. On remarque que $\displaystyle f(t) - B_k^n(f)(t) = \sum_{k=0}^{n} \left(f(t) - f\left(\frac{k}{n}\right)\right) B_k^n(t).$

b. Soit $M = \sup_{t \in [0,1]} |f'(t)|$. On montre que $\displaystyle |f(t) - B_k^n(f)(t)| \le M \sum_{k=0}^{n} \left|t - \frac{k}{n}\right| B_k^n(t).$

c. En utilisant l'inégalité de Cauchy-Schwarz et la seconde égalité obtenue dans (1), on en déduit que pour tout $t \in [0,1]$,

$$|f(t) - B_n(f)(t)| \le \frac{M}{\sqrt{n}} \sqrt{t(1-t)} \le \frac{M}{2\sqrt{n}}. \tag{2}$$

L'inégalité de Cauchy-Schwarz est une inégalité classique en mathématiques et est détaillée dans la version qui nous intéresse sur la page `https://www.bibmath.net/dico/index.php?action=affiche&quoi=./h/holder.html`

Question 5: À l'écrit

a. Rédiger une preuve rigoureuse et complète des points a et b ci-dessus à partir des éléments donnés.

b. * Rédiger une preuve rigoureuse et complète du point c ci-dessus à partir des éléments donnés.

2.4 * Interprétation géométrique des courbes de Bernstein

P. De Casteljau (ingénieur chez Citroën) a inventé un algorithme géométrique pour dessiner les courbes de Bézier, bien entendu à partir des mêmes données (les points de contrôle). D'une part, c'est un algorithme efficace comme on le verra dans la question 7, d'autre part il donne une visualisation géométrique de ce qu'on calcule. On prend $t \in [0,1]$, et on définit par récurrence des points

$$\mathbf{c}_0^1(t) = (1-t)\mathbf{c}_0 + t\mathbf{c}_1, \cdots, \mathbf{c}_{n-1}^1 = (1-t)\mathbf{c}_{n-1} + t\mathbf{c}_n,$$

3. plus précisément un barycentre si vous connaissez ce terme.

puis

$$\mathbf{c}_0^2(t) = (1-t)\mathbf{c}_0^1(t) + t\mathbf{c}_1^1(t), \cdots, \mathbf{c}_{n-2}^2(t) = (1-t)\mathbf{c}_{n-2}^1(t) + t\mathbf{c}_{n-1}^1(t),$$

et ainsi de suite $\mathbf{c}_i^{m+1}(t) = (1-t)\mathbf{c}_i^m(t) + t\mathbf{c}_{i+1}^m(t)$ pour $m = 0, \ldots, n-1$ et $i = 0, \ldots, n-m$, jusqu'à arriver à un point unique

$$\delta_{\mathbf{c}}^n(t) = \mathbf{c}_0^n(t) = (1-t)\mathbf{c}_0^{n-1}(t) + t\mathbf{c}_1^{n-1}(t),$$

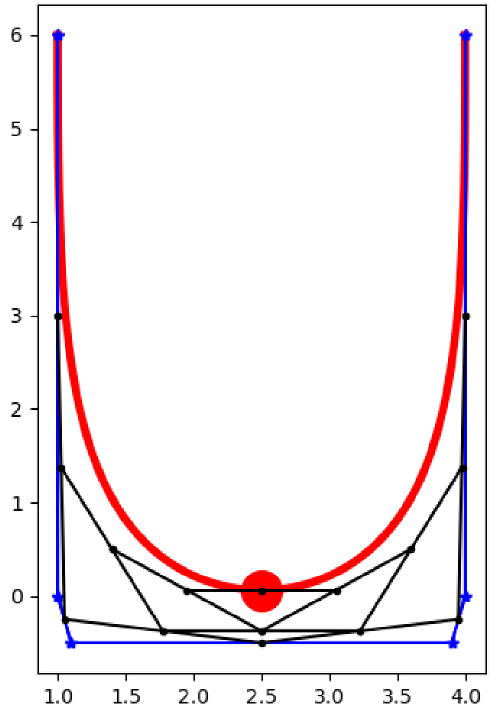

Figure 3 – Etapes de l'algorithme de Casteljau pour la lettre U précédente et pour $t = \frac{1}{2}$.

Question 6: À l'écrit

a. * Montrer que si $n = 3$, les courbes de Bezier $\gamma_{\mathbf{c}}^3$ et de De Casteljau $\delta_{\mathbf{c}}^3$ sont confondues.

b. * Montrer le cas général : quel que soit le nombre $n \in \mathbb{N}^*$ de points de contrôle, les courbes de Bezier $\gamma_{\mathbf{c}}^n$ et de De Casteljau $\delta_{\mathbf{c}}^n$ sont confondues.

Question 7: Avec Python

a. * Créer une fonction qui étant donné n points de contrôle et un vecteur de discrétisation de l'intervalle $[0, 1]$, utilise l'algorithme de Casteljau et renvoie un tableau ("array") contenant abscisses et ordonnées de la courbe de Bézier aux "temps" du vecteur de discrétisation donné.

b. * Tracer à l'aide de cet algorithme la courbe de Bézier associée à ces points de contrôle et le polygone de contrôle associé (union des segments $[c_j, c_{j+1}]$) dans le cas de la lettre U ou de la lettre e de la figure 2.

c. * Comparer la vitesse de ce nouveau code avec celui de la question 1 (utilisant les polynômes de Bernstein) dans le cas de la lettre e de la figure 2.

d. * Expliquer rapidement les différences de rapidité observées.

3 Application à la typographie informatique et extension de la méthode

3.1 Un exemple typographique

Comme mentionné dans l'introduction, les courbes de Bézier sont très utilisées en typographie informatique. On va s'intéresser dans cette section à la typographie des lettres grecques. Vous trouverez des exemples de polices grecques sur ce site[4]. Dans la question suivante on utilisera les points de contrôle donnés par la matrice c définie par

$$\mathbf{c} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & -2 & 3 & 0 \end{pmatrix}. \qquad (3)$$

Question 8: Avec Python

a. Représenter la courbe de Bézier associée aux points de contrôle ci-dessus. A quelle lettre grecque vous fait-elle penser ?

b. Modifier les points de contrôle (ajout, déplacement,...), pour que la partie inférieure droite de la lettre présente un arrondi, comme on peut le voir dans la police Abadi décrite sur le site mentionné plus haut. Justifier vos modifications.

Question 9: À l'écrit

De manière générale, si une lettre est dessinée grâce à la courbe de Bézier obtenue à partir des points de contrôle $\mathbf{c}_0, \ldots, \mathbf{c}_n \in \mathbb{R}^2$, quels nouveaux points de contrôle choisiriez-vous pour dessiner la même lettre :

a. 1 cm plus bas dans le document ?

b. trois fois plus grande ?

c. $\star$ tournée à 90 ?

3.2 Les Bézigons

Quand le nombre de points points de contrôle augmente, il devient difficile de prévoir par où la courbe de Bézier associée va passer. D'autre part, les lettres typographiques présentent souvent des discontinuités dans la dérivée. Une façon de résoudre ces problèmes est d'utiliser des courbes de Bézier par morceaux, que l'on appelle aussi Bézigons.

4. https ://pedagogie.ac-lille.fr/langues-cultures-antiquite/quelle-police-decriture-sur-ordinateur-pour-le-grec-ancien/

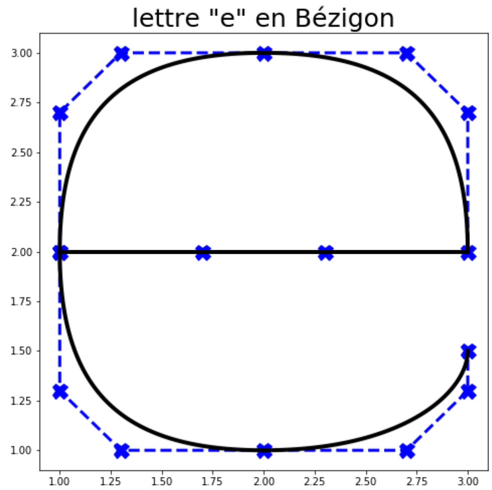

FIGURE 4 – Tentative de tracé de lettre "e" avec des Bézigon

Le Bézigon d'ordre 3 est couramment utilisé en typographie et par les logiciels de CAO [5].

Définition 2

Etant donné la famille de points de contrôle

$$\mathbf{c}_0, \cdots, \mathbf{c}_p, \mathbf{c}_{p+1}, \cdots, \mathbf{c}_{2p}, \cdots, \mathbf{c}_{(m-1)p}, \cdots, \mathbf{c}_{mp},$$

on appelle **Bézigon** *la courbe obtenue par recollement des m des courbes de Bézier de degré p, associées aux points de contrôle $\mathbf{c}_{ip}, \cdots, \mathbf{c}_{(i+1)p}$, pour $i = 0, \ldots, m-1$.*

Question 10: À l'écrit

a. Dans la figure 3.2, que valent p et m avec les notations de la définition 2 ?

b. Montrer qu'un Bézigon d'ordre p passe par les points $(\mathbf{c}_{ip})_{i=0,\ldots,m}$.

c. La courbe de la Figure 3.2 pourrait-elle être une courbe de Bézier ?

d. $\star$ A partir de la définition précédente un peu vague, proposer une paramétrisation précise d'un Bézigon, c'est-à-dire une fonction $\gamma : [0,1] \to \mathbb{R}^2$ dont l'image est bien la courbe recherchée. Montrer que cette paramétrisation γ est continue, et admet des dérivées à droite et à gauche en tout point. Donner explicitement les dérivées à droite et à gauche aux points $(\mathbf{c}_{ip})_{i=0,\ldots,m}$.

5. Conception Assistée par Ordinateur

Question 11: Avec Python

Pour les questions ci-dessous, vous choisirez entre courbes de Bézier ou Bézigons et justifierez votre choix. Vous expliquerez aussi comment vous avez choisi vos points de contrôle.

a. Créer une fonction Python qui étant donné un entier p, mp points de contrôles et un vecteur de discrétisation de $[0,1]$ renvoie un tableau ("array") contenant des abscisses et d'ordonnées de points qui discrétisent le Bézigon d'ordre p associé.

b. Donner un ensemble de points de contrôle pour tracer une lettre μ, et tracer la figure associée.

c. ⋆ Faites de même avec une autre lettre grecque de votre choix.

4 Routines Python pour résoudre ces problèmes

Pour la partie numérique de ce projet, vous devrez vous familiariser en Python avec la gestion des graphiques 2d et la manipulation des matrices. Voici quelques commandes qui vous seront particulièrement utiles.

- Extraction d'une sous matrice

```
import numpy as np
A=np.zeros((10,10))
# extraction des lignes 0 a 2 et des colonnes 2 a 4
B=A[:3,2:5]
```

- Duplication d'une matrice

```
import numpy as np
A=np.zeros((10,10))
B=np.copy(A)
```

  Attention, la commande B=A effectue un lien symbolique entre A et B. Si vous modifiez l'un, vous modifiez l'autre. Il faut utiliser la commande `np.copy`.

- Coefficients binomiaux

```
from scipy.special import binom
binom(n,k)
```

- Temps d'execution d'une commande

```
import time
start = time.clock()
# Mes commandes
end = time.clock()
print(end-start)
```

- Affichage dans un repère orthonormé (à ajouter après les commandes `plt.plot(...)`)

```
plt.axis('equal')
```

Références

[DP98] G. Demengel, J. P. Pouget, *Modèles de Béziers, de B-Splines et de NURBS : Mathématiques des courbes et surfaces*, Ellipse, 1998.

[HubHub06] F. Hubert, J. Hubbard : *Calcul scientifique. Tome 1.* Vuibert, 2006.

[Don86] R. Dony, *Calcul des parties cachées, approximation des courbes par la méthode de Bézier et des B-Splines*, Masson, 1986.

[Viv] Utilisation des courbes de Bézier en Infographie : Les police True Type et le Morphing. http://vivienfrederic.free.fr/irem/bezier.pdf