Université Claude Bernard Lyon 1
Agrégation interne de Mathématiques : Écrit
Année 2013–2014

III Courbes de Bézier

1° Polynômes de Bernstein : $B_{k,n}(t) = \binom{n}{k} t^k (1-t)^{n-k}$

a) Relations utiles

(i) partition de l'unité : $\displaystyle\sum_{k=0}^{n} B_{k,n}(t) = \sum_{k=0}^{n} \binom{n}{k} t^k (1-t)^{n-k} = (1+1-t)^n = 1$;

(ii) positivité : pour $t \in [0,1]$, $0 \leqslant B_{k,n}(t) \leqslant 1$: clair ;

(iii) symétrie : $B_{k,n}(t) = \binom{n}{k} t^k (1-t)^{n-k} = B_{n-k,n}(1-t)$ pour tout t et tous n, k ;

(iv) récurrence : $B_{k,n}(t) = (1-t)B_{k,n-1}(t) + tB_{k-1,n-1}(t)$ car $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$.

b) L'indépendance linéaire vient de ce que la valuation [1] de $B_{n,k}$ est k. Écrivons la matrice [2] de la famille $(B_{n,k})_{0\leqslant k\leqslant n}$ dans la base canonique $(t^k)_{0\leqslant k\leqslant n}$:

$$\begin{pmatrix} \binom{n}{n} & 0 & \cdots & 0 \\ * & \binom{n}{1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ * & \cdots & * & \binom{n}{n} \end{pmatrix}$$

Cette matrice est carrée, triangulaire supérieure avec des coefficients non nuls sur la diagonale donc elle est inversible : les $(B_{n,k})$ forment une base.

2° Courbes de Bézier

a) Invariance affine : conséquence de la préservation du barycentre par une application affine.
b) Enveloppe convexe : la courbe est dans l'enveloppe convexe des P_j car tous ses points sont barycentres des P_j avec des coefficients positifs ou nuls (propriété (ii)).
c) Influence de l'ordre des points : on voit sur le fichier joint que les courbes construites sur $[P_0, P_1, P_2, P_3]$ et $[P_1, P_0, P_2, P_3]$ sont bien différentes. En revanche, les courbes construites sur $[P_0, P_1, \ldots, P_n]$ et $[P_n, P_{n-1}, \ldots, P_0]$ coïncident grâce à la symétrie (propriété (iii)).
d) Contrôle pseudo-local : comme le polynôme $t^j(1-t)^{n-j}$ atteint son maximum en j/n (vérifier), le poids de P_j est maximal pour les valeurs de t proches de j/n ; autrement dit, si l'on bouge le point P_j, la courbe est modifiée « surtout » pour les valeurs de t au voisinage de j/n.
e) Interpolation aux extrémités : on a : $M(0) = P_0$, que $M(1) = P_n$.
Les coordonnées de $M(t)$ sont des polynômes en t donc dérivables. On a : $B_{k,n}(t) = \binom{n}{k} t^k + o(t^k)$, si bien que $B_{1,n}(t) = nt + o(t)$ et $B_{k,n}(t) = o(t)$ si $k \geqslant 2$; d'autre part, $B_{0,n}(t) = 1 - nt + o(t)$. Il vient [3] : $M(t) = P_0 + nt\overrightarrow{P_0P_1} + o(t)$, ce qui entraîne que $M'(0) = \overrightarrow{P_0P_1}$. Autrement dit, pour autant que P_0 et P_1 diffèrent, la courbe est tangente à (P_0P_1) en son extrémité.

1. La valuation $\mathrm{val}(P)$ d'un polynôme est le degré minimal d'un monôme non nul, ou $+\infty$ pour le polynôme nul. C'est aussi la multiplicité de 0 comme racine de P. On a : $\mathrm{val}(P+Q) \geqslant \min(\mathrm{val}(P), \mathrm{val}(Q)$ et $\mathrm{val}(PQ) = \mathrm{val}(P) + \mathrm{val}(Q)$ si P, Q sont deux polynômes quelconques.

2. Selon une convention naturelle mais peut-être pas universelle, la matrice d'une famille $\mathfrak{f}$ dans une base $\mathfrak{e}$ a pour colonnes les coordonnées des vecteurs de $\mathfrak{f}$ dans la base $\mathfrak{e}$.

3. En coordonnées dans un repère quelconque, cela s'écrit : $x(t) = x_0 + nt(x_1 - x_0) + o(t)$, idem pour y.

3° Algorithme de Casteljau

À $t \in [0,1]$ fixé, on définit $M_{j,0} = P_j$ pour $0 \leqslant j \leqslant n$ puis, à l'étape $\ell \in \{1, \dots, n\}$:

$$M_{j,\ell} = M_{j,\ell}(t) = \begin{bmatrix} M_{j,\ell-1}(t) & M_{j+1,\ell-1}(t) \\ 1-t & t \end{bmatrix} \quad (0 \leqslant j \leqslant n-\ell).$$

a) Relation fondamentale : par associativité du barycentre, le point

$$\begin{bmatrix} \begin{bmatrix} P_0 & \cdots & P_n \\ B_{0,n}(t) & \cdots & B_{n,n}(t) \end{bmatrix} & \begin{bmatrix} P_1 & \cdots & P_{n+1} \\ B_{0,n}(t) & \cdots & B_{n,n}(t) \end{bmatrix} \\ 1-t & t \end{bmatrix}$$

est le barycentre de $P_0, \dots, P_n$, où P_k $(0 \leqslant k \leqslant n+1)$ est affecté du coefficient :

$$(1-t)B_{k,n}(t) + tB_{k-1,n}(t) = B_{k,n+1}(t),$$

l'égalité venant de la relation (iv) : c'est bien $\begin{bmatrix} P_0 & \cdots & P_{n+1} \\ B_{0,n+1}(t) & \cdots & B_{n+1,n+1}(t) \end{bmatrix}$.

b) Validité de l'algorithme de Casteljau : montrons par récurrence sur n que $M_{0,n}(t) = M(t)$ pour tout t. On fixe t. Si $n = 1$, il n'y a rien à démontrer. Fixons n. On suppose la relation vraie pour tout système à n points et on se donne $P_0, \dots, P_{n+1}$. On l'applique à $[P_0, \dots, P_n]$ et à $[P_1, \dots, P_{n+1}]$ pour obtenir [4] :

$$\begin{bmatrix} P_0 & \cdots & P_n \\ B_{0,n}(t) & \cdots & B_{n,n}(t) \end{bmatrix} = M_{0,n}(t) \quad \text{et} \quad \begin{bmatrix} P_1 & \cdots & P_{n+1} \\ B_{0,n}(t) & \cdots & B_{n,n}(t) \end{bmatrix} = M_{1,n}(t).$$

En comparant la définition de $M_{0,n+1}(t) = \begin{bmatrix} M_{0,n}(t) & M_{1,n}(t) \\ 1-t & t \end{bmatrix}$ et la relation fondamentale a), on conclut : $M_{0,n+1}(t)$ est bien le point courant de la courbe sur $[P_0, \dots, P_{n+1}]$.

c) Construction effective : voir fichier Geogebra.

Construction :
- dans la fenêtre « géométrie », créer un curseur `t` variant entre 0 et 1 ;
- afficher le tableur et passer dans la fenêtre « tableur » ;
- créer les $n+1$ points $P_0 = M_{0,0}, \dots, P_n = M_{n,0}$ dans la colonne `A`, cases `A1`,...,`A(n+1)` ;
- définir en `B1` le point $M_{0,1}$ par : `t*A1+(1-t)*A2` ;
- tirer la case `B1` vers le bas jusqu'à la case `Bn` ;
- tirer la sous-colonne `B1:B(n-1)` d'une colonne vers la droite ;
- tirer la sous-colonne `C1:C(n-2)` d'une colonne vers la droite ;
- recommencer jusqu'à ce qu'à n'avoir plus qu'une case, disons `X1` ;
- dans la fenêtre « géométrie », tracer les segments entre `Ak` et `A(k+1)`, `Bk` et `B(k+1)`, etc.
- tracer le lieu de `X1` lorsque `t` varie.

On se retrouve avec des points qui implantent les $M_{i,j}$, disposés ainsi ($M_{n,n}$ en `X1`) :

$$\begin{matrix} M_{0,0} & M_{0,1} & \cdots & \cdots & M_{n,n} \\ M_{1,0} & M_{1,1} & \cdots & M_{n-1,n-1} & \\ \vdots & \vdots & \iddots & & \\ M_{n-1,0} & M_{n-1,1} & & & \\ M_{n,0} & & & & \end{matrix}$$

Il est instructif de bouger les points de contrôle `Ak` pour expérimenter des permutations non triviales et le « contrôle pseudo-local » ; de dézoomer et faire varier le curseur `t` sur un intervalle plus grand, typiquement $[-10, 10]$.

4. Cela aide de regarder le triangle de points de l'énoncé !

d) Recollement : il s'agit de permuter les deux sommes.[5] On écrit :

$$\begin{aligned}
N(u) &= \sum_{k=0}^{n}\sum_{\ell=0}^{k}\binom{n}{k}\binom{k}{\ell}(1-u)^{n-k}u^k(1-t)^{k-\ell}t^\ell P_\ell \\
&= \sum_{\ell=0}^{n}\sum_{k=\ell}^{n}\frac{n!}{k!(n-k)!}\frac{k!}{\ell!(k-\ell)!}(1-u)^{n-k}u^k(1-t)^{k-\ell}t^\ell P_\ell \quad \text{en permutant} \\
&= \sum_{\ell=0}^{n}\sum_{j=0}^{n-\ell}\frac{n!}{j!\ell!(n-\ell-j)!}(1-u)^{j}u^{n-j}(1-t)^{n-\ell-j}t^\ell P_\ell \quad (j=n-k) \\
&= \sum_{\ell=0}^{n}\frac{n!}{\ell!(n-\ell)!}\sum_{j=0}^{n-\ell}\frac{(n-\ell)!}{j!(n-\ell-j)!}(1-u)^{j}u^{n-\ell-j}(1-t)^{n-\ell-j}u^\ell t^\ell P_\ell \\
&= \sum_{\ell=0}^{n}\binom{n}{\ell}\big(1-u+u(1-t)\big)^{n-\ell}(ut)^\ell P_\ell = \sum_{\ell=0}^{n}\binom{n}{\ell}(ut)^\ell(1-ut)^{n-\ell}P_\ell.
\end{aligned}$$

Reste à voir que cette relation traduit bien le dessin : repérer les $M_{0,j}$ sur ledit dessin.

e) Application : à t fixé, grâce au recollement, on sait que la tangente en $M(t)$ est la tangente à la courbe $u \mapsto N(u)$ en $u = 1$. Par 2°e), c'est la droite reliant $N(1) = M(t)$ à l'avant dernier point de contrôle, $M_{0,n-1}(t)$. Par symétrie, c'est aussi la droite reliant $M(t)$ et $M_{1,n-1}(t)$). Comme ces trois points sont alignés, c'est la droite $(M_{0,n-1}(t)M_{1,n-1}(t))$ (génériquement bien définie...).

4° Points de contrôle pour une courbe polynomiale

Soit une courbe $M(t) = (x(t), y(t))$ $(t \in [0,1])$ avec x et y polynomes de degré $\leqslant n$. Fixons $n+1$ points $P_j = (x_j, y_j)$. Alors, M est la courbe de Bézier sur ces points de contrôle SSI

$$\forall t \in [0,1], \quad \begin{cases} x(t) = \sum_{k=0}^{n} B_{k,n}(t)x_k \\ y(t) = \sum_{k=0}^{n} B_{k,n}(t)y_k. \end{cases}$$

Comme $[0,1]$ est infini, il revient au même d'imposer des égalités de polynômes, *i.e.* de prendre pour t une indéterminée. Comme les $(B_{k,n})_{0\leqslant k\leqslant n}$ sont une base de $\mathbb{R}_n[t]$, la données de x et y offre un unique choix des x_k et des y_k.

5° Intérêt de la construction

Les courbes de Bézier et variations sont utilisées en CAO et pour le dessin vectoriel.

Premier intérêt, la compacité des données : au lieu de retenir les coordonnées de tous les points de la courbe, on ne retient que celles des points de contrôle.

Deuxième intérêt – lié à l'idée majeure du dessin vectoriel[6] : l'invariance affine permet de faire des dessins ayant la même précision quel que soit l'agrandissement. On calcule la courbe *après* avoir fixé l'agrandissement, si bien qu'elle paraît lisse et qu'on n'a pas d'effet de pixellisation, même avec des zooms importants.

Troisisème intérêt : la manipulation à la souris des points de contrôle permet de faire aisément (grâce au « contrôle pseudo-local ») des courbes qui ont la forme qu'on imagine.

5. Cela devrait être un réflexe, c'est la seule chose qu'on sait faire...

6. Les formats d'image PDF (`.pdf`), SVG, postscript (`.ps` ou `.eps`) sont adaptés au dessin vectoriel. Documentez-vous aussi sur les *polices (de caractères) vectorielles.*

Et dans les logiciels courants ?
En pratique, lorsqu'on manipule des « chemins » dans les logiciels de dessins (*GIMP*, *Inkscape* ou l'ignoble[7] *Photosh..*), on manipule des *splines* : ce sont des courbes voisines des courbes de Bézier, déterminées par deux points de contrôle et les tangentes en ces deux points.
Plus précisément, on se donne deux points $P_0 = (x_0, y_0)$ et $P_1 = (x_1, y_1)$ et deux vecteurs $\overrightarrow{T_0} = (u_0, v_0)$ et $\overrightarrow{T_1} = (u_1, v_1)$ (représentés par les « ancres » dans les logiciels) ; on cherche une courbe $t \mapsto (x(t), y(t))$ telle que

$$\begin{cases} (x(0), y(0)) = P_0, \\ (x'(0), y'(0)) = \overrightarrow{T_0} \\ (x(1), y(1)) = P_1, \\ (x'(1), y'(1)) = \overrightarrow{T_1} \end{cases}$$

On a donc 4 équations pour chaque coordonnée, on détermine donc 2 polynômes de degré $\leqslant 3$ avec ces données. Cela permet de recoller des morceaux de courbes et d'obtenir des courbes de classe $\mathscr{C}^2$ (je crois... vérifier... pas si sûr...) auxquelles on peut, si on préfère, imposer des points anguleux ici ou là.

Mathématiquement, le contenu du paragraphe précédent tient dans la répétition de la propriété suivante (une fois par coordonnée) : étant donnés (x_0, u_0, x_1, u_1), il existe un unique polynôme de degré $\leqslant 3$, disons $x(t) = a_3t^3 + a_2t^2 + a_1t + a_0$, tel que

$$\begin{cases} x(0) = x_0 \\ x'(0) = u_0, \end{cases} \qquad \begin{cases} x(1) = x_0 \\ x'(1) = u_1, \end{cases} .$$

En effet, ces conditions se traduisent par un système de Cramer en les a_k :

$$\begin{cases} & & & a_0 & = & x_0 \\ & & a_1 & & = & u_0 \\ a_3 & +a_2 & +a_1 & +a_0 & = & x_1 \\ 3a_3 & +2a_2 & +a_1 & & = & u_1. \end{cases}$$

7. En fait, ce qui est ignoble, ce sont les personnes qui l'exploitent, pas le logiciel lui-même. Mais passons.