Rapport de Projet : Splines Hermite Cubiques

BENNASSER Ahmed , EL HADARI Marouane

02 novembre 2025

Résumé

Ce rapport présente l'implémentation et l'analyse des splines Hermite cubiques pour l'interpolation de points dans le plan. Nous détaillons les aspects mathématiques, l'implémentation pratique, et comparons cette méthode avec d'autres approches d'interpolation.

Table des matières

1 Partie 1 : Splines Hermite Cubiques

1.1 Question 0 : Interpolation des points et tangentes

La courbe spline Hermite cubique est définie par :

$$P|_{[u_k,u_{k+1}]}(u) = P_k H_0(t) + P_{k+1} H_1(t) + (u_{k+1} - u_k) m_k H_2(t) + (u_{k+1} - u_k) m_{k+1} H_3(t)$$

avec $t = \frac{u-u_k}{u_{k+1}-u_k} \in [0,1]$.

Interpolation des points

Les polynômes d'Hermite cubiques vérifient :

$$\begin{aligned} H_0(0) &= 1, \quad H_0(1) = 0 \\ H_1(0) &= 0, \quad H_1(1) = 1 \\ H_2(0) &= 0, \quad H_2(1) = 0 \\ H_3(0) &= 0, \quad H_3(1) = 0 \end{aligned}$$

En $u = u_k$ $(t = 0)$:

$$P(u_k) = P_k \cdot 1 + P_{k+1} \cdot 0 + (u_{k+1} - u_k)m_k \cdot 0 + (u_{k+1} - u_k)m_{k+1} \cdot 0 = P_k$$

En $u = u_{k+1}$ $(t = 1)$:

$$P(u_{k+1}) = P_k \cdot 0 + P_{k+1} \cdot 1 + (u_{k+1} - u_k)m_k \cdot 0 + (u_{k+1} - u_k)m_{k+1} \cdot 0 = P_{k+1}$$

Interpolation des tangentes

Les dérivées des polynômes d'Hermite vérifient :

$$\begin{aligned} H_0'(0) &= 0, \quad H_0'(1) = 0 \\ H_1'(0) &= 0, \quad H_1'(1) = 0 \\ H_2'(0) &= 1, \quad H_2'(1) = 0 \\ H_3'(0) &= 0, \quad H_3'(1) = 1 \end{aligned}$$

La dérivée par rapport à u est :

$$\frac{dP}{du} = \frac{1}{u_{k+1} - u_k} \left[P_k H_0'(t) + P_{k+1} H_1'(t) + (u_{k+1} - u_k)m_k H_2'(t) + (u_{k+1} - u_k)m_{k+1} H_3'(t)\right]$$

En $u = u_k$ $(t = 0)$:

$$P'(u_k) = \frac{1}{u_{k+1} - u_k} \left[P_k \cdot 0 + P_{k+1} \cdot 0 + (u_{k+1} - u_k)m_k \cdot 1 + (u_{k+1} - u_k)m_{k+1} \cdot 0\right] = m_k$$

En $u = u_{k+1}$ $(t = 1)$:

$$P'(u_{k+1}) = \frac{1}{u_{k+1} - u_k} \left[P_k \cdot 0 + P_{k+1} \cdot 0 + (u_{k+1} - u_k)m_k \cdot 0 + (u_{k+1} - u_k)m_{k+1} \cdot 1\right] = m_{k+1}$$

1.2 Question 1 : Forme de Bernstein-Bézier : Conversion en base de Bernstein

Les polynômes d'Hermite s'expriment dans la base de Bernstein $B_i^3(t)$:

$$\begin{aligned} H_0(t) &= B_0^3(t) + B_1^3(t) \\ H_1(t) &= B_2^3(t) + B_3^3(t) \\ H_2(t) &= \frac{1}{3} B_1^3(t) \\ H_3(t) &= -\frac{1}{3} B_2^3(t) \end{aligned}$$

Avec $B_i^3(t) = \binom{3}{i} t^i (1-t)^{3-i}$.
Points de contrôle Bézier
Pour le segment $[k, k+1]$, les points de contrôle sont :

$$\begin{aligned}
\mathbf{b}_{3k} &= P_k \\
\mathbf{b}_{3k+1} &= P_k + \frac{1}{3} m_k \\
\mathbf{b}_{3k+2} &= P_{k+1} - \frac{1}{3} m_{k+1} \\
\mathbf{b}_{3(k+1)} &= P_{k+1}
\end{aligned}$$

La courbe s'écrit alors :

$$P|_{[u_k, u_{k+1}]}(u) = x_k(t) = \sum_{i=0}^{3} \mathbf{b}_{3k+i} B_i^3(t)$$

1.3 Question 2 : Représentation graphique

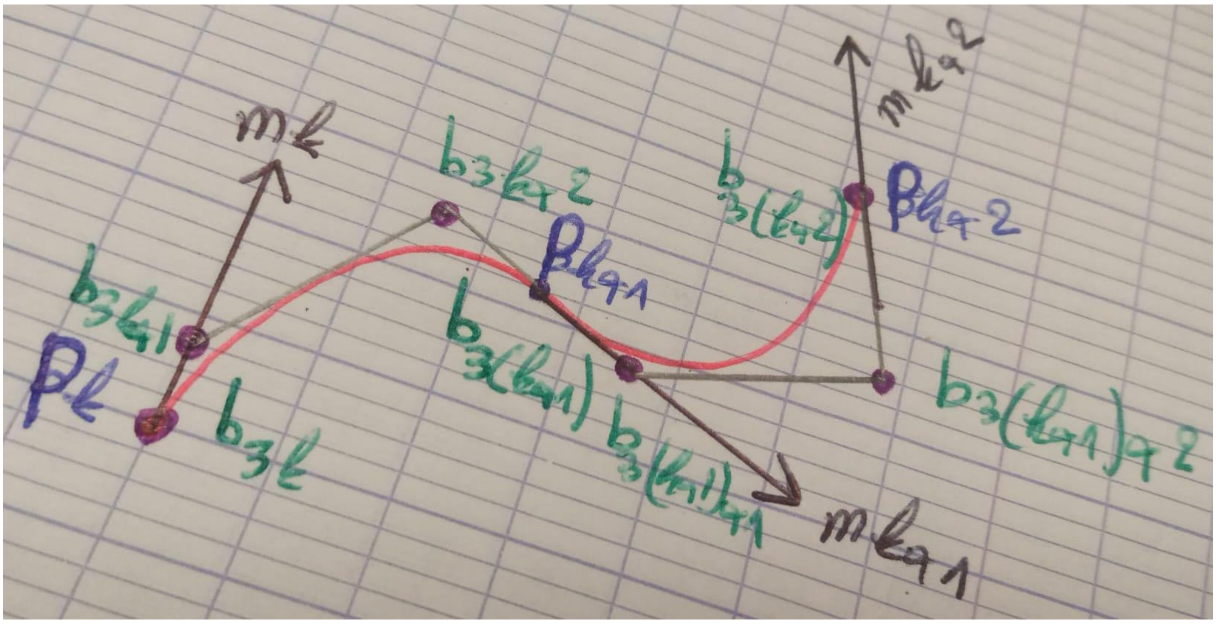

FIGURE 1 – Deux splines de Bézier

Le schéma illustre deux segments de Bézier consécutifs avec :
Points de données : P_k, P_{k+1}, P_{k+2}
Points de contrôle : b_{3k+1}, b_{3k+2}, $b_{3(k+1)}$, $b_{3(k+1)+2}$
Tangentes : m_k, m_{k+1}, m_{k+2}

1.4 Question 3 : Estimation des tangentes : Cardinal splines

Pour les points intérieurs $k = 1, \ldots, N-1$:

$$m_k = (1-c) \frac{P_{k+1} - P_{k-1}}{u_{k+1} - u_{k-1}}$$

où $c \in [0, 1]$ est le paramètre de tension.

Gestion des extrémités

Pour les points aux extrémités, on a choisie d'utilisées des **tangentes naturelles** car elles offrent un bon compromis entre continuité visuelle et stabilité numérique. Cette méthode préserve la forme globale de la courbe tout en évitant les comportements aberrants aux extrémités.

$$m_0 = 2 * \frac{P_1 - P_0}{u_1 - u_0} - m_1$$
$$m_N = 2 * \frac{P_N - P_{N-1}}{u_n - u_{n-1}} - m_{N-1}$$

(on avait un autre choix aussi , c'est : **Tangentes définies par l'utilisateur** , mais vu que cette dernière necessite un expertise du coté d'utilisateur , on l'ignore)

1.5 Question 4 : Implémentation interactive

Le programme python dans le fichier hermite-q4.py ,utilse :
Algorithme Hermite cubique
Cardinal Splines pour estimer les tangentes
Paramètre de tension c réglable par l'utilisateur
Trois paramétrisations (equidistante , chordale , centridèle) au choix
Fonctionnalités implémentées

- **Saisie interactive** : Ajout de points par clic souris
- **Paramètre de tension** : Curseur pour ajuster $c \in [0, 1]$
- **Choix des paramétrisations** :
  - Équidistante : $u_k = k$
  - Chordale : $u_k = u_{k-1} + \|P_k - P_{k-1}\|$
  - Centripète : $u_k = u_{k-1} + \sqrt{\|P_k - P_{k-1}\|}$
- **Visualisation** : Affichage des courbes, points de contrôle,

1.5.1 Qualité des Cardinal splines

Quand on varie le parametre de tension c , on remarque que :
* c petit ->La courbe dépasse les points (trop de vitesse)
* c grand -> La courbe s'arrête aux points (pas assez de vitesse)
* c moyen -> La courbe passe juste aux points (vitesse parfaite)

Les cardinal splines produisent des courbes généralement lisses, mais :

- Pour $c = 0$: Courbes très lisses mais pouvant présenter des oscillations
- Pour $c = 1$: Courbes plus "tendues", moins d'oscillations
- La convexité n'est pas toujours préservée

1.5.2 Bessel-tangents pour calculer les m_k

comme le choix de méthode de calcul des m_k influence le comportement de la courbe , on modifie le code de hermite-q4.py , en ajoutant au debut un choix de la methode ; soit bessel_tangents ,soit Cardinal splines .

en effet ,la methode de Bessel calcul :

$m_k = interpolation\_parabolique(P_{k-1}, P_k, P_{k+1})$

-> Courbe plus lisse : La parabole donne une meilleure estimation

-> Moins d'oscillations : Mieux adapté à la vraie forme de la courbe

1.6 Question 5 : Analyse de la courbure

le traçage de la courbure est fait à l'aide du code python dans le fichier Courbure.py

et on donne cet example , ou on a tracer 3 fonctions selon 2 methodes differentes (Cardinal avec c=0.3 et c=0.6) et Bessel , avec leurs fonction courbure ;

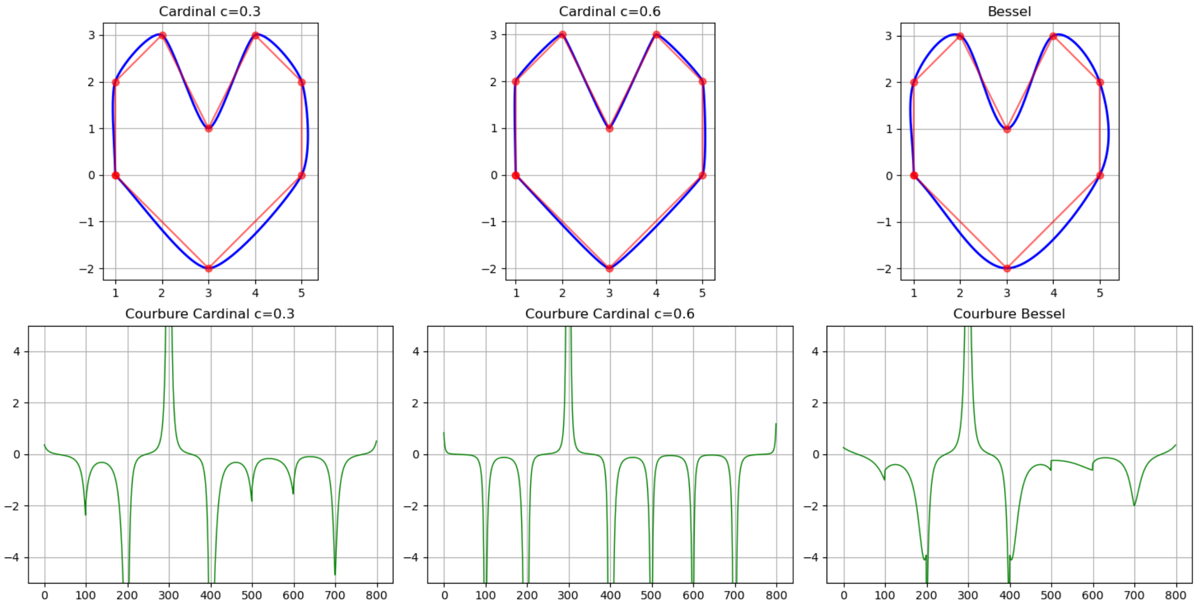

Figure 2 – Courbure de 3 fonctiond

ici , on remarque que la courbure donne un jugement sur la regularite de la fonction , car cette courbure est plus en plus regulié que la fonction elle meme est regulié ; la methode Bessel pour les memes points de controls donne plus de regularite vu que sa courbure ne fait pas beaucoup de sauts (pas trop de discontinuité)

Conditions sur la fonction de courbure

Une courbe de bonne qualité présente généralement : Une courbure continue (l'exemple de courbe C^2) Pas de variations brusques de courbure et Une courbure bornée.

en effet , L'ordre de continuité de la courbure est lié au nombre de fois qu'on peut dériver la dérivée seconde avant de perdre la continuité .

Pourquoi un graphe de courbure ? ?

Le graphe de courbure permet de :

— Détecter les défauts de lissage

— Identifier les points d'inflexion

— Vérifier la régularité de la courbe (C1 , C2 , ou discontinue)

est ce que notre nouveau chois des m_k améliore la forme de la courbe ?

oui , effectivement , a l'aide de l'exemple , on voie que le régularite de la courbure pour la methode **Bessel** est surement mieux que la methode **Cardinal**.

1.7 Question 6 : Exemples des objets

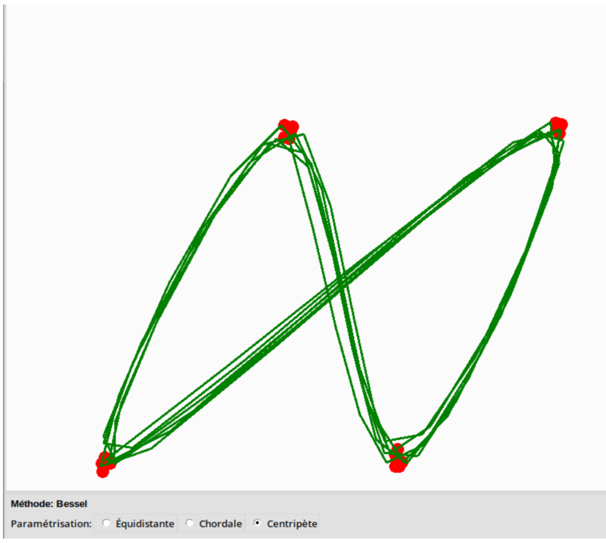

FIGURE 3 – objet 1 : infinie

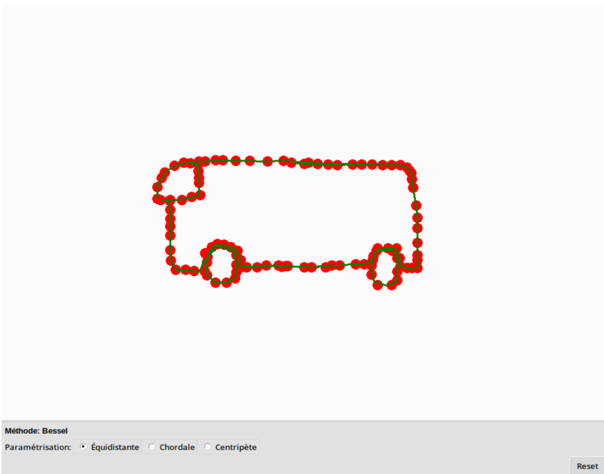

FIGURE 4 – Objet2 : Car

FIGURE 5 – objet3 : house

2 Partie 2 : Comparaison des méthodes d'interpolation

2.1 Question 7a : Interpolation de Lagrange

Algorithme d'Aitken-Neville

On a choisi L'algorithme d'Aitken-Neville qui permet d'éviter le calcul direct des polynômes de Lagrange :

$$P_{i,j}(t) = \frac{(t-u_i)P_{i+1,j}(t) - (t-u_j)P_{i,j-1}(t)}{u_j - u_i}$$

avec $P_{i,i}(t) = P_i$. cet algorithme est implementé dans le fichier comparaison.py avec les autres algorithmes qu'on va comparer .

2.2 Question 7b : Splines cubiques C²

Formulation mathématique

Pour chaque segment $[u_k, u_{k+1}]$, on a un polynôme cubique :

$$S_k(u) = a_k + b_k(u-u_k) + c_k(u-u_k)^2 + d_k(u-u_k)^3$$

Les conditions C² imposent :

$$\begin{aligned} S_k(u_k) &= P_k \\ S_k(u_{k+1}) &= P_{k+1} \\ S'_k(u_{k+1}) &= S'_{k+1}(u_{k+1}) \\ S''_k(u_{k+1}) &= S''_{k+1}(u_{k+1}) \end{aligned}$$

(implementé dans le fichier comparaison.py)

2.3 Question 8 : Superposition des courbes

dans le fichier comparaison.py ; on a implementé les courbes de lagrange et les splines C2 avec les courbes d'hermite deja definis

Donc cette fonctionnalité de superposition va nous permettre de comparer visuellement :

- Splines Hermite cubiques
- Interpolation de Lagrange
- Splines cubiques C^2

en comparant les 3 courbes pour le meme vecteur de points de controle

2.4 Question 9 : Analyse comparative

6points : Tous est Bon (Hermite!!!!)

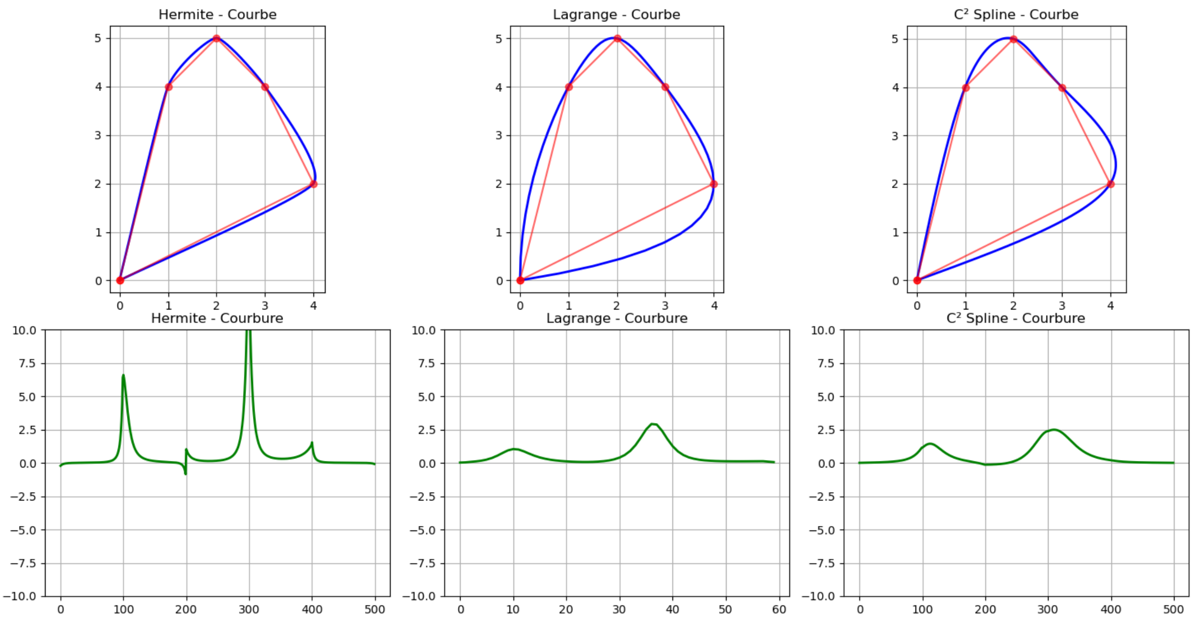

FIGURE 6 – 6points

Cet exemple pour 6 points de controles montre que les courbes de
Lagrange et splines C2 sont mieux qu'Hermite surtout du coté Lissage
mais les 3 presente un absence d'oscilations et ils sont stables
—-> Lissage : Hermite (Bon) , Lagrange et Splines C2 (excellents)
—-> Oscilation : Les 3 sont bons

2.5 9points : Lagrange ? ! ! !

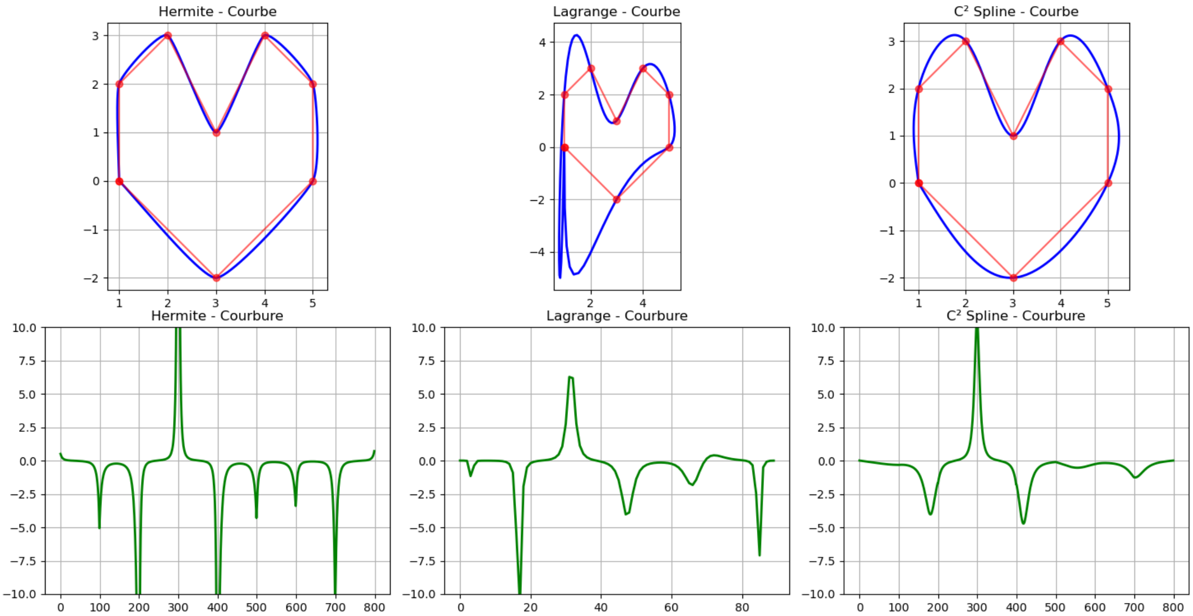

FIGURE 7 – 9points

cet exemple pour 9 points de controles montre que les courbes de
Lagrange et splines C2 sont mieux qu'Hermite du coté Lissage
mais la courbe de Lagrange est tres mauvaise en Stabilite et surtout en Oscilation .
—-> Lissage : Hermite (Bon) , Lagrange et Splines C2 (excellents)
—-> Oscilation : Lagrange est une catastrophe .
—-> Stabilite : Lagrange est mauvaise courbe

2.6 12points : Tous est ! ! !

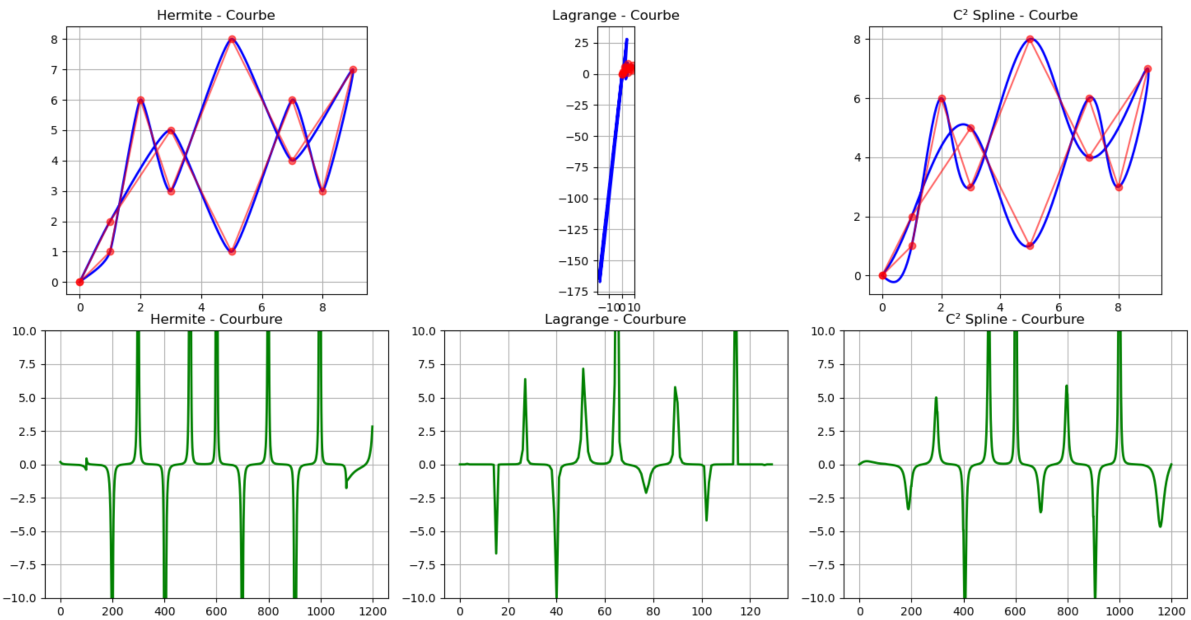

FIGURE 8 – 12points

cet exemple pour 12 points de controles montre que les courbes de
splines C2 sont mieux qu'Hermite et lagrange du coté Lissage
mais la courbe de Lagrange est tres mauvaise en Stabilite et surtout
en Oscilation , aussi la courbe de spline C2 et Hermite ont des bonnes performances comparée a Lagrange.

—-> Lissage : Hermite (Bon) , Lagrange et Splines C2 (excellents)

—-> Oscilation : Lagrange est une catastrophe .

—-> Performance : Spline C2 et Hermite > LAgrange

d'apres les exemples qu'on deja cité , on peut remarquer qu'en generale :

Méthode	**Lissage**	**Oscillations**	**Performance**	**Stabilité**
Hermite	Bon	Moyennes	Bonne	Bonne
Lagrange	Excellent	Fortes	Médiocre	Mauvaise
Splines C^2	Excellent	Faibles	Moyenne	Excellente

TABLE 1 – Comparaison des méthodes d'interpolation

2.7 Observations :

1. Les **splines Hermite** offrent un bon compromis entre lissage et contrôle
2. L'**interpolation de Lagrange** souffre du phénomène d'oscilation pour de nombreux points (a partir de 9 points)
3. Les **splines C^2** produisent les courbes les plus lisses mais sont plus coûteuses
4. Le **choix des tangentes** est crucial pour la qualité des splines Hermite

3 Conclusion

3.1 Bilan du projet

Ce projet a permis d'implémenter et d'analyser les splines Hermite cubiques. Les objectifs ont été atteints :

— Implémentation complète des splines Hermite

— Interface interactive fonctionnelle

— Comparaison avec d'autres méthodes d'interpolation

— Analyse qualitative des résultats

— Beaucoup d'exemples , illustrations .

FIGURE 9 – A vous