Interpolation avec des splines cubiques

L'interpolation permet de faire passer une courbe à travers certains points choisis dans le plan. Les méthodes d'interpolation sont très utiles dans beaucoup de domaines : industrie, animation, image, économie, etc... En général, on peut voir l'interpolation comme le fait de deviner des valeurs que l'on a pas en fonction de valeurs que l'on connaît. Nous allons étudier ici une méthode d'interpolation très connue et largement utilisée : l'interpolation par splines cubiques.

Démarrez le programme Matlab en entrant la commande **matlab** dans un terminal.

De plus, dézippez les trois fichiers zip joints avec cet énoncé : vous trouverez trois dossiers, chacun correspondant à l'un des trois exercices de ce TP.

1 Des splines 1d pour gérer des carrosseries de voitures

1. Dans Matlab, allez dans le dossier **Exo1**, et dessinez des points sur un graphe à l'aide de la commande

   ```
   [x,y] = EntrerPoints();
   ```

   Puis, affichez ces points à l'aide de la commande

   ```
   plot(x,y,'*');
   ```

2. Nous souhaitons faire passer une courbe à travers ces points. On va commencer avec des lignes droites, juste pour s'amuser... Réalisez cette opération à l'aide des commandes suivantes :

   ```
   plot(x,y);
   hold all;
   plot(x,y,'*');
   ```

3. On va maintenant d'utiliser l'interpolation polynomiale. Pour ce faire, utilisez la commande suivante :

   ```
   InterpolationPolynomiale(x,y);
   ```

   Quel est le problème de cette méthode ?

4. On va utiliser des splines cubiques pour l'interpolation. La première méthode que l'on va tester consiste à dire qu'à chaque point (x_i, y_i) du plan ($i \in [1; n]$), la courbe C_i respecte les propriétés suivantes :

$$\begin{cases} C_i(x_i) = y_i \\ C_i(x_{i+1}) = y_{i+1} \\ C_i'(x_i) = C_i'(x_{i-1}) \\ C_i''(x_i) = C_i''(x_{i-1}) \end{cases}$$

On considérera $C_1'(x_1) = C_1''(x_1) = 0$.
Implémentez cette méthode d'interpolation dans le fichier **ConstruireSpline.m**. Testez la ensuite avec la commande

```
ConstuireSpline(x,y);
```

Quel est le problème de cette méthode ?

5. On va utiliser la méthode dite de Catmull-Rom. Cette fois-ci, les conditions sont les suivantes :

$$\begin{cases} C_i(x_i) = y_i \\ C_i(x_{i+1}) = y_{i+1} \\ C_i'(x_i) = l_i \\ C_i'(x_{i+1}) = l_{i+1} \end{cases}$$

et on pose $l_i = \frac{y_{i+1}-y_{i-1}}{x_{i+1}-x_{i-1}}$.
Implémentez cette méthode d'interpolation dans le fichier **ConstruireSplineCatmull.m**. Testez la ensuite avec la commande

```
ConstuireSplineCatmull(x,y);
```

6. Enfin, testez l'approche classique des splines consistant à poser

$$\begin{cases} C_i(x_i) = y_i \\ C_i(x_{i+1}) = y_{i+1} \\ C_i'(x_i) = C_i'(x_{i-1}) \\ C_i''(x_i) = C_i''(x_{i-1}) \end{cases}$$

avec cette fois-ci $C_1'(x_1) = C_{n-1}'(x_n) = 0$. Cette approche est déjà codée est vous pouvez la tester avec la commande

```
ConstuireSplineClassique(x,y);
```

7. Pour finir, testez les deux dernières méthodes de spline sur un profil de voiture en faisant

```
load('Twingo.mat');
image(Image);
ConstuireSplineCatmull(x,y);
```

Vous pouvez aussi tester sur le fichier **Fiat500.mat**.

2 Des splines 2d

On souhaite maintenant faire des splines 2d (des surfaces splines), à l'aide des programmes réalisés précédemment.

1. Allez dans le dossier **Exo2** et chargez le fichier **Terrain.mat** à l'aide de la commande

   ```
   load('Terrain.mat');
   ```

   Visualisez les points relevés du terrain à l'aide des commandes :

   ```
   surf(x1, y1, z1);
   camproj('perspective');
   ```

2. Complétez le fichier **ConstruireSpline2d.m** afin de réaliser une interpolation par splines 2d, et testez votre programme à l'aide de la commande

   ```
   ConstruireSpline2d(x1, y1, z1);
   ```

3. Testez la même commande avec l'ensemble des points $(x2, y2, z2)$, et comparez les résultats en visualisant le "vrai terrain" avec la commande

   ```
   surf(Terrain, 'EdgeColor','none','LineStyle','none');
   camproj('perspective');
   ```

3 Agrandir une image

On va utiliser ce que l'on a fait précédemment pour agrandir intelligemment une image.

1. Allez dans le dossier **Exo3** et chargez l'image du poisson à l'aide de la commande

   ```
   Im = imread('poisson.png');
   image(Im,'CDataMapping','scaled');
   colormap(gray);
   ```

2. Complétez le fichier **ZoomImage.m** permettant d'agrandir une image, et testez votre programme avec la commande

   ```
   Zoom = ZoomImage(Im, 5);
   image(Zoom,'CDataMapping','scaled');
   colormap(gray);
   ```