Géométrie et Espaces de Formes - TP1

Tong ZHAO (tong.zhao@eleves.enpc.fr)

1 Noyaux reproduisants et interpolation

Exo 1. Tout d'abord on montre que $\forall f \in H^1_\sigma$, $\langle f, K(\cdot, x)\rangle_{H^1_\sigma} = f(x)$.

$$\begin{aligned}
\langle f, K(\cdot, x)\rangle_{H^1_\sigma} &= \int_{\mathbb{R}} f(t)K(t,x) + \sigma^2 f'(t)K'(t,x)dt \\
&= \int_{-\infty}^{x} \frac{1}{2\sigma} f(t)e^{\frac{t-x}{\sigma}} + \sigma^2 f'(t)\frac{1}{2\sigma^2}e^{\frac{t-x}{\sigma}}dt + \int_{x}^{\infty} \frac{1}{2\sigma} f(t)e^{\frac{x-t}{\sigma}} + \sigma^2 f'(t)\frac{1}{2\sigma^2}e^{\frac{x-t}{\sigma}}dt
\end{aligned}$$

En utilisant l'intégration par parties, on a:

$$\begin{aligned}
\int_{-\infty}^{x} \frac{1}{2\sigma} f(t)e^{\frac{t-x}{\sigma}} + \sigma^2 f'(t)\frac{1}{2\sigma^2}e^{\frac{t-x}{\sigma}}dt &= \frac{1}{2\sigma}\int_{-\infty}^{x} \frac{1}{2\sigma} f(t)e^{\frac{t-x}{\sigma}}dt + \frac{1}{2}\Big(f(t)e^{\frac{t-x}{\sigma}}|_{-\infty}^{x} - \int_{-\infty}^{x}\frac{1}{\sigma}f(t)e^{\frac{t-x}{\sigma}}dt\Big) \\
&= \frac{1}{2}f(x)
\end{aligned}$$

$$\begin{aligned}
\int_{x}^{\infty} \frac{1}{2\sigma} f(t)e^{\frac{x-t}{\sigma}} + \sigma^2 f'(t)\frac{1}{2\sigma^2}e^{\frac{x-t}{\sigma}}dt &= \frac{1}{2\sigma}\int_{x}^{\infty} \frac{1}{2\sigma} f(t)e^{\frac{x-t}{\sigma}}dt + \frac{1}{2}\Big(f(t)e^{\frac{x-t}{\sigma}}|_{x}^{\infty} - \int_{x}^{\infty}\frac{1}{\sigma}f(t)e^{\frac{x-t}{\sigma}}dt\Big) \\
&= \frac{1}{2}f(x)
\end{aligned}$$

On en déduit que $\langle f, K(\cdot, x)\rangle_{H^1_\sigma} = f(x)$ et donc H^1_σ est un espace à noyau reproduisant dont le noyau est la fonction $K_\sigma : \mathbb{R}^2 \to \mathbb{R}$ définie par:

$$\forall x, y \in \mathbb{R}, K_\sigma(x,y) = \frac{1}{2\sigma}e^{-\frac{|x-y|}{\sigma}}$$

Exo 2. On prend $\sigma = 2$ et calcule plusieurs valeurs de fonctions en utilisant deux méthodes. Le résultat est comme suit:

x	Noyau	Fonction
0.5	0.778797	0.778801
1.5	0.105403	0.105399
2.5	0.00193054	0.00193045

On observe que la propiété se tient numériquement.

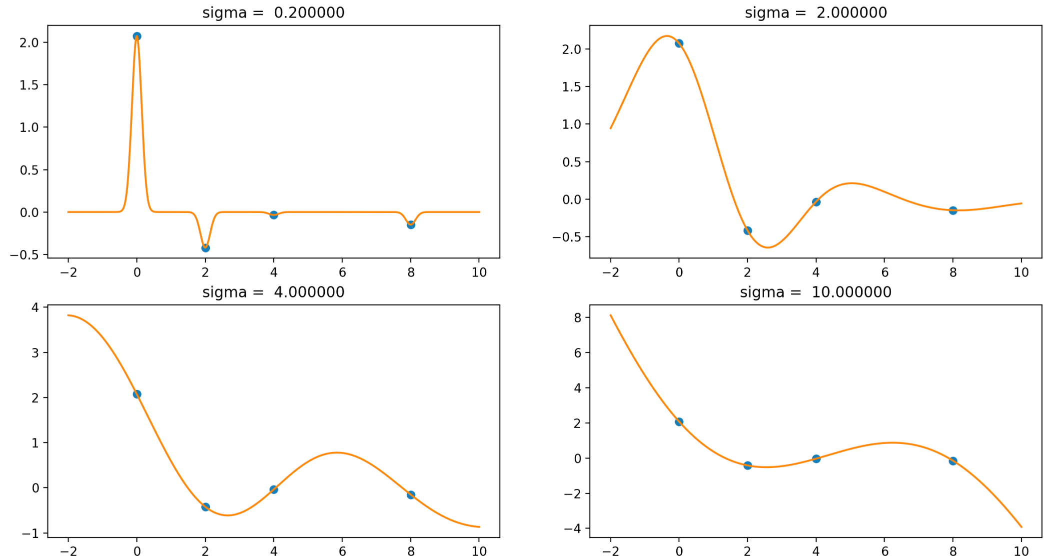
Exo 3. La fonction est écrite comme suit:

```
def KernelMatrix(x, y, h):
  x = np.expand_dims(x, axis = 1)
  y = np.expand_dims(y, axis = 0)
  return h(np.linalg.norm(x - y, axis = -1))
```

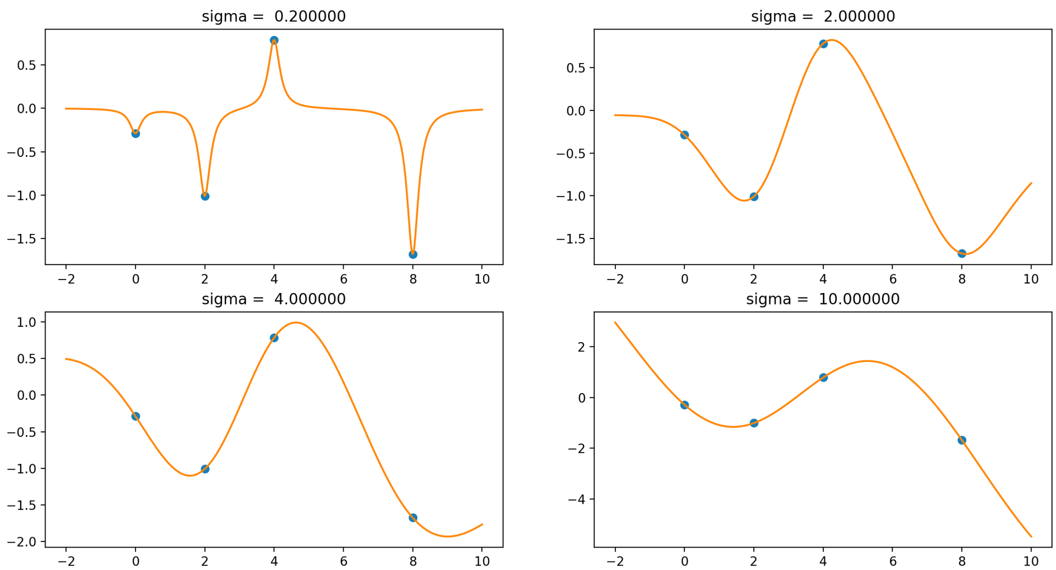
Exo 4. La fonction est écrite comme suit:

```
def Interp(x, y, c, h):
  Kh_yy = KernelMatrix(y, y, h)
  A = np.linalg.solve(Kh_yy, c)
  Kh_xy = KernelMatrix(x, y, h)
  return Kh_xy.dot(A)
```

Exo 5. On utilise tout d'abord le noyau gaussien et on pose $\sigma \in \{0.2, 2, 4, 10\}$.



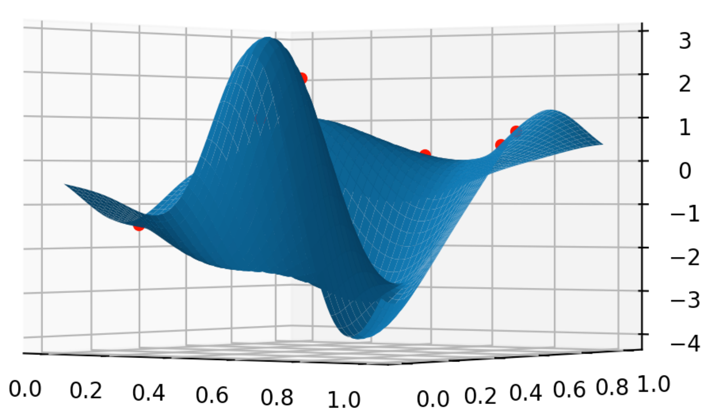
Après on teste le noyau de Cauchy.



Exo 6. On vérifie numériquement que la norme de la solution produite par le noyau $K_\sigma(x,y) = \frac{1}{2\sigma}e^{-\frac{|x-y|}{\sigma}}$ dans H^1_σ est toujours inférieure à la norme d'autres solutions. Ici on compare la norme de trois noyaux: K_σ, K_{gauss} et K_{cauchy}:

```
norm of h1 kernel: 4.219232
norm of gauss kernel: 4.663408
norm of cauchy kernel: 4.557497
```

Exo 7. On choisit 10 points y_i dans $[0,1]^2$ et on interpole la fonction sur une grille uniforme de 100×100. Le résultat est comme suit:



Exo 8. La solution du problème est sous forme de $f(y_i) = \sum_{j=1}^{n} K(y_i, y_j)\alpha_j$ où les $\alpha_j \in \mathbb{R}$ sont solutions du système linéaire:

$$\forall i, 1 \leq i \leq n, \sum_{j=1}^{n} K(y_i, y_j)\alpha_j = c_i$$

Donc on a:

$$\begin{aligned} J(f) &= \lambda \|f\|_H^2 + \sum_{i=1}^{n} \Big(f(y_i) - c_i\Big)^2 \\ &= \lambda \sum_{i=1}^{n} \alpha_i \sum_{j=1}^{n} K_H(y_i, y_j)\alpha_j + \sum_{i=1}^{n} \Big(c_i - \sum_{j=1}^{n} K_H(y_i, y_j)\alpha_j\Big)^2 \end{aligned}$$

On calcule le gradient par rapport à α_j:

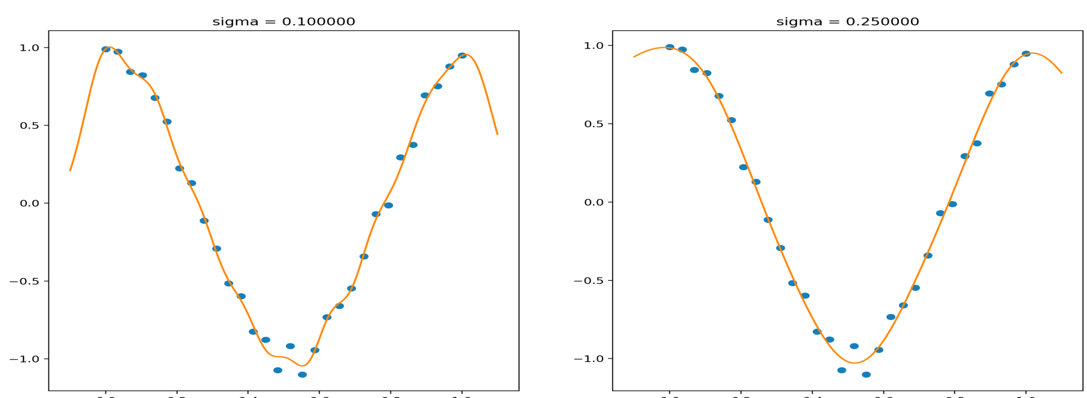
$$DJ_{\alpha_k}(h_k) = 2\lambda \sum_{i=1}^{n} K_H(x_i, x_k)\alpha_i h_k + 2\sum_{i=1}^{n} \Big(c_i - \sum_{j=1}^{n} K_H(x_i, x_j)\alpha_j\Big) \cdot \Big(-K_H(x_k, x_i)h_k\Big)$$

Le gradient est égale à 0 si et seulement si:

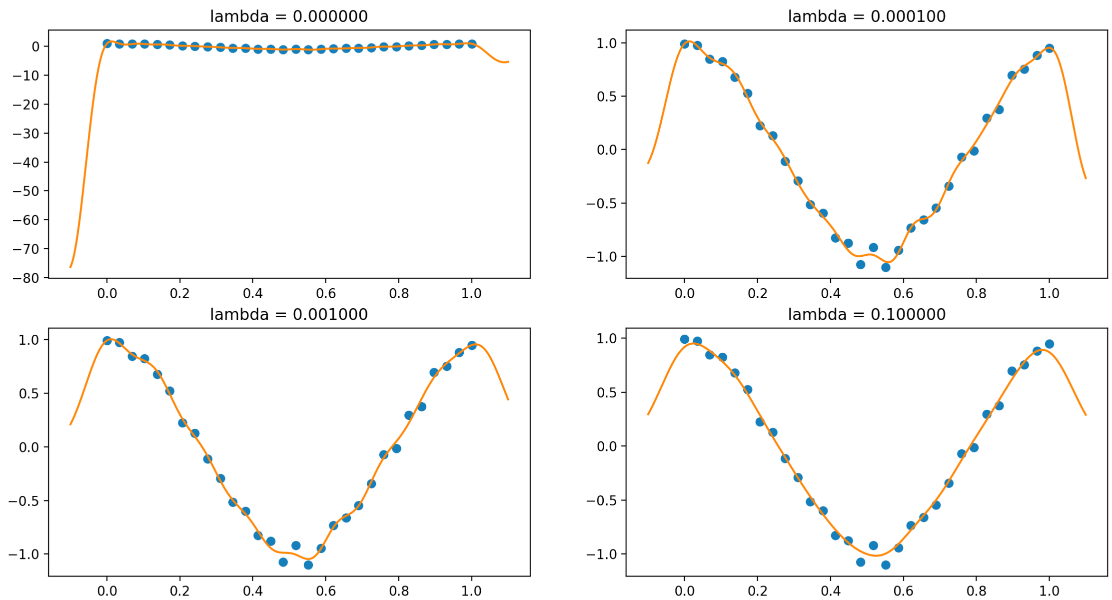
$$\forall i, 1 \leq i \leq n, \Big(\lambda + \sum_{j=1}^{n} K(x_i, x_j)\Big)\alpha_j = c_i$$

Donc on a: $(\lambda Id + \mathbb{K})\alpha_j = c_I$.

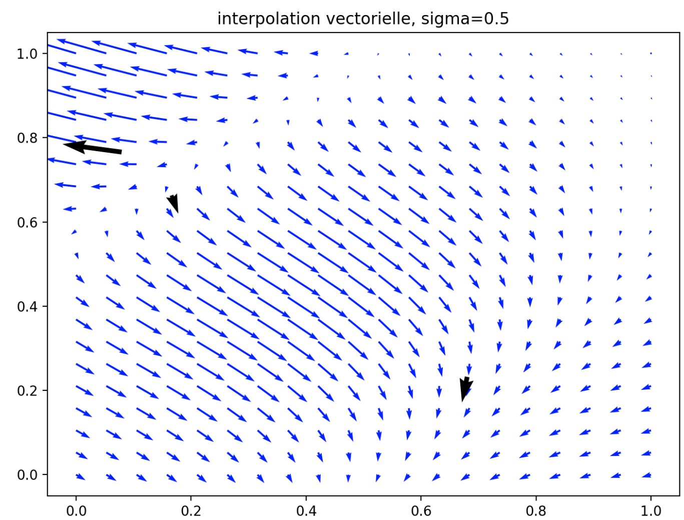
On utilise le noyau gaussien et on pose $\sigma \in \{0.1, 0.25\}$ et $\lambda = 0.001$. Le résultat est comme suit.



Maintenant on fixe $\sigma = 0.1$ et on pose $\lambda \in \{0, 0.0001, 0.001, 0.1\}$ et le résultat est comme suit.

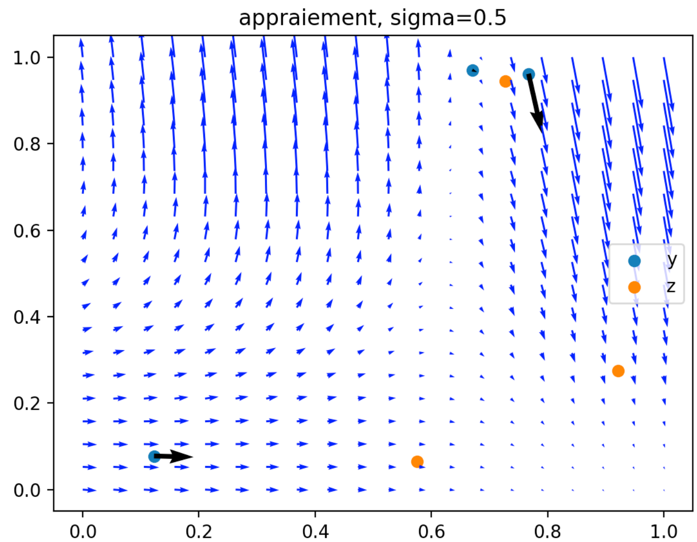


Exo 10. On teste la fonction *InterpGrid2D* avec $d = m = 2$. On génère trois vecteurs et on évalue sur une grille uniforme de taille 20×20. Le résultat est comme suit.



2 Appariement de points labellisés

Exo 11. On prend deux ensembles de points $\mathcal{Y}$ et $\mathcal{Z}$. En posant $\gamma_i = z_i - y_i$, on retrouve le problème d'interpolation vectorielle en grille. On visualise un résultat:



Exo 12. On labelise deux ensembles de points C_1 et C_2 à la main et on transforme C_1 à C_2 en utilisant le méthode d'appariement.