Sur la caractérisation d'outils relatif à la résolution d'équations aux dérivées partielles

Gaggini Lorenzo

Les résultas ci-aprés sont tous donné en coordoné cartésienne, Quand un résultats est affirmé, ça source est indiqué en annex par sont repére (1) associé.

On cherche dans un premier temps à appréhendé plusieurs outils en lien avec la résolution des EDP

Soit u: $\begin{array}{l} \mathbb{R}^2 \longrightarrow \mathbb{R} \\ (x,y) \longmapsto u(x,y) \end{array}$ une application que l'on carractérise de «champs scalaire »(1)

Soit V: $\begin{pmatrix} x_1 \\ x_2 \\ . \\ . \\ x_n \end{pmatrix} \longmapsto \begin{pmatrix} V_1(x_1, x_2 \ldots, x_n) \\ V_2(x_1, x_2 \ldots, x_n) \\ . \\ . \\ V_n(x_1, x_2 \ldots, x_n) \end{pmatrix}$ une application que l'on carractérise de «champs vectoriel»

avec les V_i des application de $\mathbb{R}^n \longrightarrow \mathbb{R}^n$ toutes de classes C^n sur $\mathbb{R}^n$ (2)

On donne dans un premier temps $\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$ l'opérateur laplacien (3) en dimention 2 associé à u

Avec par ailleur : Grad(u) le gradiant de u tel que $\text{Grad}(u) = \frac{\partial u}{\partial x}(x,y) e_x + \frac{\partial u}{\partial y}(x,y) e_y$ (4)

Finalement on donne Div(V) la divergence de V donner par : $\text{Div}(V) = \sum_{i=1}^{n} \frac{\partial V_{x_i}}{\partial x_i}$ (5)

Sur les pages suivantes sont joint les graph associé au proposistion qui précède, illustré avec un exemple en dimention 2

Annex

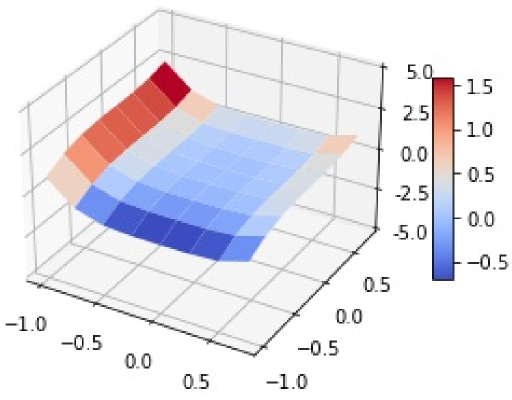
(1) http://www.tangentex.com/OperateursDiff.htm

(2)https://fr.wikipedia.org/wiki/Champ_de_vecteurs

(3),(4),(5): http://www.tangentex.com/OperateursDiff.htm

(6) https://matplotlib.org/mpl_toolkits/mplot3d/tutorial.html7

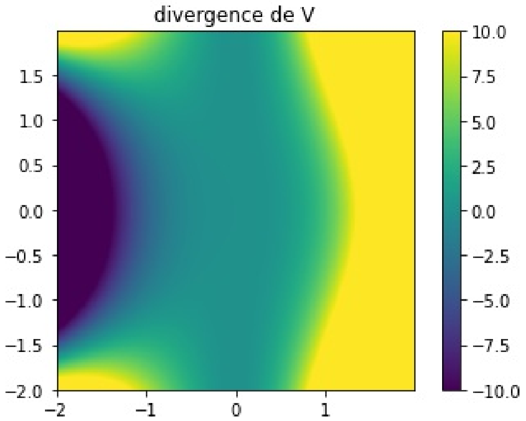
(7),(8) : http://www.tangentex.com/BibliothequeCodes.htm#Ind52



Algorithme pour le graph de u

```
import matplotlib.pyplot as plt
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStr-
Formatter
import numpy as np


fig = plt.figure()
ax = fig.gca(projection='3d')

# Make data.
X = np.arange(-1, 1, 0.25)
Y = np.arange(-1, 1, 0.25)
X, Y = np.meshgrid(X, Y)
Z = 2*X**4+Y**3

# Plot the surface.
surf = ax.plot_surface(X, Y, Z, cmap=cm.coolwarm,
                       linewidth=0, antialiased=True)

# Customize the z axis.
ax.set_zlim(-5.01, 5.01)
ax.zaxis.set_major_locator(LinearLocator(5))
ax.zaxis.set_major_formatter
(FormatStrFormatter('%.1f'))

# Add a color bar which maps values to colors.
fig.colorbar(surf, shrink=0.5, aspect=10)

plt.show()
```

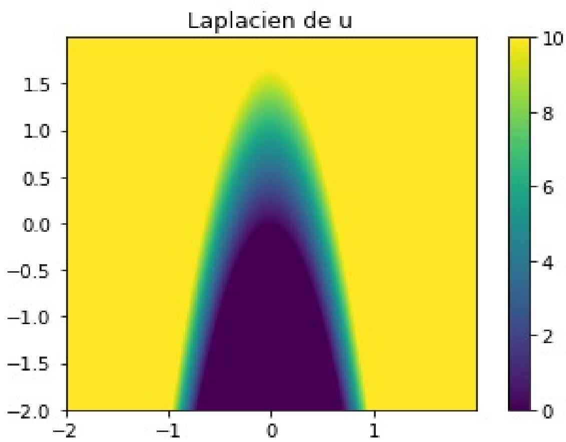
(6)



(7)

Algorithme pour la divergence de V

```
from scipy import meshgrid, diff, arange
from matplotlib.pyplot import *

# Définition du champ
def Champ(x,y):
    Ex = x**4
    Ey = x**2*y**3
    return Ex,Ey

# Calcul de la divergence
def Divergence(Fx,Fy):
    div = (diff(Fx,axis=1)/hx)[:-1,:] + (diff(Fy,axis=0)/hy)[:,:-1]
    return div


# Définition de la grille de calcul
hx = 0.01
hy = 0.01
X = arange(-2.,2.,hx)
Y = arange(-2.,2.,hy)
x,y = meshgrid(X,Y)

# Calcul du champ
Ex, Ey = Champ(x,y)

# calcul de la divergence du champ
divE = Divergence(Ex,Ey)

# tracé de la divergence
figure()
title("divergence")
pcolormesh(x, y, divE, vmin=-10, vmax=10)
colorbar()
gca().set_aspect("equal")
show()
```



(8)

Algorithme pour le laplacien de u

```
from scipy import meshgrid, diff, arange, sqrt
from matplotlib.pyplot import *

#Définition de la fonction scalaire
def Fonction(x,y):
    return 2*x**4 + y**3

#Routine de calcul du Laplacien
def LaplacienScalaire(x,y,F):
    dFx = (diff(F,n=2,axis=1)/hx**2)[:-2,:]
    dFy = (diff(F,n=2,axis=0)/hy**2)[:,:-2]
    return dFx + dFy

#Définition de la grille de calcul
hx = 0.001
hy = 0.001
X = arange(-2,2,hx)
Y = arange(-2,2,hy)
x,y = meshgrid(X,Y)

#Calcul de la fonction
f = Fonction(x,y)

#Calcul du laplacien
delta2 = LaplacienScalaire(x,y,f)

#Tracé du laplacien
figure()
title('Laplacien de u')
pcolormesh(x[:-2,:-2], y[:-2,:-2], delta2, vmin=0, vmax=10)
colorbar()
gca().set_aspect("equal")
show()
```