Zadání příkladu pro cvičení z předmětu Programování prakticky

Úloha č. 5 — 26. května 2021

Z mechaniky znáte dobře kmity harmonického oscilátoru buzeného silou s harmonickým časovým průběhem. V tomto příkladě půjde o složitější systém – membránu daného tvaru uchycenou podél svého obvodu. Místo jedné výchylky harmonického oscilátoru tak budeme hledat výchylku ve vybraných bodech membrány. Váš program bude muset pořídit seznam takových bodů, sestavit a vyřešit soustavu rovnic určující jejich výchylku a na konec výsledky hezky vykreslit.

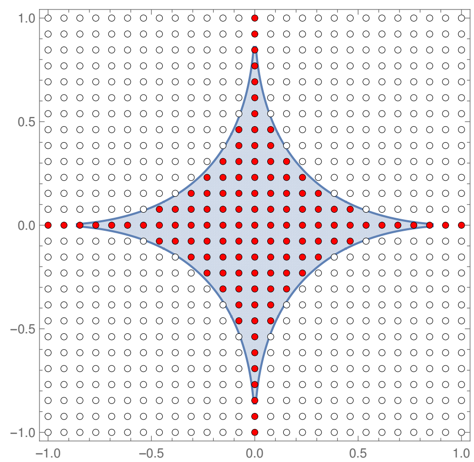
Uvažujte oblast Ω, která leží uvnitř čtverce $<\!-1,1\!> \times <\!-1,1\!>$. Oblast je určená podmínkou

$$\sqrt{|x|} + \sqrt{|y|} < 1 \; .$$

Membránu tohoto tvaru nahardíme sadou bodů u_b, které všechny budou oscilovat v souladu s Newtonovým zákonem

$$F_b^{\text{memb}} - m\, a_b = F_b^{\text{bud}},$$

kde F_b^{memb} představuje silové působení sousedních bodů membrány, F_b^{bud} nějakou budící sílu. Pro harmonický průběh je zrychlení úměrné výchylce a dostaneme lineární problém, který budeme řešit. Mnoho bodů membrány si vyžádá řešit úlohu jako maticovou soustavu lineárních rovnic pro neznámé výchylky způsobená danou silou.



Obrázek 1. Vyplněná plocha znázorňuje oblast Ω, body mříže, jenž leží uvnitř ní, jsou zvýrazněné.

1. Napište funkci `bodMembrany`, která vrátí `True` právě když její parametry x, y leží uvnitř Ω. Funkci otestujte alespoň takto:

```
print( bodMembrany(0.2,0.3), bodMembrany(0.4,0.3) )

True False
```

Představte si, že celý čtverec pokrývá mříž bodů o souřadnicích

$$x = \frac{i}{n}, \quad y = \frac{j}{n}, \qquad i, j = -n, -n+1, ... -1, 0, 1, ..., n-1, n,$$

kde n je vhodně velké celé číslo. V rámci programu by n mohlo představovat globální konstantu. Taková mříž je znázorněna pro konkrétní oblast Ω na Obrázku 1.

2. Napište funkci, která vrátí seznam bodů mříže, které pro dané n leží unvitř Ω. Ve finálním výpočtu zvolte n tak velké, aby počet bodů mříže v seznamu byl asi 1000. Délka seznamu S nechť v dalším představuje dimenzi D vektorového prostoru $I\!R^D$.

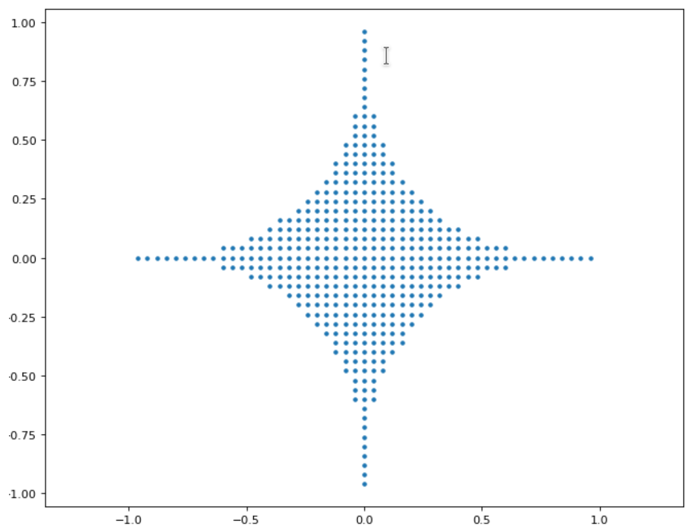
Funkci napište tak, aby vrácená hodnota měla podobu čtveřice seznamů indexů i_a, j_a, x_a, y_a, kde a je index bodu ze seznamu. Ověřte, že pro $n = 4$ dostanete seznamy

```
print( seznamyBodu(4) )

([-3, -2, -1, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3],
 [0, 0, 0, -3, -2, -1, 0, 1, 2, 3, 0, 0, 0],
 [-0.75, -0.5, -0.25, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.25, 0.5, 0.75],
 [0.0, 0.0, 0.0, -0.75, -0.5, -0.25, 0.0, 0.25, 0.5, 0.75, 0.0, 0.0, 0.0])
```

Pořadí bodů na zseznamu může být samozřejmě odlišné. Vyzkoušejte, že příkazy

```
( sezI, sezJ, sezX, sezY ) = seznamyBodu(25)

import matplotlib.pyplot as plt
plt.axis('equal')
plt.plot(sezX,sezY,'.')
plt.show()
```

vytvoří to, co je na Obr. 2.



Obrázek 2. Souřadnice $\{x, y\}$ vrácené voláním `seznamyBodu(25)`.

3. Napište funkci `VzdalenostBoduVMrizi(a,b,sezI,sezJ)`, která se podívá na a-tý a b-tý bod v seznamu reprezentovaném dvěma argumenty `sezI,sezJ` a vráti hodnotu $|i_a - i_b| + |j_a - j_b|$, kde $\{i_a, j_a\}$ jsou celočíselné souřadnice a-tého bodu v seznamu. Podobně $\{i_b, j_b\}$ pro b-tý bod v seznamu S. Odsud je vidět, že je výhodné si ukládat i celočíselné souřadnice bodů ze seznamu.

Vše zkontrolujte spočtením počtu bodů na seznamu a dvojic bodů se vzdáleností 0 a 1.

```
( sezI, sezJ, sezX, sezY ) = seznamyBodu(5)

vzdalenostiDvojic = [VzdalenostBoduVMrizi(a,b,sezI,sezJ) for a in range(len(sezI)) for b in range(len(sezI))]
print(print(len(sezI)), vzdalenostiDvojic.count(0), vzdalenostiDvojic.count(1) )

21 21 48
```

4. Napište funkci `maticeSoustavy(kappa_n2, sezI, sezJ)` , která pro dané reálné číslo κ (související s frekvencí jíž působí budící síla) vrátí řešení matici soustavy lineárních rovnic která jednak popisuje sílu, jíž na sebe působí sousední body membrány a také zahrnuje zrychlení/setrvačnost kmitavého pohyby daného bodu, a je dána předpisem

$$A_{cd} = \begin{cases} 1 & \text{vzdalenost\_bodu\_v\_mrizi}(c,d) = 1, \\ \dfrac{\kappa}{n^2} - 4 & \text{pokud } c = d, \\ 0 & \text{jinak.} \end{cases} \tag{1}$$

Zde c, d jsou indexy bodů ze seznamu. Pokud souhlasí vaše pořadí bodů na seznamu s tím z kontroly v bodě 2, měla by další kontrola dopadnout takto:

```
import numpy as np

( sezI, sezJ, sezX, sezY ) = seznamyBodu(4)
print( np.array( maticeSoustavy(1, sezI, sezJ) ) )

[[-3  1  0  0  0  0  0  0  0  0  0  0  0]
 [ 1 -3  1  0  0  0  0  0  0  0  0  0  0]
 [ 0  1 -3  0  0  0  1  0  0  0  0  0  0]
 [ 0  0  0 -3  1  0  0  0  0  0  0  0  0]
 [ 0  0  0  1 -3  1  0  0  0  0  0  0  0]
 [ 0  0  0  0  1 -3  1  0  0  0  0  0  0]
 [ 0  0  1  0  0  1 -3  1  0  0  1  0  0]
 [ 0  0  0  0  0  0  1 -3  1  0  0  0  0]
 [ 0  0  0  0  0  0  0  1 -3  1  0  0  0]
 [ 0  0  0  0  0  0  0  0  1 -3  0  0  0]
 [ 0  0  0  0  0  0  1  0  0  0 -3  1  0]
 [ 0  0  0  0  0  0  0  0  0  0  1 -3  1]
 [ 0  0  0  0  0  0  0  0  0  0  0  1 -3]]
```

5. Předpokládejte, že je budící síla je dána předpisem

$$b_a = 7x_a + 5y_a + 3,$$

kde x_a, y_a jsou souřadnice a-tého bodu. Napište funkci `pravaStranaSoustavy(sezX, sezY)`, která vrátí seznam b_a. Vyzkoušejte, že pokud použijete `numpy` a konvertujete

```
    x = np.array(sezX)
    y = np.array(sezY)
```

můžete ve funkci jednoduše napsat

```
        return 7*x + 5*y +3
```

Funkci otestujte (pořadí nemusí sedět, máte-li jinak seřazené body v seznamu)

```
( sezI, sezJ, sezX, sezY ) = seznamyBodu(5)
print( pravaStranaSoustavy(sezX, sezY) )

[-2.25 -0.5  1.25 -0.75 0.5  1.75  3.   4.25  5.5  6.75  4.75  6.5  8.25]
```

6. Vyřešte nyní soustavu rovnic pro amplitudu oscilací

$$\boldsymbol{A}\vec{z} = \vec{b},$$

kde $\vec{z}, \vec{b} \in I\!R^D$. Jde o maticový zápis tzv. Helmholtzovy rovnice, jiř při studiu ještě potkáte. Vektor $\vec{b}$, jak již víme, má složky dané amplitudou síly, která na membránu působí v jednotlivých bodech, matice soustavy zahrnuje setrvačnost harmonického pohybu a síly od sousedních bodů membrány.

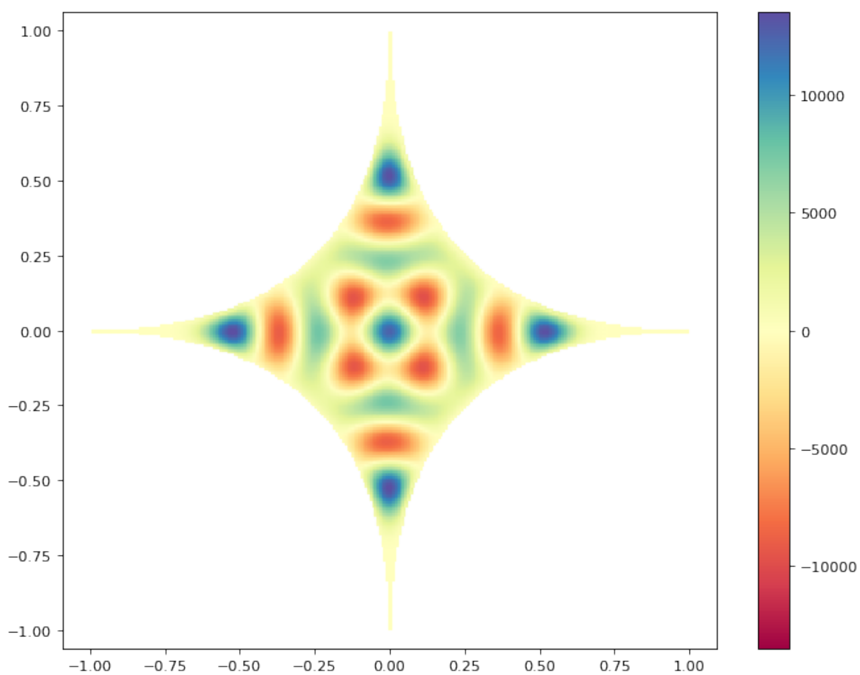
za pomoci následující nápovědy / testu:

```
( sezI, sezJ, sezX, sezY ) = seznamyBodu(4)

A = maticeSoustavy(2, sezI, sezJ)
b = pravaStranaSoustavy(sezX, sezY)

amplitudy = np.linalg.solve(A,b)

print( amplitudy )

[ 6.875 11.5  15.625 5.125 9.5  14.375 21.   8.125 -0.5  -3.625 6.875 -2.5  -5.375]
```

7. Pokud jste postupovali dobře je potřeba zvýšit počet bodů tak, aby se situce více podobala spojitému problému a výsledek vhodně vykreslit. Zkuste něco jako

```
n = 40
kappa = 400

( sezI, sezJ, sezX, sezY ) = seznamyBodu(n)


A = maticeSoustavy(kappa/n**2, sezI, sezJ)
b = pravaStranaSoustavy(sezX, sezY)

amplitudy = np.linalg.solve(A,b)

maxAmplituda = max(amplitudy.max(),-amplitudy.min() )

plt.figure(figsize=(10, 8), dpi=80)
plt.axis('equal')
plt.scatter(sezX,sezY,c=amplitudy,cmap='Spectral',vmax=maxAmplituda,vmin=-maxAmplituda,s=(220/n)**2, marker="s")
plt.colorbar()

plt.show()
```

Výsledkem bude zajímavý obrázek. Pro jiné hodnoty parametrů pak dopadne třeba takto



Obrázek 3. Amplitudy oscilací pro $n = 130, \kappa = 690$.

8. Odpovězta na následující dotazy:

1. Jaký význam mají jednostlivé parametry `plt.scatter(...)`? Použijte dokumentaci knihovny `matplotlib`.
2. K čemu slouží `plt.axis('equal')`?
3. Experimentálně změřte, s jakou mocninou n velmi zhruba roste čas výpočtu. Popište a zkuste odůvodnit nalezené vysledky.

Řešení odevzdejte jako `*.ipynb` soubor na stránce k tomu určené. Email používejte pro kladední dotazů, místo větších příloh však raději posílejte odkazy na sdílené sešity na `colab.research.google.com/` atp.