

```

1
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import random as rd
5
6 def distance(x1,x2):
7     '''distance euclidienne entre deux points'''
8     d=0
9     for i in range(len(x1)):
10         d+=(x1[i]-x2[i])**2
11     return np.sqrt(d)
12
13 def centre(liste):
14     '''calcule l'isobarycentre de la liste de points'''
15     p=len(liste) #le nombre de points
16     #p ne pourra pas être nul car il y aura toujours au moins le centre
17     n=len(liste[0]) #la dimension de nos points
18
19     c=[] #pour stocker les coordonnées du centre
20     for i in range(n):#calcul de la ième coordonnée du centre
21         s=0
22         for j in range(p):
23             s+=liste[j][i]
24         c.append(s/p)
25     return c
26
27 def debut(liste,k):
28     '''choisit au hasard k points parmi la liste'''
29     #attention rd.choices(liste,k=k) ne fonctionne pas car il peut y avoir des
30     #répétitions
31     choix=[]
32     while len(choix)<k:
33         x=rd.choice(liste)
34         if x not in choix:
35             choix.append(x)
36     return choix
37
38 def clusters(Lc,Lx):
39     '''Lc est la liste des centres, Lx la liste des points,retourne la liste des
40     clusters'''
41     k=len(Lc) #le nombre de clusters
42     liste_clusters=[[ ] for i in range(k)]
43     for point in Lx: #on va affecter chaque point à un cluster
44         i_min=0
45         d_min=np.inf
46         for i in range(k):
47             d=distance(point,Lc[i])
48             if d<d_min:
49                 i_min=i
50                 d_min=d
51         liste_clusters[i_min].append(point)
52     return liste_clusters
53
54 def arret(Lc_old,Lc_new,epsilon):
55     k=len(Lc_old)
56     test=True
57     for i in range(k):
58         if distance(Lc_old[i],Lc_new[i])>epsilon:
59             test=False
60     #le test sera encore True si toutes les distances ont été plus petites que epsilon
61     return test
62
63 def kmeans(Lx,k):
64     Lc=debut(Lx,k) #on initialise aléatoirement les centres
65     test=False
66     compteur=0 #pour savoir combien d'itérations seront nécessaires
67     while test==False:
68         compteur+=1
69         liste_clusters=clusters(Lc,Lx) #la liste des clusters
70
71

```

```

71     #on va recalculer tous les centres
72     Lc_new=[]
73     for liste in liste_clusters:
74         Lc_new.append(centre(liste))
75
76     test=arret(Lc,Lc_new,0.01)
77
78     Lc=Lc_new #on actualise les centres
79
80     return Lc,liste_clusters,compteur
81
82
83
84
85
86 def kmeans_tracé(Lx,k):
87     numéro=0 #pour numéroter les figures sauvegardées
88     #un choix de k couleurs au hasard
89     liste_couleurs = [[rd.random(),rd.random(),rd.random()] for i in range(k)]
90
91     Lc=debut(Lx,k) #on initialise aléatoirement les centres
92
93     #tracé de tous les individus et des premiers centres
94     liste_x=[coord[0] for coord in Lx]
95     liste_y=[coord[1] for coord in Lx]
96     plt.plot(liste_x,liste_y,'ob')
97     listec_x=[coord[0] for coord in Lc]
98     listec_y=[coord[1] for coord in Lc]
99     plt.scatter(listec_x,listec_y,s=300,marker='*',color='red')
100    plt.savefig('étape'+str(numéro)+'.png')
101    plt.show()
102    numéro+=1
103
104    test=False
105    compteur=0 #pour savoir combien d'itérations seront nécessaires
106    while test==False and compteur<3:
107
108        compteur+=1
109        liste_clusters=clusters(Lc,Lx) #la liste des clusters
110
111        #le tracé
112        i=0
113        for liste in liste_clusters:
114            couleur=liste_couleurs[i]
115            liste_x=[coord[0] for coord in liste]
116            liste_y=[coord[1] for coord in liste]
117            plt.plot(liste_x,liste_y,'o',color=couleur)
118            i+=1
119        listec_x=[coord[0] for coord in Lc]
120        listec_y=[coord[1] for coord in Lc]
121        plt.scatter(listec_x,listec_y,s=300,marker='*',color='red')
122        plt.savefig('étape'+str(numéro)+'.png')
123        plt.show()
124
125        numéro+=1
126
127        #on va recalculer tous les centres
128        Lc_new=[]
129        for liste in liste_clusters:
130            Lc_new.append(centre(liste))
131
132
133        test=arret(Lc,Lc_new,0.01)
134
135        Lc=Lc_new #on actualise les centres
136
137        #le tracé
138        i=0
139        for liste in liste_clusters:
140            couleur=liste_couleurs[i]
141            liste_x=[coord[0] for coord in liste]
142            liste_y=[coord[1] for coord in liste]

```

```

143         plt.plot(liste_x,liste_y,'o',color=couleur)
144         i+=1
145         listec_x=[coord[0] for coord in Lc]
146         listec_y=[coord[1] for coord in Lc]
147         plt.scatter(listec_x,listec_y,s=300,marker='*',color='red')
148         plt.savefig('étape'+str(numéro)+'.png')
149         plt.show()
150         numéro+=1
151
152     print(compteur)
153
154     #on récupère un nuage de points pour tester notre programme
155     points_ex=[[24, 6],
156               [31, 27],
157               [47, 18],
158               [11, 1],
159               [13, 0],
160               [47, 30],
161               [45, 37],
162               [47, 38],
163               [38, 33],
164               [41, 39],
165               [22, 46],
166               [8, 32],
167               [55, 36],
168               [13, 33],
169               [53, 0],
170               [51, 20],
171               [4, 38],
172               [3, 41],
173               [43, 32],
174               [34, 49],
175               [49, 37],
176               [46, 19],
177               [50, 37],
178               [43, 27],
179               [15, 40],
180               [11, 36],
181               [29, 25],
182               [57, 35],
183               [26, 6],
184               [58, 29],
185               [33, 7],
186               [29, 7],
187               [59, 24],
188               [1, 2],
189               [17, 15],
190               [46, 21],
191               [48, 2],
192               [6, 37],
193               [18, 12],
194               [43, 41],
195               [36, 35],
196               [6, 5],
197               [32, 43],
198               [12, 17],
199               [4, 46],
200               [50, 28],
201               [19, 19],
202               [57, 22],
203               [26, 19],
204               [12, 24],
205               [94, 19],
206               [69, 13],
207               [92, 20],
208               [67, 1],
209               [100, 17],
210               [74, 1],
211               [65, 20],
212               [81, 16],
213               [72, 6],
214               [78, 16],

```

215 [89, 22],
216 [72, 20],
217 [97, 4],
218 [82, 2],
219 [69, 19],
220 [81, 22],
221 [67, 5],
222 [69, 0],
223 [100, 21],
224 [82, 14],
225 [92, 7],
226 [73, 14],
227 [69, 3],
228 [72, 25],
229 [87, 3],
230 [81, 24],
231 [93, 16],
232 [70, 22],
233 [82, 21],
234 [76, 10],
235 [69, 15],
236 [89, 16],
237 [88, 25],
238 [65, 17],
239 [79, 22],
240 [73, 18],
241 [79, 3],
242 [69, 3],
243 [84, 11],
244 [77, 0],
245 [69, 23],
246 [90, 4],
247 [74, 9],
248 [72, 2],
249 [91, 19],
250 [65, 4],
251 [74, 1],
252 [96, 5],
253 [80, 5],
254 [92, 20],
255 [50, 45],
256 [36, 49],
257 [58, 75],
258 [13, 75],
259 [29, 58],
260 [31, 20],
261 [46, 86],
262 [31, 95],
263 [56, 2],
264 [47, 71],
265 [53, 32],
266 [30, 32],
267 [59, 99],
268 [27, 60],
269 [59, 88],
270 [10, 99],
271 [35, 35],
272 [34, 72],
273 [12, 85],
274 [25, 50],
275 [52, 69],
276 [35, 80],
277 [27, 85],
278 [53, 47],
279 [55, 29],
280 [52, 87],
281 [12, 85],
282 [57, 1],
283 [57, 95],
284 [46, 94],
285 [36, 74],
286 [14, 45],

```
287 [51, 66],
288 [46, 15],
289 [33, 60],
290 [29, 43],
291 [29, 34],
292 [52, 69],
293 [59, 91],
294 [11, 94],
295 [52, 45],
296 [34, 35],
297 [27, 70],
298 [38, 73],
299 [23, 82],
300 [24, 45],
301 [60, 31],
302 [47, 88],
303 [10, 36],
304 [15, 40]]
305
306 kmeans_tracé(points_ex,3)
```