Code Test

1. Different methods to initialize Agents

This model used a class to initialize agents in the most original version, but finally, I use a list to represent each agent and write all the code in one python file to improve code integrity so now we only need one file to do everything.

————Source code in Version 1.0————

class Agent:

def \_\_init\_\_(self,environment,agents,x,y,z):

self.environment = environment

self.agents = agents

self.x = 50

self.y = 150

self.z = 75

def move(self):

b = r.random()

if b < 0.05:

self.x = self.x - 1

elif 0.05 <= b <= 0.15:

self.y = self.y - 1

elif 0.15 < b <=0.25:

self.y = self.y + 1

elif 0.25 < b <= 1.0:

self.x = self.x + 1

def drop(self):

a = r.random()

if self.z >= 75:

if a < 0.20:

self.z = self.z + 1

if 0.20 <= a < 0.3:

self.z = self.z

if 0.3 <= a < 1.0:

self.z -= 1

elif 0 < self.z < 75:

self.z = self.z - 1

————Source code in final Version————

Please refer to the code in the ‘run.py’ file.

1. Improvement of reading and files

At first I used ‘CSV’ to read the raster data file, but then I found out that using ‘pandas’ to read the file would be more effective and I converted the data into an array for subsequent mapping.

————Source code in Version 1.0————

import csv

f = open('wind.txt', newline='')

reader = csv.reader(f, quoting=csv.QUOTE\_NONNUMERIC)

for row in reader:

parsed\_row = row

rowlist = []

for value in parsed\_row:

rowlist.append(value)

environment.append(rowlist)

f.close()

————Source code in final Version————

import pandas as pd

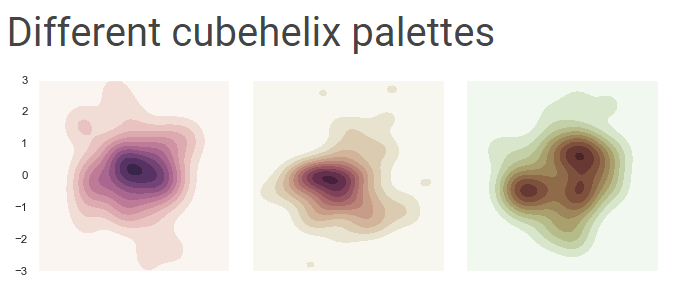
import numpy as np

img\_train = pd.read\_csv('wind.txt',header=None)

img\_train = np.array(img\_train)

1. Different methods for displaying density maps

At first, I only use scatter plot to display the distribution of particles, and then I add a cubehelix palettes to display them. More information about cubehelix palettes can be found at <https://seaborn.pydata.org/examples/cubehelix_palette.html>



————Source code in Version 1.0————

import matplotlib.pyplot

for i in range(num\_agents):

matplotlib.pyplot.scatter(agents[i].x,agents[i].y)

matplotlib.pyplot.show()

————Source code in final Version————

import seaborn as sns

X=[]

Y=[]

for x in range(img\_train.shape[0]):

for y in range(img\_train.shape[1]):

if img\_train[x,y]!=0:

X.append(x)

Y.append(y)

fig = plt.figure(figsize=(10, 10))

ax = sns.kdeplot(X,Y, shade = True, cmap = "PuBu")

1. Improvement of displaying particles

In order to make the particles clearly distinguishable, I designed a list to let the particles get different colors

————Source code in final Version————

c=['r','c','g','b','r','y','g','b','m']# A list represents different colours

t=[]

for i in range(len(y)):

t.append(random.choice(c))