from scipy.interpolate import interp1d  
import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
def read\_csv\_to\_dict(csv\_path, column\_name):  
 sea\_data = pd.read\_csv(csv\_path)  
 data\_dict = {'sea\_pressure': sea\_data['Sea pressure - decibar'].values,  
 'sea\_salinity': sea\_data[column\_name].values}  
 return data\_dict  
  
def get\_salinity\_value\_from\_dict(data\_dict, pressure\_value):  
 salinity\_interp = interp1d(data\_dict['sea\_pressure'], data\_dict['sea\_salinity'], kind='linear', fill\_value='extrapolate')  
 salinity\_value = salinity\_interp(pressure\_value)  
 return salinity\_value  
  
def calculate\_density(depth, salinity\_dict, temperature\_dict):  
 pressure\_value = depth \* 0.10045  
 salinity\_value = get\_salinity\_value\_from\_dict(salinity\_dict, pressure\_value)  
 temperature\_value = get\_salinity\_value\_from\_dict(temperature\_dict, pressure\_value)  
 density\_value = temperature\_value \* (-173.4852) + salinity\_value \* 114.9477 + 1025  
 if depth > 2000:  
 return 2928.716731784  
 return density\_value  
  
def get\_density\_interpolator(salinity\_dict, temperature\_dict):  
 depth\_values = np.arange(0, 6001, 10)  
 density\_values = [calculate\_density(depth, salinity\_dict, temperature\_dict) for depth in depth\_values]  
  
 min\_density, max\_density = 1025, 1070  
 density\_range = max\_density - min\_density  
  
 # 修改这里，使用 np.linspace 生成递增的压缩密度值  
 compressed\_density = min\_density + density\_range \* (np.linspace(0, 1, len(depth\_values)) - min(np.linspace(0, 1, len(depth\_values)))) / (max(np.linspace(0, 1, len(depth\_values))) - min(np.linspace(0, 1, len(depth\_values))))  
  
 density\_interpolator = interp1d(depth\_values, density\_values, kind='cubic', fill\_value='extrapolate')  
  
 return density\_interpolator  
  
  
  
  
# 提供一个深度值，例如 3000 米  
# desired\_depth = 1077.88  
#  
# # 提供盐度和温度 CSV 文件的路径  
# salinity\_csv\_path = "D:\\数模\\argo\_db\\linear\_90382204\_70-28.csv"  
# temperature\_csv\_path = "D:\\数模\\argo\_db\\linear\_90382204\_68-28.csv"  
#  
# # 读取 CSV 数据为字典  
# salinity\_dict = read\_csv\_to\_dict(salinity\_csv\_path, 'Practical salinity adjusted - psu')  
# temperature\_dict = read\_csv\_to\_dict(temperature\_csv\_path, 'Sea temperature adjusted - degree\_Celsius')  
#  
# # 调用函数获取给定深度的压缩后密度值  
# result\_density = get\_density\_at\_depth(desired\_depth, salinity\_dict, temperature\_dict)  
  
# print(f"The density at depth {desired\_depth} is: {result\_density}")  
# 取消注释并运行以下代码以生成并保存插值器  
# desired\_depth = 1077.88  
salinity\_csv\_path = "D:\\MCM\_ICM\\argo\_db\\linear\_90382204\_70-28.csv"  
temperature\_csv\_path = "D:\\MCM\_ICM\\argo\_db\\linear\_90382204\_68-28.csv"  
salinity\_dict = read\_csv\_to\_dict(salinity\_csv\_path, 'Practical salinity adjusted - psu')  
temperature\_dict = read\_csv\_to\_dict(temperature\_csv\_path, 'Sea temperature adjusted - degree\_Celsius')  
# 读取 CSV 数据为字典  
salinity\_dict = read\_csv\_to\_dict(salinity\_csv\_path, 'Practical salinity adjusted - psu')  
temperature\_dict = read\_csv\_to\_dict(temperature\_csv\_path, 'Sea temperature adjusted - degree\_Celsius')  
  
# 获取整个函数的插值器  
density\_interpolator = get\_density\_interpolator(salinity\_dict, temperature\_dict)  
  
# 保存插值器至文件  
import pickle  
  
def save\_interpolator(interpolator, file\_path):  
 with open(file\_path, 'wb') as file:  
 pickle.dump(interpolator, file)  
# 保存插值器至文件  
# 保存插值器至文件  
save\_interpolator(density\_interpolator, 'density\_interpolator.pkl')  
#  
# test\_depths = [0, 1000, 2000]  
#  
# # 打印测试点的深度和插值器计算的密度值  
# for depth in test\_depths:  
# density\_value = density\_interpolator(depth)  
# print(f"Depth: {depth} m, Density: {density\_value} kg/m^3")  
  
# 取消注释并运行以下代码以加载插值器并使用它  
# loaded\_interpolator = load\_interpolator('density\_interpolator.pkl')  
# result\_density = loaded\_interpolator(desired\_depth)  
# print(f"The density at depth {desired\_depth} is: {result\_density}")  
  
  
depth\_values = np.arange(0, 6001, 10)  
  
# 计算每个深度对应的密度值  
density\_values = density\_interpolator(depth\_values)  
  
# 绘制插值器函数图像  
plt.figure(figsize=(10, 6))  
plt.plot(depth\_values, density\_values, label='Interpolated Density Function', color='blue')  
plt.scatter(depth\_values, density\_values, c='red', s=5, label='Sampled Data Points') # 显示采样数据点  
plt.xlabel('Depth (m)')  
plt.ylabel('Density (kg/m^3)')  
plt.title('Interpolated Density Function')  
plt.legend()  
plt.grid(True)  
plt.show()