張博士的微信頭像定位問題求解

StarKing

July 01, 2021

目录

1 問題描述

已知某點 P 距離北京 2490 km, 距離東京 3550 km, 距離香港 6?? km, 求該點 P 的位置.

2 定位模型建立

如圖 1所示, 建立直角三維坐標系. 則地球上一點 P 與球心 O 構成的向量 $\overrightarrow{OP}$ 坐標為:

$$\overrightarrow{OP} = R(\cos\theta\cos\varphi, \cos\theta\sin\varphi, \sin\theta) \tag{2.1}$$

不妨設北京位於 A 點, 設東京位於 B 點, 香港位於 C 點, 則三個矢量為:

$$\overrightarrow{OA} = R(\cos\theta_1\cos\varphi_1, \cos\theta_1\sin\varphi_1, \sin\theta_1) \tag{2.2}$$

$$\overrightarrow{OB} = R(\cos\theta_2\cos\varphi_2, \cos\theta_2\sin\varphi_2, \sin\theta_2) \tag{2.3}$$

$$\overrightarrow{OC} = R(\cos\theta_3\cos\varphi_3, \cos\theta_3\sin\varphi_3, \sin\theta_3) \tag{2.4}$$

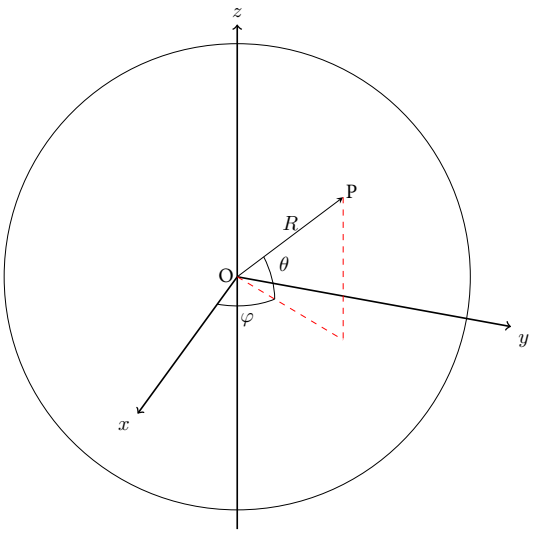

图 1: 地球上一點 P 在三維坐標系的坐標. 其中 O 為球心，R 為地球半徑,φ 為經度,θ 為緯度.

設 P 分別與 A, B, C 點的距離為:d_{PA}, d_{PB}, d_{PC}. 則：

$$d_{PA} = R\cos\alpha \tag{2.5}$$

其中 α 為向量 $\overrightarrow{OP}$ 與向量 $\overrightarrow{OA}$ 的夾角, 如圖 2所示. 夾角 α 可根據兩向量的內積求得:

$$\cos\alpha = \frac{\overrightarrow{OP}\cdot\overrightarrow{OA}}{|\overrightarrow{OP}|\cdot|\overrightarrow{OA}|} \tag{2.6}$$

根據方程 eq(2.1),eq(2.2),eq(2.5),eq(2.6), 求得：

$$d_{PA} = R\arccos[\cos\theta\cos\theta_1\cos(\varphi-\varphi_1) + \sin\theta\sin\theta_1] \tag{2.7}$$

同理, 有:

$$d_{PB} = R\arccos[\cos\theta\cos\theta_2\cos(\varphi-\varphi_2) + \sin\theta\sin\theta_2] \tag{2.8}$$

$$d_{PC} = R\arccos[\cos\theta\cos\theta_3\cos(\varphi-\varphi_3) + \sin\theta\sin\theta_3] \tag{2.9}$$

由於 d_{PA}, d_{PB} 已知,d_{PC} 未知. 故先改寫方程 eq(2.7) 與 eq(2.8) 得到待求的點 P 的經緯度 θ, φ 一個非綫性方程組:

$$\begin{cases} f_1(\theta,\varphi) = \cos\theta\cos\theta_1\cos(\varphi-\varphi_1) + \sin\theta\sin\theta_1 - \cos\frac{d_{PA}}{R} = 0 \\ f_2(\theta,\varphi) = \cos\theta\cos\theta_2\cos(\varphi-\varphi_2) + \sin\theta\sin\theta_2 - \cos\frac{d_{PB}}{R} = 0 \end{cases} \tag{2.10}$$

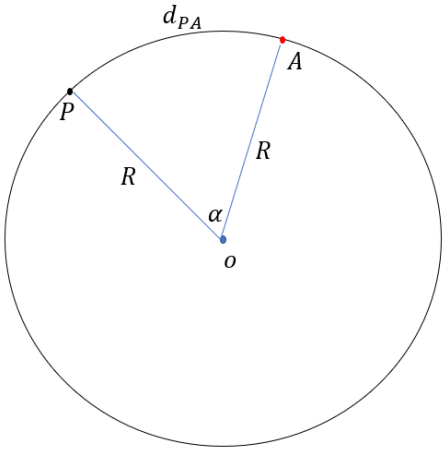

图 2: 點 P 與點 A 的球面距離為 $d_{PA} = R\alpha$.

通過求解該非綫性方程組即可求得點 P 的位置. 可采用牛頓迭代法求解該非綫性方程組.

不妨將該方程組記作向量形式 $\mathbf{F} = (f_1, f_2)^T = \mathbf{0}$, 則向量函數 $\mathbf{F}$ 的雅可比矩陣為:

$$\mathbf{J} = \begin{pmatrix} \frac{\partial f_1}{\partial \theta} & \frac{\partial f_1}{\partial \varphi} \\ \frac{\partial f_2}{\partial \theta} & \frac{\partial f_2}{\partial \varphi} \end{pmatrix} = \begin{pmatrix} -\sin\theta\cos\theta_1\cos(\varphi-\varphi_1) + \cos\theta\sin\theta_1 & -\sin(\varphi-\varphi_1)\cos\theta\cos\theta_1 \\ -\sin\theta\cos\theta_2\cos(\varphi-\varphi_2) + \cos\theta\sin\theta_2 & -\sin(\varphi-\varphi_2)\cos\theta\cos\theta_2 \end{pmatrix} \tag{2.11}$$

牛頓迭代法的表達式為:

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \mathbf{J}(\mathbf{x}^k)^{-1}\mathbf{F}(\mathbf{x}^k), \quad k = 0, 1, 2, \ldots, .. \tag{2.12}$$

其中 $\mathbf{x} = (\theta, \varphi)^T$,$k$ 為迭代次數.

3 問題求解

易知北京的經緯度為 $(\theta_1, \varphi_1) = (39.9042°N, 116.4074°E)$, 東京的經緯度為 $(\theta_2, \varphi_2) = (35.6804°N, 139.7690°E)$, 香港的經緯度為 $(\theta_3, \varphi_3) = (22.3193°N, 114.1694°E)$, 地球半徑 R=6378.1370 km, d_{PA}=2490 km, d_{PB}=3550 km, d_{PC}=6?? km.

此時求得函數 f_1, f_2 的等值綫圖如圖 3所示. 兩個函數的零等值綫圖交於兩點 D, E.

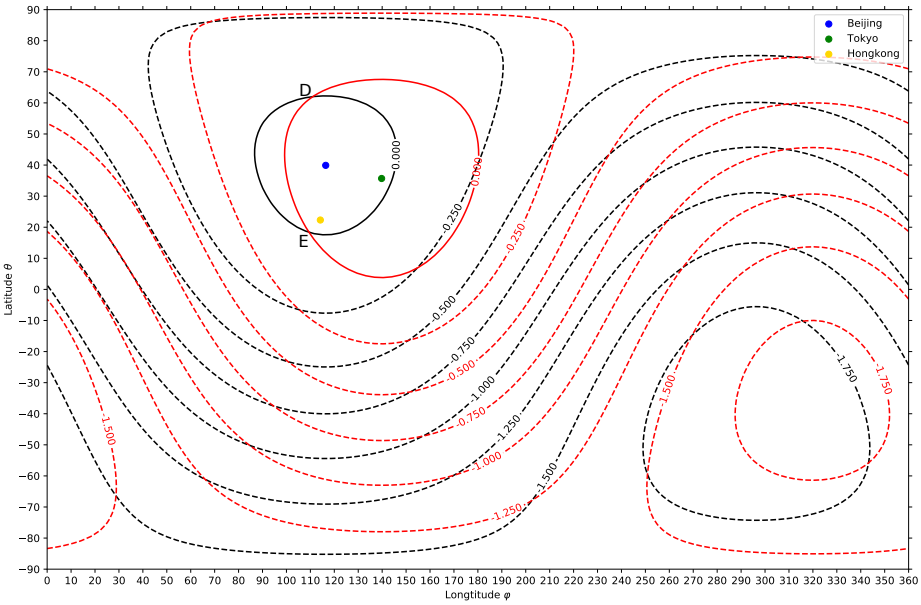

图 3: 函數 f_1, f_2 的等值綫圖; 零值曲綫交與兩點 D, E.

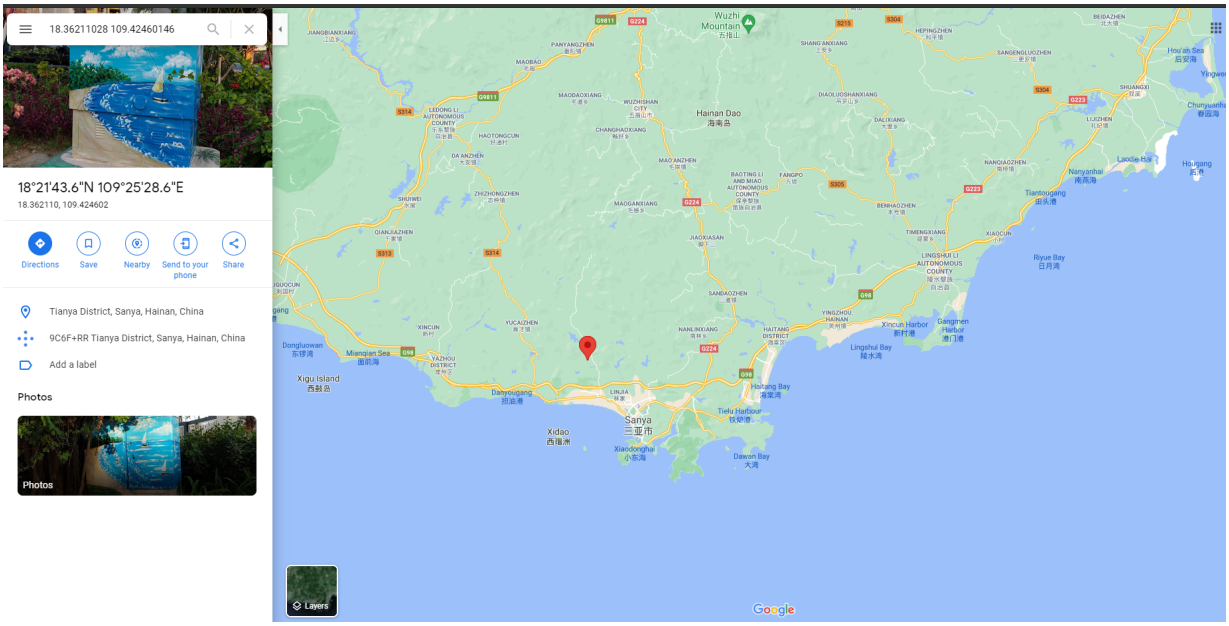

图 4: 點 P 在地圖的位置

通過牛頓迭代法求得這兩點的經緯度分別為:

$$D(62.03072717^\circ N, 110.99207792^\circ E), \quad E(18.36211028^\circ N, 109.42460146^\circ E) \tag{3.1}$$

將上述兩點帶入方程 eq(2.9), 求得:

$$d_{DC} = 4427.309517001271km, \qquad d_{EC} = 662.6996866644358km \tag{3.2}$$

而待求的點 P 到香港 C 的距離為 $d_{PC} = 6??km$, 與上述兩個距離比較, 則點 E 即為待求的點 P.

故點 P 的經緯度為 $(\theta, \varphi) = (18.36211028^\circ N, 109.42460146^\circ E)$.

如圖 4所示, 點 P 位於三亞市天涯區.

4 附錄：PYTHON 代碼

數值求解的 PYTHON 代碼如下:

```
import numpy as np
import matplotlib.pyplot as plt
import math
from matplotlib.ticker import MultipleLocator,
    FormatStrFormatter,FuncFormatter,LinearLocator

def Newton_system(F, J, x, epsilon):
    F_value = F(x)
    F_norm = np.linalg.norm(F_value, ord=2) # l2 norm of vector
    iteration_counter = 0
    while abs(F_norm) > epsilon and iteration_counter < 100:
        delta = np.linalg.solve(J(x), -F_value)
        x = x + delta
        F_value = F(x)
        F_norm = np.linalg.norm(F_value, ord=2)
        iteration_counter += 1
        #print(x)
    # Here, either a solution is found, or too many iterations
    if abs(F_norm) > epsilon:
        iteration_counter = -1
    return x, iteration_counter

def distance_sphere(R,theta1,theta2,phi1,phi2):
    delta_theta=math.acos((math.cos(theta1)*math.cos(theta2)*(math.cos(phi1)*math.cos(phi2)
    +math.sin(phi1)*math.sin(phi2))+math.sin(theta1)*math.sin(theta2)))
    distance=R*delta_theta
    return distance

R=6378.1370 # unit:km
# Beijing
theta_beijing=math.radians(39.9042)
phi_beijing=math.radians(116.4074)
# Tokyo
theta_tokyo=math.radians(35.6804)
```

```
phi_tokyo=math.radians(139.7690)

# Hongkong
theta_hongkong=math.radians(22.3193)
phi_hongkong=math.radians(114.1694)

dis_x_beijing=2490
dis_x_tokyo=3550

# Find location
def F(x):
    return np.array([np.cos(x[0])*np.cos(theta_beijing)*np.cos(x[1]-phi_beijing)
    +np.sin(x[0])*np.sin(theta_beijing)-np.cos(dis_x_beijing/R)
    ,np.cos(x[0])*np.cos(theta_tokyo)*np.cos(x[1]-phi_tokyo)+np.sin(x[0])
    *np.sin(theta_tokyo)-np.cos(dis_x_tokyo/R)])

def J(x):
    return
        np.array([[-np.sin(x[0])*np.cos(theta_beijing)*np.cos(x[1]-phi_beijing)
    +np.cos(x[0])*np.sin(theta_beijing),-np.sin(x[1]-phi_beijing)
    *np.cos(x[0])*np.cos(theta_beijing)],[-np.sin(x[0])*np.cos(theta_tokyo)
    *np.cos(x[1]-phi_tokyo)+np.cos(x[0])*np.sin(theta_tokyo),
    -np.sin(x[1]-phi_tokyo)*np.cos(x[0])*np.cos(theta_tokyo)]])

x0=np.array([0,2.5])
epsilon=0.0001
x,n=Newton_system(F, J, x0, epsilon)
print('lat_long:',x*180/np.pi)
distance_x_hongkong=distance_sphere(R,x[0],theta_hongkong,x[1],phi_hongkong)
print('distance_x_kongkong[km]:',distance_x_hongkong)

theta=np.linspace(-np.pi/2,np.pi/2,101)
phi=np.linspace(0,2*np.pi,101)
THETA,PHI=np.meshgrid(theta,phi)
F1=np.cos(THETA)*np.cos(theta_beijing)*np.cos(PHI-phi_beijing)
+np.sin(THETA)*np.sin(theta_beijing)-np.cos(dis_x_beijing/R)
F2=np.cos(THETA)*np.cos(theta_tokyo)*np.cos(PHI-phi_tokyo)
+np.sin(THETA)*np.sin(theta_tokyo)-np.cos(dis_x_tokyo/R)
```

```
fig, ax = plt.subplots(figsize=(15,10))
plt.scatter(phi_beijing*180/np.pi,theta_beijing*180/np.pi,c='blue',label='Beijing')
plt.scatter(phi_tokyo*180/np.pi,theta_tokyo*180/np.pi,c='green',label='Tokyo')
plt.scatter(phi_hongkong*180/np.pi,theta_hongkong*180/np.pi,c='gold',label='Hongkong')
CS = ax.contour(PHI*180/np.pi,THETA*180/np.pi,F1,colors='black')
ax.clabel(CS, inline=True, fontsize=10)
CS1=ax.contour(PHI*180/np.pi,THETA*180/np.pi,F2,colors='red')
ax.clabel(CS1, inline=True, fontsize=10)
ax.set_xlabel('Longtitude $\\varphi$')
ax.set_ylabel('Latitude $\\theta$')
ax.xaxis.set_ticks_position('both')
ax.yaxis.set_ticks_position('both')
xmajorLocator = MultipleLocator(10)
ymajorLocator = MultipleLocator(10)
ax.xaxis.set_major_locator(xmajorLocator)
ax.yaxis.set_major_locator(ymajorLocator)
ax.legend()
plt.savefig('location.pdf',format='pdf',dpi=300)
plt.show()
```