Odwzorowanie Gaussa-Krügera: układy współrzędnych płaskich stosowanych w Polsce.

Autor: Michał Ambroży

Grupa I

Nr: 1

Wstęp

Celem tego ćwiczenia było zaznajomienie się z praktycznymi aspektami geodezji, w tym z transformacją współrzędnych, redukcją odwzorowawczą oraz analizą różnic w pomiarach na elipsoidzie i płaszczyźnie. Poprzez praktyczne przykłady, takie jak długości odcinków na elipsoidzie oraz pole trapezu na różnych układach współrzędnych, ćwiczenie miało na celu poszerzenie wiedzy z zakresu geodezji w tej dziedzinie.

Przebieg

Zaimportowałem wymagane biblioteki, wpradziłem moje obliczone punkty z projektu 3 oraz dodkonąc odpowiednich transformacji i stosując odpowiednie układ współrzędnych obliczyłem współrdzędne punktów w układach PL -1992 oraz PL - 2000. Następnie porównałem je ze współrzędnymi punktów z poprzedneigo projektu.

Następnie zajmowałem się redukcjami długości odcinków z płaszczyzny Gaussa-Krügera na elipsoidę. Tutaj kluczowym elementem było uwzględnienie średnich promieni krzywizny dla poszczególnych odcinków. Porównałem wyniki z długościami z poprzedniego projektu.

W końcowej części zadania obliczyłem pole trapezu na dwóch różnych płaszczyznach - PL-1992 i PL-2000. Przy okazji, przeliczyłem również to pole do układu PL-LAEA. Różnice w polach wynikają z odmiennych odkształceń powierzchni na różnych płaszczyznach.

Zadanie 1. Przeliczenie współrzędnych z ćwiczenia 3. do układów PL-1992 i PL-2000

In [ ]:
```
import numpy as np
from pyproj import Proj, transform, Geod, CRS
from pyproj import Transformer

input_projection = CRS.from_epsg(4326)
output_projection = CRS.from_epsg(2180)
trans_2000 = Transformer.from_proj(input_projection, output_projection)

# Współrzędne punktów z zadania 3
# Punkt 1
phi_1 = 50 + 15/60
lambda_1 = 18 + 15/60

nr = 1
# dane elipsoidy
a = 6378137
e = 0.0818191910428158
e2 = e**2

# Wielkości pomocnicze
m0 = 0.999923

# Współrzędne punktów w układzie PL-2000
geod = Geod(ellps='WGS84')

# Kolejne długości i azymuty linii geodezyjnych:
# długość s [m] azymut A [◦]
s1_2 = 40000
s2_3 = 100000
s3_4 = 40000
s4_1 = 100000

A1_2 = 0
```

```
A2_3 = 90
A3_4 = 180
A4_1 = 270

# Kolejne punkty
p1 = geod.fwd(lambda_1,phi_1, 0, 0)
p2 = geod.fwd(lambda_1,phi_1,A1_2,s1_2)
p3 = geod.fwd(p2[0],p2[1],A2_3,s2_3)
p4 = geod.fwd(p3[0],p3[1],A3_4,s3_4)
p5 = geod.fwd(p4[0],p4[1],A4_1,s4_1)

# Transformacja współrzędnych geodezyjnych na płaszczyźnie Gaussa-Krügera
p1_2000 = trans_2000.transform(p1[1], p1[0])
p2_2000 = trans_2000.transform(p2[1], p2[0])
p3_2000 = trans_2000.transform(p3[1], p3[0])
p4_2000 = trans_2000.transform(p4[1], p4[0])

points_2000 = [p1_2000, p2_2000, p3_2000, p4_2000] # współrzędne punktów w układzie PL-2000

# Współrzędne punktów w układzie PL-1992
p1_1992 = p1_2000[0] - 5300000, p1_2000[1] - 1000000
p2_1992 = p2_2000[0] - 5300000, p2_2000[1] - 1000000
p3_1992 = p3_2000[0] - 5300000, p3_2000[1] - 1000000
p4_1992 = p4_2000[0] - 5300000, p4_2000[1] - 1000000

points_1992 = [p1_1992, p2_1992, p3_1992, p4_1992] # tablica współrzędnych punktów w układzie PL-1992

# wypiszmy w po kolei współrzędne punktów
for i in range(len(points_1992)):
    print(f'Punkt {i+1}:\nPL-2000: {points_2000[i]}\nPL-1992: {points_1992[i]}\n')
```

```
Punkt 1:
PL-2000: (265025.8454874512, 446545.25895352184)
PL-1992: (-5034974.154512549, -553454.7410464782)

Punkt 2:
PL-2000: (304997.2039412623, 446948.6057136208)
PL-1992: (-4995002.796058738, -553051.3942863792)

Punkt 3:
PL-2000: (303986.2751140101, 546874.5259955525)
PL-1992: (-4996013.72488599, -453125.47400444746)

Punkt 4:
PL-2000: (264014.77496792376, 547230.7980176203)
PL-1992: (-5035985.225032076, -452769.20198237966)
```

ZAD.2

In [ ]:
```python
# 2. Redukcje odwzorowawcze
s1_2_2000 = np.sqrt((p2_2000[0] - p1_2000[0])**2 + (p2_2000[1] - p1_2000[1])**2) # długość odcinka 1-2 w układzie PL-2000
s2_3_2000 = np.sqrt((p3_2000[0] - p2_2000[0])**2 + (p3_2000[1] - p2_2000[1])**2) # długość odcinka 2-3 w układzie PL-2000
s3_4_2000 = np.sqrt((p4_2000[0] - p3_2000[0])**2 + (p4_2000[1] - p3_2000[1])**2) # długość odcinka 3-4 w układzie PL-2000
s4_1_2000 = np.sqrt((p1_2000[0] - p4_2000[0])**2 + (p1_2000[1] - p4_2000[1])**2) # długość odcinka 4-1 w układzie PL-2000

# Redukcja długości z płaszczyzny układu PL-2000 na elipsoidę:
sgk1_2 = s1_2_2000/m0
sgk2_3 = s2_3_2000/m0
sgk3_4 = s3_4_2000/m0
sgk4_1 = s4_1_2000/m0

# współrzędne geodezyjne środkowego punktu odcinka
phi_m1_2 = (phi_1 + p2[1])/2
phi_m2_3 = (p2[1] + p3[1])/2
phi_m3_4 = (p3[1] + p4[1])/2
phi_m4_1 = (p4[1] + phi_1)/2

# średni promień krzywizny dla odcinka - funkcja pomocnicza
def Rm_value(phi, a, e) -> tuple:
    phi_rad = np.radians(phi)
```

```
    M = a * (1 - e ** 2) / np.sqrt((1 - e ** 2 * np.sin(phi_rad) ** 2) ** 3)
    N = a / np.sqrt(1 - e ** 2 * np.sin(phi_rad) ** 2)
    R = M / N
    m = N / np.sqrt(1 - e ** 2 * np.sin(phi_rad) ** 2)
    return R, m

# oblicz średni promień krzywizny dla odcinka
Rm1_2, Rm1_2 = Rm_value(phi_m1_2, a, e)
Rm2_3, Rm2_3 = Rm_value(phi_m2_3, a, e)
Rm3_4, Rm3_4 = Rm_value(phi_m3_4, a, e)
Rm4_1, Rm4_1 = Rm_value(phi_m4_1, a, e)

# oblicz redukcje długości
r1_2 = sgk1_2 * (p1_2000[1]**2 + p1_2000[1]*p2_2000[1] + p2_2000[1]**2)/(6*Rm1_2**2)
r2_3 = sgk2_3 * (p2_2000[1]**2 + p2_2000[1]*p3_2000[1] + p3_2000[1]**2)/(6*Rm2_3**2)
r3_4 = sgk3_4 * (p3_2000[1]**2 + p3_2000[1]*p4_2000[1] + p4_2000[1]**2)/(6*Rm3_4**2)
r4_1 = sgk4_1 * (p4_2000[1]**2 + p4_2000[1]*p1_2000[1] + p1_2000[1]**2)/(6*Rm4_1**2)

# Długość odcinka na elipsoidzie
selip1_2 = sgk1_2 - r1_2
selip2_3 = sgk2_3 - r2_3
selip3_4 = sgk3_4 - r3_4
selip4_1 = sgk4_1 - r4_1

geod = Geod(ellps='GRS80') # elipsoida GRS80

s12_inv = geod.inv(p1[0], p1[1], p2[0], p2[1])[2] # długość odcinka 1-2 na elipsoidzie
s23_inv = geod.inv(p2[0], p2[1], p3[0], p3[1])[2] # długość odcinka 2-3 na elipsoidzie
s34_inv = geod.inv(p3[0], p3[1], p4[0], p4[1])[2] # długość odcinka 3-4 na elipsoidzie
s41_inv = geod.inv(p4[0], p4[1], p1[0], p1[1])[2] # długość odcinka 4-1 na elipsoidzie

print(f'Punkt 1-2:\nZadanie 3: {s1_2_2000}\nZadanie 4: {s12_inv}\nZadanie 5: {selip1_2}\n')
print(f'Punkt 2-3:\nZadanie 3: {s2_3_2000}\nZadanie 4: {s23_inv}\nZadanie 5: {selip2_3}\n')
print(f'Punkt 3-4:\nZadanie 3: {s3_4_2000}\nZadanie 4: {s34_inv}\nZadanie 5: {selip3_4}\n')
print(f'Punkt 4-1:\nZadanie 3: {s4_1_2000}\nZadanie 4: {s41_inv}\nZadanie 5: {selip4_1}\n')
```

```
Punkt 1-2:
Zadanie 3: 39973.39346680417
Zadanie 4: 39999.999999853666
Zadanie 5: 39879.18618247465

Punkt 2-3:
Zadanie 3: 99931.0338247571
Zadanie 4: 100000.00000098243
Zadanie 5: 99636.83412552827

Punkt 3-4:
Zadanie 3: 39973.08786774281
Zadanie 4: 39999.99999985258
Zadanie 5: 39830.291113101404

Punkt 4-1:
Zadanie 3: 100690.61545260095
Zadanie 4: 100760.08984818093
Zadanie 5: 100394.16744866867
```

Obliczenie Pola Trapezu na płaszczyźnie

In [ ]:
```python
# funkcja obliczająca pole trapezu
def trapezoid_area(points: list):
    p = 0
    for i in range(1, len(points) - 1):
        p += points[i][0] * (points[i+1][1] - points[i-1][1])
    p += points[-1][0] * (points[0][1] - points[-2][1])
    p += points[0][0] * (points[1][1] - points[-1][1])
    return abs(p/2)

t_area_1992 = trapezoid_area(points_1992) # pole trapezu w układzie PL-1992
t_area_2000 = trapezoid_area(points_2000) # pole trapezu w układzie PL-2000
pl_laea = CRS.from_epsg(3035) # układ PL-LAEA
laea_transform = Transformer.from_proj(input_projection, pl_laea) # transformacja do układu PL-LAEA

# transformacja współrzędnych punktów do układu PL-LAEA
pl_laea_1 = laea_transform.transform(p1[1], p1[0])
pl_laea_2 = laea_transform.transform(p2[1], p2[0])
```

```
pl_laea_3 = laea_transform.transform(p3[1], p3[0])
pl_laea_4 = laea_transform.transform(p4[1], p4[0])

laea_points = [pl_laea_1, pl_laea_2, pl_laea_3, pl_laea_4] # współrzędne punktów w układzie PL-LAEA
pole_trapezu_laea = trapezoid_area(laea_points) # pole trapezu w układzie PL-LAEA

print(f'Pole figury (trapezu) w układzie PL-1992: {t_area_1992:.2f} m2')
print(f'Pole trapezu (trapezu) w układzie PL-2000: {t_area_2000:.2f} m2')
print(f'Pole trapezu (trapezu) w układzie PL-LAEA: {pole_trapezu_laea:.2f} m2')
```

```
Pole figury (trapezu) w układzie PL-1992: 4009387178.28 m2
Pole trapezu (trapezu) w układzie PL-2000: 4009387178.28 m2
Pole trapezu (trapezu) w układzie PL-LAEA: 4014857711.21 m2
```

Wnioski:

Różnice w polu trapezu w układzie PL-1992: Widoczna jest znaczna rozbieżność w polu trapezu wyznaczonego w układzie PL-1992 w porównaniu do wyników uzyskanych w innych układach. Różnica na poziomie około 2 km2 może być rezultatem większej niedokładności transformacji współrzędnych, spowodowanej jednolitością układu dla całego kraju, co wiąże się z zastosowaniem jednej strefy odwzorowawczej o szerokości 10 stopni.

Porównanie wyników z układów PL-1992 i PL-2000: Wyniki otrzymane w układzie PL-2000 są średnio bardziej zbliżone do wartości obliczonych w poprzednim zadaniu, co sugeruje lepszą zgodność tego układu z danymi geodezyjnymi.

Duże różnice między odległościami: Obserwuje się znaczące rozbieżności między odległościami obliczonymi w zadaniu poprzednim a tymi wyznaczonymi w układach PL-1992 i PL-2000. Im dłuższe są odcinki łączące punkty, tym większe są te różnice.

Potencjalne błędy w obliczeniach: Różnice sięgające 20-50 metrów w obu układach wydają się być zbyt duże, co sugeruje możliwość błędu obliczeniowego lub koncepcyjnego w przeprowadzonych operacjach. Warto podkreślić konieczność dokładnej analizy wyników oraz ewentualnych źródeł błędów.

Zniekształcenia w układzie PL-1992: Możliwe, że znaczące rozbieżności w układzie PL-1992 wynikają z zniekształceń charakterystycznych dla tego układu, co warto uwzględnić przy interpretacji wyników.