PRÁCTICA 4: PCA Y ANALOGÍA

Autor *Martín Fernández de Diego*
Compañera de código *Belén Sánchez Centeno*

1. INTRODUCCIÓN

A partir de 4 archivos NetCDF del re-análisis climatológico NCEP, consideramos la atmósfera como un sistema de 6 variables (t, x, y, p, Z, T) que representan tiempo, longitud, latitud, presión, temperatura y altura geopotencial, respectivamente, donde solo las variables Z y T son dinámicas respecto al tiempo t.

Las variables del sistema están discretizadas. La longitud en 144 valores, la latitud en 73 y la presión en 17.

Apartado i) **Sea el sistema $S = \{a_d, X_d\}_{d=1}^{365}$ formado por los días a_d de 2021 y las variables de estado $X_d := \{Z_{i,j,k}\}_{i=1,j=1,k=1}^{i=144,j=73,k=17}$. Representa las 4 componentes principales fijado p_k a $500hPa$ y di qué porcentaje de varianza se explica.**

Apartado ii) **Sea el subsistema $\sigma \in S$ con valores en la región $x \in (-20°, 20°)$ e $y \in (30°, 50°)$. Encuentra los 4 días de 2021 más análogos al $a_0 =$ 2022-01-11, a través de la distancia euclidea, y calcula el error absoluto medio de $\{T_{i,j,1000hPa}\}_{i=1,j=1}^{i=144,j=73}$ previsto para a_0 según la media de los análogos.**

Adicionalmente, y fuera del guión de la práctica, **se trasladará la escala de longitud** para trabajar sobre la visión usual de los mapas —donde España está centrada— y **se realizará una comprobación visual** de la corrección del resultado de predicción de temperatura para el día dado obtenido en el segundo apartado.

2. MATERIAL USADO

La estructura lógica de la práctica mantiene un carácter secuencial dada su especificidad, en lugar del modular de las anteriores.

Antes de comenzar, trasladamos la escala de longitud de los mapas de $(0, 2\pi)$ a $(-\pi, \pi)$ mediante la función `lons_normal_ref(dataset)`. Basta desplazar los datos una distancia π. Para ello, intercambiamos los datos de las longitudes de ambas mitades del intervalo, moviendo así el origen —el meridiano de Greenwich— a la mitad del vector.

2.1. Apartado i)

Dados los datos de altura geopotencial de 2021, tomamos una submatriz X cuyo nivel de presión sea $500hPa$. Aplicamos el análisis de componentes principales (PCA), tanto a X como a su transpuesta $Y = X^T$, y estudiamos cuál es más beneficiosa en cuanto a explicación de varianza.

El **análisis de componentes principales** interpreta la matriz proporcionada como un conjunto de vectores, evoluciones temporales de un elemento origen, obtiene la matriz de covarianza y calcula sus autovalores y autovectores ordenándolos de mayor a menor según los autovalores. Finalmente, selecciona el número de componentes deseadas por orden.

Por último, mostramos de manera gráfica las 4 componentes principales resultantes.

2.2. Apartado ii)

Restringimos todos los sistemas a la región especificada.

En primer lugar, tomamos los datos del día a_0 del subsistema de altura geopotencial de 2022 y calculamos la distancia euclidia, a través de `dist_euclidea(dia0,dia)`, con los datos de todos los días del subsistema de altura geopotencial de 2021. Las distancias resultantes se almacenan en un vector ordenado de menor a mayor del que se extraen las 4 primeras posiciones. Así, obtenemos **los 4 días más análogos a a_0**.

En segundo lugar, tomamos los datos de esos 4 días del subsistema de temperatura de 2021 y hacemos la media —de esta forma, obtenemos una **predicción de temperaturas** para el día a_0—.

En tercer lugar, Para obtener **el error absoluto medio**, calculamos la media del valor absoluto de la diferencia entre la predicción y el día a_0.

Por último, mostramos gráficamente la información de la altura geopotencial seguida de la predicción de temperaturas y de la situación finalmente observada el día a_0 para comprobar visualmente la eficacia del cálculo.

3. RESULTADOS

3.1. Apartado i)

La varianza explicada de las 4 primeras componentes para X es `[0.4724878, 0.06072688, 0.03592642, 0.02815213]`. La suma explica un $60\,\%$ de varianza.

La varianza explicada de las 4 primeras componentes para $Y = X^T$ es `[0.8877314, 0.05177603, 0.00543984, 0.00357636]`. La suma explica más del $94\,\%$ de varianza y, por tanto, es más útil que X.

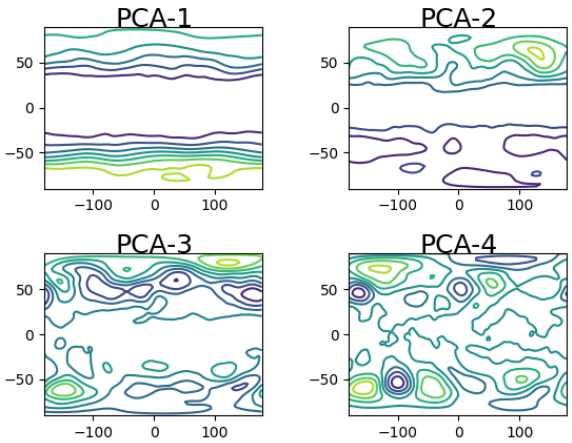

Fig. 1. Las 4 componentes principales.

3.2. Apartado ii)

Los 4 días de 2021 más análogos al 2022-01-11 localmente son 2021-03-23, 2021-01-16, 2021-01-12 y 2021-03-16.

El error absoluto medio local de la temperatura prevista para el 2022-01-11 es 1,4193446568080357.

Además, en la Figura 2 mostramos las gráficas arrojadas por la predicción y observación del día a_0.

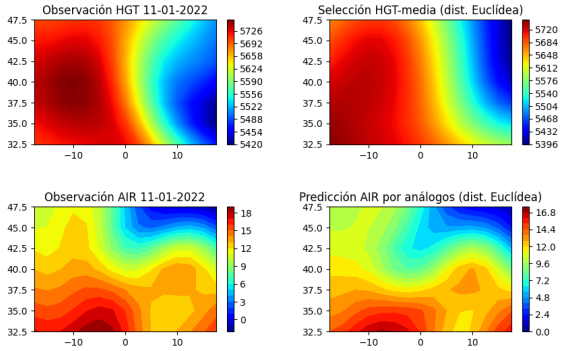

Fig. 2. Predicción y observación para el 2022-01-11.

4. CONCLUSIÓN

En el primer apartado, observamos que una sola componente principal ya explicaría una varianza del $88\,\%$ y las siguientes tres solo conseguirían mejorar este resultado en un $6\,\%$ aproximadamente. Podríamos suprimir las tercera y cuarta componentes principales puesto que apenas mejorarían el resultado en un $1\,\%$.

En el segundo apartado, los 4 días más análogos al 2022-01-11 en la zona geográfica restringida a las longitudes $(-20°, 20°)$ y latitudes $(30°, 50°)$ son días de invierno correspondientes a los meses de enero y marzo. Uno de ellos, el 2021-01-12, apenas se diferencia en un día. Esta es una observación útil para verificar la corrección del resultado.

Con las gráficas podemos ver la clara similitud entre las temperaturas predichas (gráfica inferior derecha) y la observación que posteriormente se realizó el 2022-01-11 (gráfica inferior izquierda). Ambas relacionadas con la muestra de la altura geopotencial (gráficas superiores).

5. ANEXO CON EL SCRIPT Y CÓDIGO UTILIZADO

5.1. Código

```
"""
PRÁCTICA 4: PCA Y ANALOGÍA
Belén Sánchez Centeno
Martín Fernández de Diego
"""

"""
Referencias:

    Fuente primaria del reanálisis
    https://psl.noaa.gov/data/gridded/data.ncep.reanalysis2.pressure.html

    Altura geopotencial en niveles de presión
    https://psl.noaa.gov/cgi-bin/db_search/DBListFiles.pl?did=59&tid=97457&vid=1498

    Temperatura en niveles de presión:
    https://psl.noaa.gov/cgi-bin/db_search/DBListFiles.pl?did=59&tid=97457&vid=4237

    Temperatura en niveles de superficie:
    https://psl.noaa.gov/cgi-bin/db_search/DBListFiles.pl?did=59&tid=97457&vid=1497
"""

import datetime as dt  # Python standard library datetime  module
import numpy as np
import math
import matplotlib.pyplot as plt
from netCDF4 import Dataset
#from scipy.io import netcdf as nc
from sklearn.decomposition import PCA
from copy import copy

# FORMATO
class Formato:
    BOLD = "\033[1m"
    RESET = "\033[0m"

#workpath = "/Users/martin/Documents/Estudios/Matematicas e Ingenieria Informatica
    /2021-2022/GCom/Git/Computational-Geometry/Laboratorio/P4"

f = Dataset("air.2021.nc", "r", format="NETCDF4")
time = f.variables['time'][:].copy()
time_bnds = f.variables['time_bnds'][:].copy()
time_units = f.variables['time'].units
level = f.variables['level'][:].copy()
lats = f.variables['lat'][:].copy()
lons = f.variables['lon'][:].copy()
air21 = f.variables['air'][:].copy()
air_units = f.variables['air'].units
#air_scale = f.variables['air'].scale_factor
#air_offset = f.variables['air'].add_offset
f.close()

f = Dataset("air.2022.nc", "r", format="NETCDF4")
time = f.variables['time'][:].copy()
time_bnds = f.variables['time_bnds'][:].copy()
time_units = f.variables['time'].units
air22 = f.variables['air'][:].copy()
f.close()

f = Dataset("hgt.2021.nc", "r", format="NETCDF4")
time21 = f.variables['time'][:].copy()
time_bnds = f.variables['time_bnds'][:].copy()
time_units = f.variables['time'].units
hgt21 = f.variables['hgt'][:].copy()
hgt_units = f.variables['hgt'].units
#hgt_scale = f.variables['hgt'].scale_factor
```

```
#hgt_offset = f.variables['hgt'].add_offset
f.close()

#f = nc.netcdf_file(workpath + "/" + files[0], 'r')
f = Dataset("hgt.2022.nc", "r", format="NETCDF4")
time22 = f.variables['time'][:].copy()
time_bnds = f.variables['time_bnds'][:].copy()
time_units = f.variables['time'].units
hgt22 = f.variables['hgt'][:].copy()
f.close()



"""
Dada una matriz (latitud,longitud) con valores en el dominio de longitud [0,2pi]
devuelve la lista con valores en el dominio de longitud [-pi,pi]
"""
def lons_normal_ref(dataset):
    return_dataset = copy(dataset) # Necesario para que no copie por referencia
    for i in range(72):
        return_dataset[:,:,:,i] = dataset[:,:,:,i+72]
    for i in range(72):
        return_dataset[:,:,:,i+72] = dataset[:,:,:,i]
    return return_dataset

# Normalizamos los mapas para que España esté en medio
hgt21a = lons_normal_ref(hgt21)
hgt22a = lons_normal_ref(hgt22)
air21a = lons_normal_ref(air21)
air22a = lons_normal_ref(air22)



# APARTADO i)
print("\n" + Formato.BOLD + "Apartado i)" + Formato.RESET)

hgt21b = hgt21a[:,level==500.,:,:].reshape(len(time21),len(lats)*len(lons))

n_components = 4

X = hgt21b
Y = hgt21b.transpose()
pca = PCA(n_components=n_components)

# Interpretar el siguiente resultado
pca.fit(X)
print(pca.explained_variance_ratio_)
out = pca.singular_values_
"""
Salida: [0.4724878  0.06072688 0.03592642 0.02815213]
La primera componente principal explica el 47% de la varianza del dataset.
En total, las cuatro primeras componentes principales explican entorno al 60% del
    dataset al entrenar X.
"""

# Interpretar el siguiente resultado
pca.fit(Y)
print(pca.explained_variance_ratio_)
out = pca.singular_values_
"""
Salida: [0.8877314  0.05177603 0.00543984 0.00357636]
La primera componente principal explica más del 88% de la varianza del dataset.
En total, las cuatro primeras componentes principales explican más del 94% del dataset
    al entrenar Y = tr(X).
"""

"""
Dado que es capaz de explicar más varianza con menos componentes,
se entrena el análisis de componentes principales PCA con Y = tr(X)
"""
```

```
Element_pca0 = pca.fit_transform(Y)
Element_pca0 = Element_pca0.transpose(1,0).reshape(n_components,len(lats),len(lons))

# Ejercicio de la práctica - Opción 1
fig = plt.figure()
fig.subplots_adjust(hspace=0.4, wspace=0.4)
for i in range(1, 5):
    ax = fig.add_subplot(2, 2, i)
    ax.text(0.5, 90, 'PCA-'+str(i), fontsize=18, ha='center')
    plt.contour(lons-180, lats, Element_pca0[i-1,:,:])
plt.show()



# APARTADO ii)
print("\n" + Formato.BOLD + "Apartado ii)" + Formato.RESET)

def dist_euclidea(dia0, dia):
    dist = 0
    for lat in range(len(dia0[0])):
        for lon in range(len(dia0[0][0])):
            dia0_500 = 0.5*dia0[level==500.,lat,lon]
            dia0_1000 = 0.5*dia0[level==1000.,lat,lon]
            dia_500 = 0.5*dia[level==500.,lat,lon]
            dia_1000 = 0.5*dia[level==1000.,lat,lon]
            dist += (dia0_500 - dia_500)**2 + (dia0_1000 - dia_1000)**2
    dist = math.sqrt(dist)
    return dist

"""
Sistemas base
"""
# Restringimos el espacio de búsqueda a las longitudes (-20,20) y latitudes (30,50)
hgt21c = hgt21a[:, :, :, np.logical_and(160 < lons, lons < 200)]
hgt21c = hgt21c[:, :, np.logical_and(30 < lats, lats < 50),:]

hgt22c = hgt22a[:, :, :, np.logical_and(160 < lons, lons < 200)]
hgt22c = hgt22c[:, :, np.logical_and(30 < lats, lats < 50),:]

air21c = air21a[:, :, :, np.logical_and(160 < lons, lons < 200)]
air21c = air21c[:, :, np.logical_and(30 < lats, lats < 50),:]

air22c = air22a[:, :, :, np.logical_and(160 < lons, lons < 200)]
air22c = air22c[:, :, np.logical_and(30 < lats, lats < 50),:]

lons = lons[np.logical_and(0 < lons, lons < 40)]-20
lats = lats[np.logical_and(30 < lats, lats < 50)]

# Almacenamos los días de 2021 y 2022 en las siguientes estructuras
dt_time21 = [dt.date(1800, 1, 1) + dt.timedelta(hours=t) for t in time21]
dt_time22 = [dt.date(1800, 1, 1) + dt.timedelta(hours=t) for t in time22]

"""
Obtención de los días más análogos
"""
# Tomamos el índice correspondiente al día 11 de enero de 2022
dia0 = dt.date(2022, 1, 11)
idx0 = dt_time22.index(dia0)
# Obtenemos los datos del día 11 de enero de 2022
a0 = hgt22c[idx0, :, :, :]

# Calculamos la distancia euclidia entre el día 11 de enero de 2022 y los días de 2021
distancia_idx = [[dist_euclidea(a0,hgt21c[i, :, :, :]), i] for i in range(hgt21c.shape
    [0])]

# Mostramos los 4 días más análogos considerando solo Z
num_dias = 4
distancia_idx.sort()
idx_analogos = [idx for _, idx in distancia_idx[0:num_dias]]

```

```
# Buscamos el día correspondiente al índice almacenados en el vector de pares distancia-índice
dias_analogos = [dt_time21[idx_analogos[i]].isoformat() for i in range(num_dias)]
print("Los", num_dias, "días de 2021 localmente más análogos al", dia0, "son:")
print(dias_analogos)

"""
Error absoluto medio de T según la media de los análogos
"""
# Media de los análogos
media_analogos = np.mean(air21c[idx_analogos,level==1000.,:,:],axis=0)
# Error absoluto medio
error_abs_medio = np.mean(abs(media_analogos - air22c[idx0,level==1000.,:,:]))
print("El error absoluto medio local de la temperatura prevista para el", dia0, "es:")
print(error_abs_medio)

"""
Opcional: Comprobación gráfica de los resultados
"""
fig = plt.figure(figsize=(10,6))
fig.subplots_adjust(hspace=0.5, wspace=0.3)

ax = fig.add_subplot(2, 2, 1)
ax.set_title('Observación HGT 11-01-2022')
p = plt.contourf(lons, lats, hgt22c[idx0,5,:,:], 200, cmap='jet')
fig.colorbar(p)

ax = fig.add_subplot(2, 2, 2)
ax.set_title('Selección HGT-media (dist. Euclídea)')
p = plt.contourf(lons, lats, np.mean(hgt21c[idx_analogos,5,:,:],axis=0), 200, cmap='jet')
fig.colorbar(p)

ax = fig.add_subplot(2, 2, 3)
ax.set_title('Observación AIR 11-01-2022')
p = plt.contourf(lons, lats, air22c[idx0,0,:,:]-273, 20, cmap='jet')
fig.colorbar(p)

ax = fig.add_subplot(2, 2, 4)
ax.set_title('Predicción AIR por análogos (dist. Euclídea)')
p = plt.contourf(lons, lats, media_analogos-273, 20, cmap='jet')
fig.colorbar(p)

plt.show()
```