TD 4 : Analyse Raster

Partie 4 : Croisement Vecteur-Raster (Zonal Statistics)

M2 GER
Université de Nantes -- IGARUN

Objectif Final

Nous avons deux sources d'information géographiquement distinctes :

RASTER (MNT)

Information continue (Altitude, Pente, Température) sur toute la surface.

VECTEUR (QUARTIERS)

Unités administratives discrètes (Polygones).

1. Les Statistiques Zonales

Le Concept

Le principe est de superposer les polygones au raster.
Pour chaque polygone :

1. Identifier tous les pixels qui tombent "dedans".
2. Récupérer leurs valeurs.
3. Calculer une statistique (Moyenne, Max, Médiane).

Pixels $\cap$ Polygone

Outil : rasterstats

Bien qu'on puisse le faire manuellement avec `rasterio`, la librairie **`rasterstats`** automatise tout ce processus.

> **INSTALLATION (SI NÉCESSAIRE)**
>
> Si vous ne l'avez pas, installez-la via Anaconda Prompt :
> `pip install rasterstats`
> *(C'est une librairie pure Python, pip est sûr ici).*

```python
from rasterstats import zonal_stats
```

2. Mise en Pratique

Prérequis : Projections

CRITICAL

RÈGLE D'OR

Le Raster et le Vecteur DOIVENT avoir le même CRS (Système de Coordonnées).

Si votre MNT est en WGS84 et vos quartiers en Lambert-93, le calcul sera faux ou vide.

```
# Vérification avant de commencer
print("CRS Raster :", src.crs)
print("CRS Vecteur :", quartiers.crs)

# Si différent, reprojeter le Vecteur vers le CRS du Raster
quartiers_proj = quartiers.to_crs(src.crs)
```

Calcul des Statistiques

La fonction `zonal_stats` prend les géométries et le chemin du fichier raster.

```
# Calculer la moyenne et le max pour chaque polygone
stats = zonal_stats(
    quartiers_proj,                 # GeoDataFrame (Polygones)
    "TD4_Data/MNT_Nantes.tif",      # Chemin du Raster
    stats=["mean", "max", "min"]    # Statistiques voulues
)

# Le résultat est une liste de dictionnaires
print(stats[0])
# Sortie : {'min': 2.0, 'max': 55.0, 'mean': 24.5}
```

Intégration dans le GeoDataFrame

Pour cartographier ces résultats, il faut les réinjecter dans le tableau des quartiers.

```
# 1. Convertir la liste de dicts en DataFrame
df_stats = pd.DataFrame(stats)

# 2. Concaténer (coller côte à côte) avec les quartiers
# Attention : l'ordre des lignes doit être préservé !
quartiers_enrichis = pd.concat(
    [quartiers_proj, df_stats],
    axis=1
)

quartiers_enrichis.head()
```

Visualisation du Résultat


```
fig, ax = plt.subplots(figsize=(10, 10))

quartiers_enrichis.plot(
    column="mean",
    cmap="magma",
    legend=True,
    legend_kwds={'label': "Altitude Moyenne (m)"},
    edgecolor="white",
    ax=ax
)

plt.title("Topographie des Quartiers de Nantes")
```

3. Mini-Projet Final TD4

Exercice de Synthèse

SCÉNARIO : ÎLOTS DE FRAÎCHEUR

Vous disposez : 1. Des **Quartiers** (Vecteur). 2. Du **NDVI** calculé en Partie 3 (Raster).

Objectif : Identifiez le quartier le plus "vert" de Nantes en calculant le **NDVI moyen** par quartier.

Méthodologie

1. Chargez le fichier `NDVI_Nantes.tif` (créé en Partie 3).
2. Vérifiez que les quartiers sont dans le même CRS que ce fichier.
3. Lancez `zonal_stats` avec la stat `mean`.
4. Incorporez le résultat dans le GeoDataFrame.
5. Triez pour trouver le top 3 (`sort_values`).

Code de Solution


```
# Calcul
stats_ndvi = zonal_stats(
    quartiers_proj,
    "NDVI_Nantes.tif",
    stats="mean"
)

# Intégration
quartiers_proj["NDVI_Moyen"] = [x['mean'] for x in stats_ndvi]

# Résultat
top_vert = quartiers_proj[["nom", "NDVI_Moyen"]].sort_values(
    by="NDVI_Moyen", ascending=False
)
print(top_vert.head(3))
```

Conclusion du TD 4

Vous savez désormais manipuler l'information géographique sous ses deux formes principales :

VECTEUR (GEOPANDAS)

Objets, Attributs, Relations topologiques.

RASTER (RASTERIO)

Pixels, Analyse physique, Statistiques de surface.

Fin du Module Raster

Merci de votre attention.