EPFL

La programmation en science de l'information géographique

LGB/GEOME

Gabriel Kathari

Sommaire

1. Introduction
2. Les différents langages de programmation pour les SIG
3. Quelques exemples

Introduction

- Qu'est-ce que la Programmation ?

La **programmation** est le processus de conception, d'écriture, de test et de maintenance d'un **ensemble d'instructions** destiné à être exécuté par un ordinateur.

- **Algorithmes** : Suites d'instructions logiques pour résoudre un problème donné.
- **Structures de contrôle** : Boucles, conditions et branchements permettant d'organiser l'exécution du code.
- **Structures de données** : Manière dont les informations sont organisées et manipulées en mémoire.
- **Interaction avec le matériel (hardware)** : Utilisation des ressources informatiques (processeur, mémoire, stockage).

Introduction

- Un **langage de programmation** est un ensemble de règles et de syntaxes permettant d'écrire des programmes compréhensibles par un ordinateur. Il sert de pont entre l'humain et la machine, en traduisant des instructions compréhensibles en code exécutable.

Langages de bas niveau (ex. Assembleur, C) : proches du langage machine, plus complexes mais très performants.

Langages de haut niveau (ex. Python, Java, JavaScript, R) : plus abstraits, faciles à utiliser et plus lisibles pour les humains.

- **Principaux paradigmes** :

Impératif (C, Python) : instructions exécutées séquentiellement.

Orienté objet (Java, C++) : organisation du code autour d'objets et de classes.

Déclaratif (SQL, Prolog) : on décrit le résultat souhaité plutôt que les étapes pour y parvenir.

PYPL – Popularity of programming language

Worldwide, Mar 2025 :

Rank	Change	Language	Share	1-year trend
1		Python	30.27 %	+1.8 %
2		Java	14.89 %	-0.9 %
3		JavaScript	7.78 %	-0.9 %
4	↑	C/C++	7.12 %	+0.6 %
5	↓	C#	6.11 %	-0.6 %
6		R	4.54 %	-0.1 %
7		PHP	3.74 %	-0.7 %
8	↑↑	Rust	3.14 %	+0.6 %
9	↓	TypeScript	2.78 %	-0.1 %
10	↑	Objective-C	2.74 %	+0.3 %
11	↓↓	Swift	2.44 %	-0.3 %
12		Go	2.06 %	-0.2 %
13		Kotlin	1.9 %	+0.0 %
14		Matlab	1.68 %	+0.1 %
15	↑	Ada	1.33 %	+0.3 %

https://pypl.github.io/PYPL.html

Les différents langages de programmation pour la science de l'information géographique

Les principaux langages de programmation pour les SIG

- Python : Langage principal de développement SIG
- R : Analyse statistique et traitement spatial
- SQL : Manipulation de données spatiales dans PostGIS, Requêtes spatiales pour l'analyse des données
- Javascript : Cartographie web interactive, extraction de données
- C++ / Java : Développement avancé d'algorithmes pour SIG (exemple → GeoDa)

Python

- Large écosystème de bibliothèques SIG (geopandas, folium, GDAL, shapely, pyQGIS)
- Facile à apprendre et à utiliser, avec une syntaxe intuitive.
- Excellente compatibilité avec les logiciels SIG comme QGIS, ArcGIS et PostGIS
- Adapté à l'automatisation des processus et au traitement des grandes données spatiales.
- Intégration avec l'Intelligence Artificielle et le Machine Learning pour l'analyse prédictive et la classification d'images satellites.

Javascript

- Langage incontournable pour le développement d'applications SIG web interactives.
- Utilisation avec des bibliothèques comme Leaflet.js
- Permet la création de cartes dynamiques et interactives.
- Excellente compatibilité avec les API de cartographie (Google Maps, ArcGIS API for JavaScript).

C++

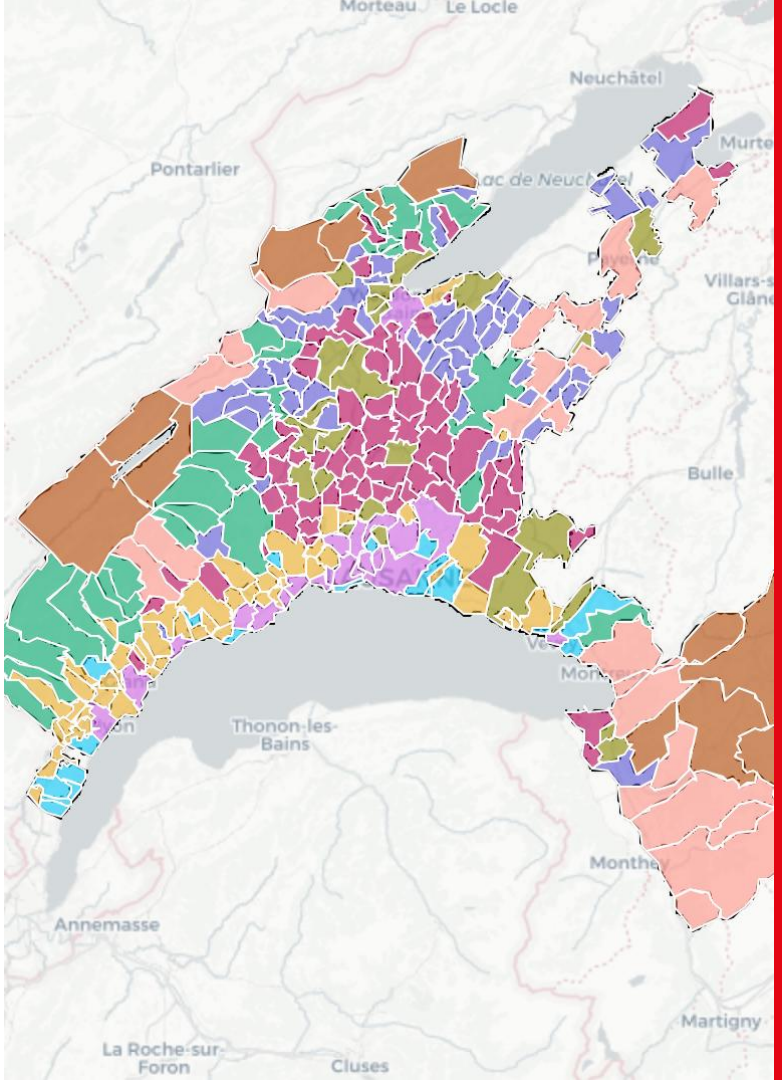
- Utilisé pour développer les moteurs SIG comme GeoDa
- Très performant pour le traitement et l'analyse de grandes quantités de données spatiales.
- Indispensable pour l'optimisation des algorithmes SIG et les traitements lourds.
- Excellente gestion de la mémoire et des calculs complexes.

SQL

- Manipulation efficace des bases de données géospatiales avec PostGIS et Spatialite.
- Optimisé pour les requêtes spatiales complexes et l'analyse à grande échelle.
- Idéal pour le stockage et la gestion de données SIG massives.
- Peut être combiné avec Python et R pour des analyses avancées.

R

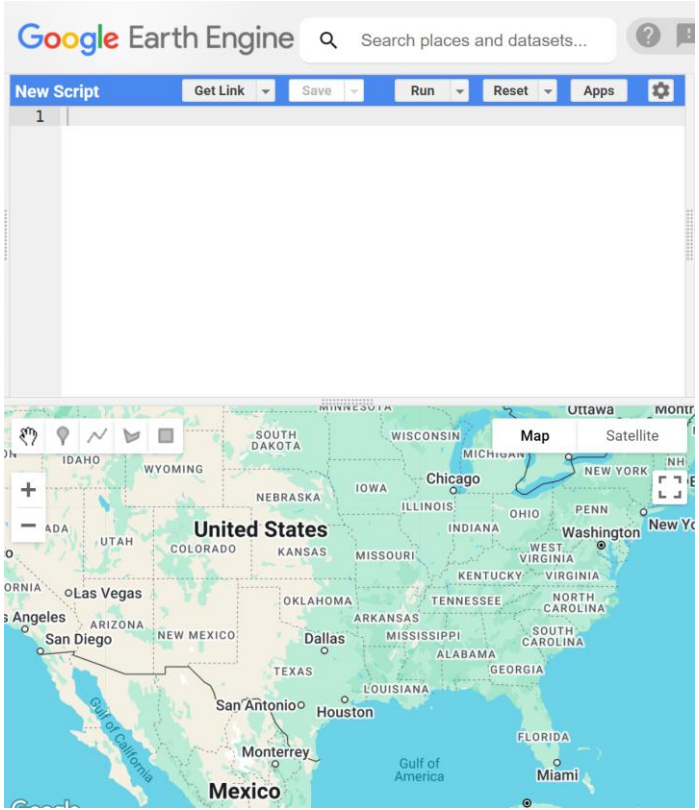
- Langage de référence pour l'analyse statistique
- Packages spécialisés pour la statistique spatiale (sf, sp, raster, gg2plot)
- Idéal pour les études environnementales, épidémiologiques et l'analyse des tendances spatiales.


Morteau
Le Locle
Neuchâtel
Pontarlier
Bulle
Thonon-les-Bains
Annemasse
Martigny
La Roche-sur-Foron
Cluses

Exemples

Exemple : Utiliser la console google earth engine

- Permet de faire une extraction en masse de données satellites.
- Permet de spécifier la zone, la date ainsi que les propriétés d'une extraction et d'utiliser les modèles d'interpolation de données de google → Afin d'éviter toute couverture nuageuse par exemple.
- Fonctionne en ***javascript*** directement dans la console ou depuis un code ***Python*** dans un environnement prédéfini.
- https://code.earthengine.google.com/



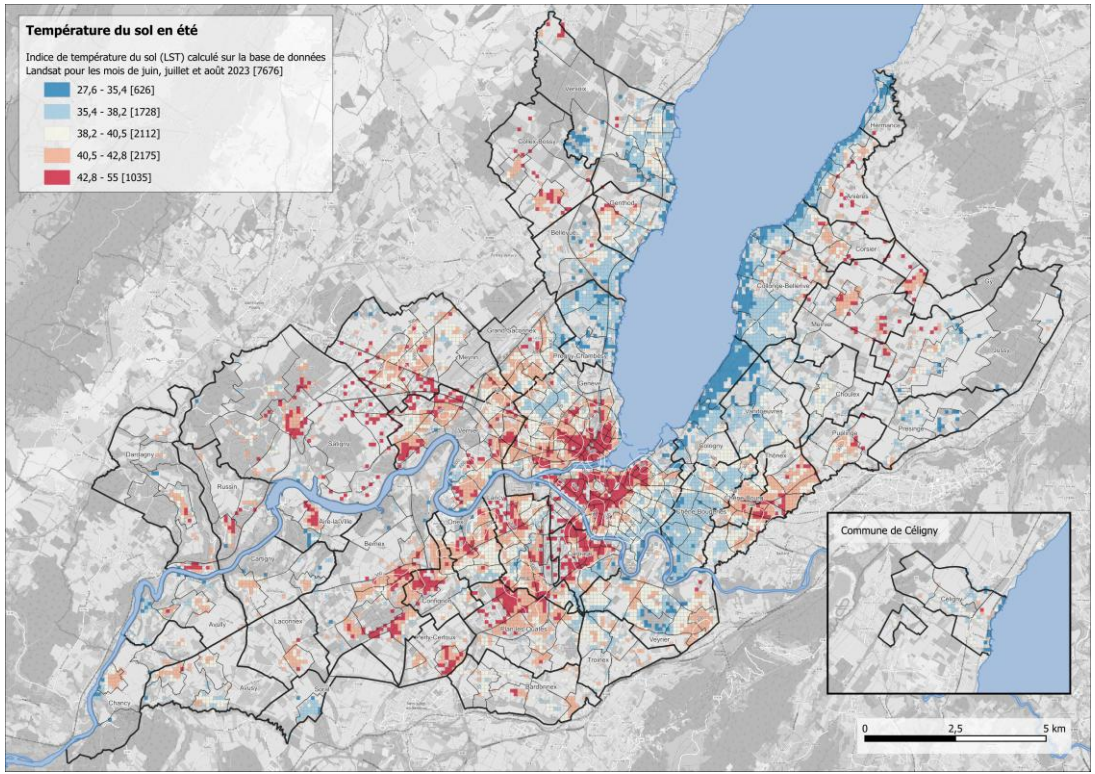
Exemple : Utiliser la console google earth engine

- Les satellites d'observation de la Terre fournissent des données essentielles pour la surveillance de notre planète.
- Les principaux satellites sont:
  - **Landsat**: Surveillance environnementale à long terme.
  - **Sentinel**: Données haute résolution pour les études terrestres, océaniques et atmosphériques.
  - **MODIS**: Surveillance à grande échelle du climat et de l'écosystème.

- Plateformes pour accéder à ces données :
  - Google earth engine
  - USGS
  - Sentinel-hub

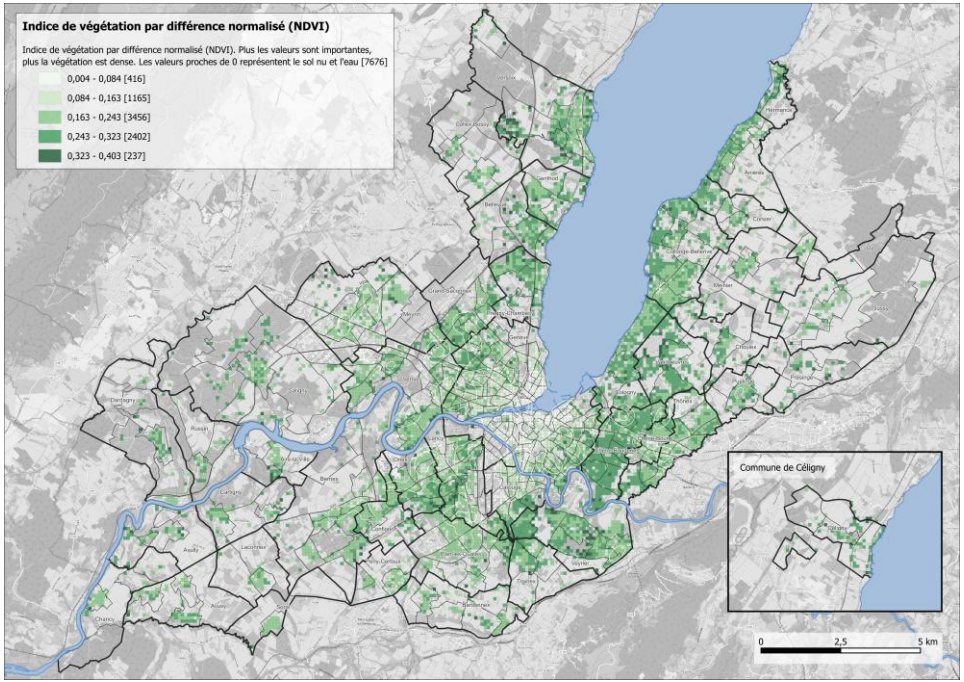
Exemple : Utiliser la console google earth engine

- LST : Indice de température au sol



Exemple : Utiliser la console google earth engine

- NDVI : Indice de végétalisation du sol



NDVI.js

Exemple : Utiliser la console google earth engine

- Extraction du LST pour plusieurs régions avec un code ***javascript***

```javascript
var states_list = ['Alabama','Texas']

var satellite = 'L8';
var date_start = '2022-05-01';
var date_end = '2022-08-31';
var use_ndvi = true;

var LandsatLST = require('users/sofiaermida/landsat_smw_lst:modules/Landsat_LST.js')

calculate_LST(states_list);

function calculate_LST(states_list){
  states_list.forEach(function(state) {
    // trouver les geometry correspondants aux etats selectionnes
    var aoi = ee.FeatureCollection("FAO/GAUL_SIMPLIFIED_500m/2015/level1").filter(ee.Filter.eq('ADM1_NAME', state)).geometry();
    // récupère les données landsat
    var LandsatColl = LandsatLST.collection(satellite, date_start, date_end, aoi, use_ndvi);
    var compositeImage = LandsatColl.median();
    var clippedImage = compositeImage.clip(aoi);

    // récupère le LST pour les états selectionnees et les dates sélectionnes
    Export.image.toDrive({
      image: clippedImage.select('LST'),
      description: 'LST_Arusha_May_2022'+'_'+state,
      scale: 30,
      region: aoi,
      fileFormat: 'GeoTIFF',
    });
  });
}
```

LST_multiple_states.js

To calculate the Land Surface Temperature (LST), we will rely on the calculations from "Google Earth Engine Open-Source Code for Land Surface Temperature Estimation from the Landsat Series" by Sofia L. Ermida (https://doi.org/10.3390/rs12091471).

Exemple : création d'application web interactives

- En ***javascript*** avec la librairie leaflet

→ Exemple de la carte de Lausanne

https://leafletjs.com/

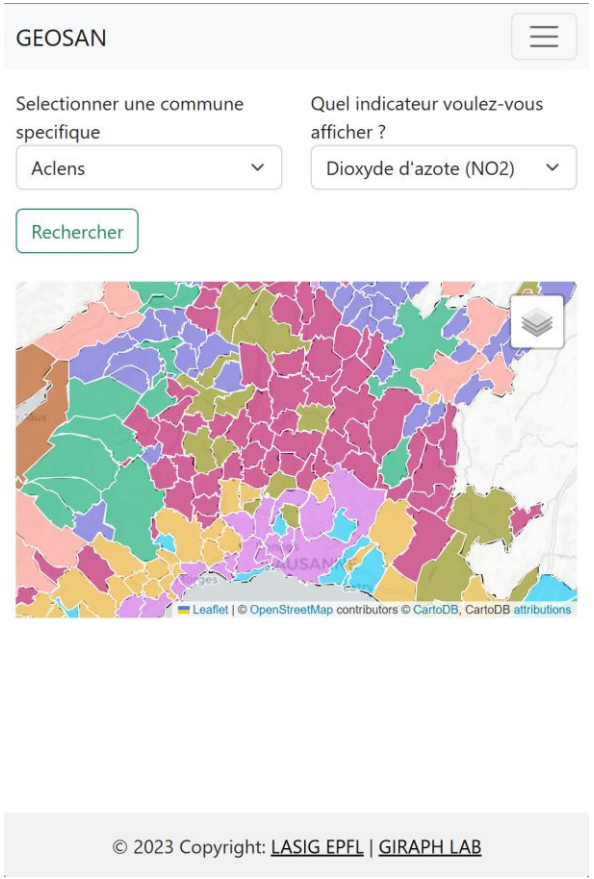
- En ***Python*** avec la librairie folium

→ Exemple de geosan

https://python-visualization.github.io/folium/latest/

- Ces cartes peuvent être intégrées dans un framework pour construire des web-app ou encore des ligiciels.

→ En Python on peut utiliser le freamwork Django par exemple.

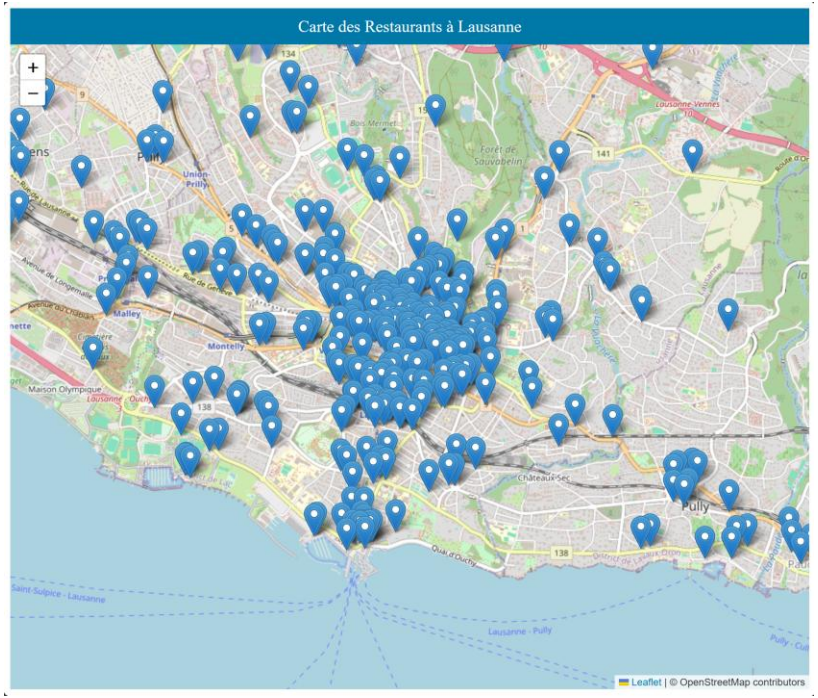


Exemple : création de cartes web interactives

- Création d'une carte web interactive avec la localisation des restautants de genève et des informations sur ces derniers.
- Leaflet.js
- Overpass-api (OSM)

https://wiki.openstreetmap.org/wiki/Overpass_API

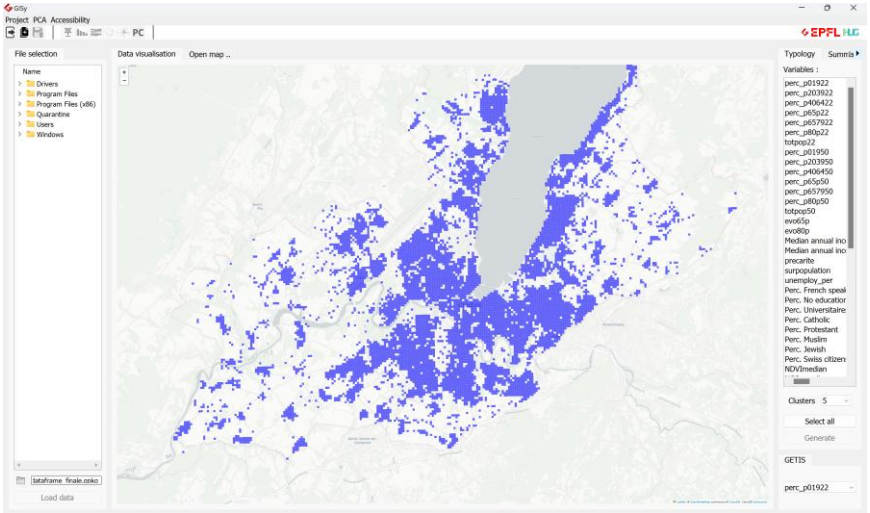
restaurants_Lausanne.js



Speaker

Exemple : création de software - GISy

- Permet de définir la typologie d'une population en fonction de différentes variables géoréférencées → calcul de PCA, clustering hierarchique, visualisation du résultat, aide à la définition.
- Permet de calculer les temps de trajet avec OpenStreetMap d'un point au centre d'interêt le plus proche
- Auteur : Reza Jabbir (travail de master) et Gabriel Kathari



Exemple : Affichage du chemin le plus court à vélo

- Calcul et affiche le chemin le plus court à vélo entre deux points.
- Utilisation de 2 librairies python :

→ osmnx

(https://osmnx.readthedocs.io/en/stable/)

→ networkxx

(https://networkx.org/)

trajet_velo.py

trajet_velo_folium.py

```python
import osmnx as ox
import networkx as nx
import matplotlib.pyplot as plt

def shortest_path_bike(start, end):
    # Télécharger le graphe routier de Genève
    G = ox.graph_from_place("Genève, Switzerland", network_type='bike')

    # Convertir les adresses en coordonnées GPS
    start_location = ox.geocode(start)
    end_location = ox.geocode(end)

    # Trouver les nœuds OSM les plus proches des coordonnées
    orig_node = ox.nearest_nodes(G, start_location[1], start_location[0])
    dest_node = ox.nearest_nodes(G, end_location[1], end_location[0])

    # Calculer le chemin le plus court en distance
    shortest_route = nx.shortest_path(G, orig_node, dest_node, weight='length')

    # Afficher le chemin sur une carte
    fig, ax = ox.plot_graph_route(G, shortest_route, route_linewidth=3, node_size=0, bgcolor='white')
    plt.show()

# Points de départ et d'arrivée
start_point = "Servette, Genève, Switzerland"
end_point = "Eaux-Vives, Genève, Switzerland"

shortest_path_bike(start_point, end_point)
```

MERCI POUR VOTRE ATTENTION