**Geospatial Data Developer Take Home Challenge answers**

By Miguel González Jiménez

# **Part 1. Setting up the infrastructure with Docker 🐳 (3 points)**

While we wrangle data at different levels and stages of maturity, PostgreSQL and its extensions ecosystem has demonstrated to be great at solving quite a lot of the challenges we face in our day to day, specially when it comes to answering questions that lay on different geospatial data sources and types, including raster, vector and spatial index (H3) data mainly.

The first part of the challenge is to set up a Docker image that fulfills the following requirements:

* Runs PostgreSQL, with minimum major version 14
* Has PostGIS extension installed, with minimum major version 3
* Has the h3-pg extension installed

To perform this exercise, I searched for a Docker image that would satisfy the exercise's indications. As long as the image matches your requirement, it always be much easier to find an existing one rather than building your own one.

Finally I found it in the Docker Hub, in the public repository of [IvanLonel](https://hub.docker.com/u/ivanlonel). It can be easily accesed through this [link](https://hub.docker.com/r/ivanlonel/postgis-with-extensions). To make sure that the Postgre and Postgis versions are the ones I want, I selected versions 16 and 3.4, respectively.

Once the image to be used was chosen, I proceeded to its **execution**. To do so, I have taken the following steps:

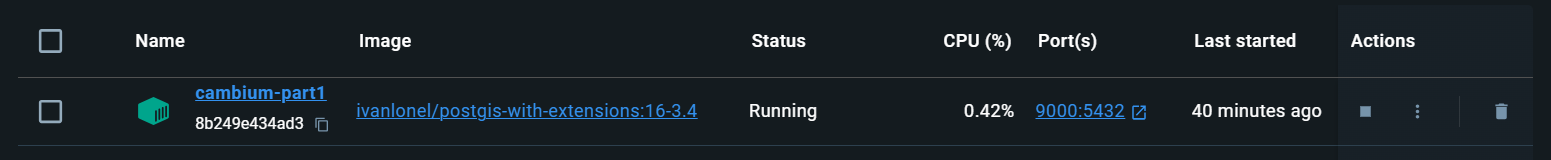
1. **Installing Docker Desktop** and registering as a new user.
2. Since the image is hosted within the Docker Hub, I was able to easily **pull** it to my local device by using the following command:

>>> docker pull ivanlonel/postgis-with-extensions:16-3.4

1. Next, I created the **container** 'cambium-part1' from the image, starting with the following command:

>>> docker run --name cambium-part1 -d -p 9000:5432 -e POSTGRES\_PASSWORD=mypswd ivanlonel/postgis-with-extensions:16-3.4

The container is running correctly, as shown in the figure below in DockerDesktop:



To make the container reachable from a local port, I had to **bind** it to the 9000 port, what is indicated by the -p flag. The -d flag indicates that I have run it in backaground, and the -e flag allows me to enter my Docker password via the POSTGRES\_PASSWORD argument.

It is worth mentioning that the order of the arguments matters, although to avoid errors it has been easier to use PowerShell command prompt than cmd.

If I had not found this image, I would have had to set it up myself. To do this, I would have had to program a Dockerfile and convert it to an image by executing the following command:

>>> docker build dockerfile .

I would have tried to do this FROM the official Docker image of postgis, which has Postgres versions higher than 14.

# **Part 2. Building a suitability analysis tool 🌎 (7 points)**

The challenge is to create a tool (this could be a web application, an API, a notebook or simply a script, you're free to choose) using Python as your main language (SQL is also your friend if you like) that answers the following requirements:

* Allows an input geospatial dataset in one of the following formats: GeoParquet, ESRI Shapefile, GeoJSON, KML. This dataset has a geometry column that represents the area of interest of our team, the area will always be a region within the argentinian province of Corrientes, generally ranging from 100-5.000 hectares. You can find an example area of interest in the data folder in GeoJSON format.

This has been achieved by programming the 'Read\_Dataset' **class**, programmed in Python and hosted in the 'cambium\_tools.py' module.

This class, through the geopandas spatial library, allows the **reading** of the four different types of datasets and **checks** that the dataset contains some polygonal object and is located within the province of Corrientes.

The class is initialized by a single positional argument, which is the path to the file, and is built using the 'read\_file' function, from geopandas. Other arguments can be passed to this function as \*\*kwargs. Reading the dataset generates a **geopandas dataframe**, which can be accessed via the ‘gdf’ attribute.

Since reading each different dataset format usually requires some particular feature, the class has **private functions** that customize this reading (except for the Shapefile type, which is not a problem). It is possible that the class may also be able to read other types of formats, such as zip files or url links.

Since I understand that the area of interest of your team has to correspond to a **geometric object** **of polygon** type, the class includes, through the internal function \_is\_polygon(), a check of the geometry type. In case of being composed of multiple different geometries, it only keeps the polygonal type (polygon or multi-polygon), and in case of not being a polygon, an error is raised. In case you want to visualize a layer that is not a polygon, you can pause this object type check with the input argument polygon=False.

The class also performs a **second check**: a spatial check of the input dataset to verify that it is within the Argentine province of Corrientes. In case it is not inside, it checks if the whole area falls outside or if there is any portion inside. A warning is triggered if the area is not within the general range (from 100 to 5,000 ha).

In addition, the class is well documented and has errors and/or warnings included when requirements are not met.

In the jupyter notebook 'Part2.1.ipynb' you can see an **example** of how this class works with several test datasets.

* Provides a visual output (map) that has areas divided into 3 categories representing the suitability of the area for our projects (**Low, Medium, High**) and any additional information that may help business users take data-driven decisions (total usable area, etc).

This exercise was carried out through a **GIS analysis** of the layers of interest, with the objective of obtaining a definitive polygonal layer showing the all the possible areas. Subsequently, the polygons of this layer have been divided into three categories according to their suitability and represented on a map.

Although the GIS analysis was initially started by scripting in Python with the rasterio library, for reasons of efficiency, QGIS and ArcMAP were finally used. The categorization of the areas and visualization of the final map was done in Python using the geopandas and folium libraries.

All the steps carried out to perform the GIS analysis and generate the final map are shown as follows:

1. **Downloading necessary datasets:**

The following layers have been used in this exercise:

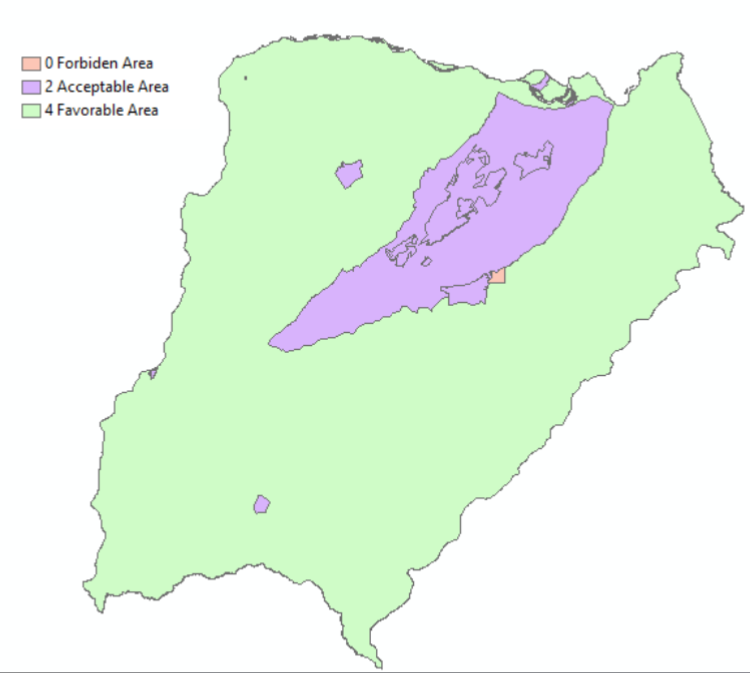
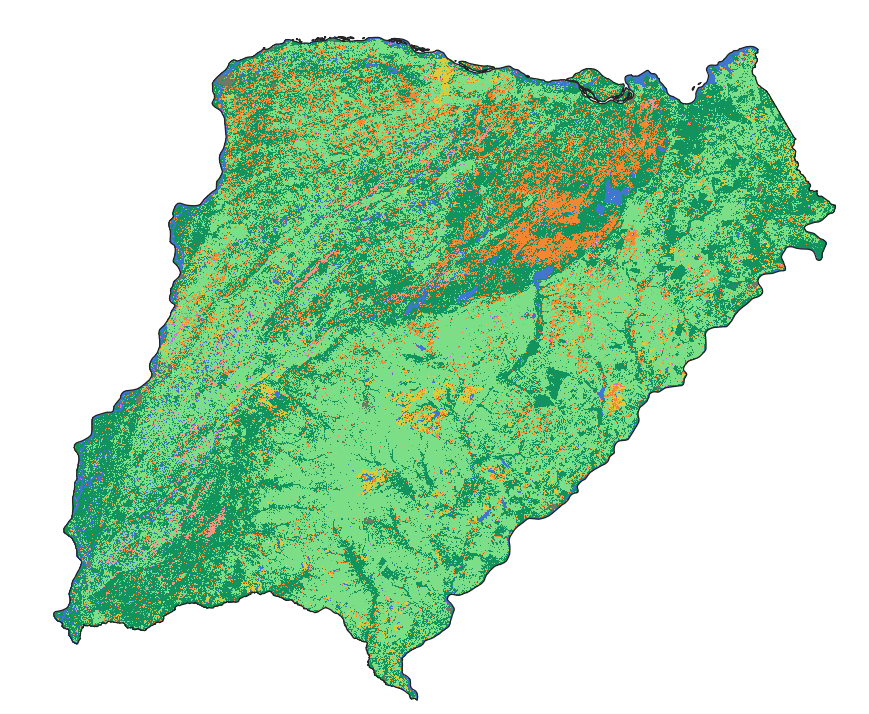
* **Dynamic World V1 (DW1)** 10 meter resolution of March 2020 to respect the existence of forest cover 5 years old. Mainly used to discard forest areas. For a more rigorous analysis, it would be convenient to cross this layer with a current one, in order to discard possible new forest areas.
* **Global 30m Height Above Nearest Drainage** (HAND). Used for consideration of well-drained soils.
* **ALOS World 3D-30m**. Used to discard areas with poor drainage (slope < 1%) and very steep slopes.
* **Protected Areas of Argentina:** Mainly used to discard wetlands. I consider that the date of creation of the layer is prior to March 2020. For the same reason as for the DW1 layer, it would be convenient to make sure that there are no new areas declared as wetlands.

The DynamicWorld-V1, ALOS and HAND layers have been downloaded from GEE by programming scripts (available in the following directory: data\GEE-scripts), while the protected areas layer has been obtained from the exercise link.

1. **Datasets managing and formatting:**

The geographic reference system of all the layers (raster and vector) is the geographic one (WSG 89, epsg: 4236), so **first of all**, in order to perform the analysis in a metric reference system, all the layers have been **re-projected** to the "Campo Inchauspe / Argentina 5" system (EPSG: 22195; one of the possible ones for the study area), and aligned with respect to the DW raster.

**Secondly**, the rasters have been **merged** (merge) forming a mosaic and **clipped** to the study area (province of Corrientes), as shown in the figure below.



**Third**, since the rasters have different spatial resolutions (ALOS: 30X30, HAND: 30X30, DW: 10x10), the lower resolution raster (ALOS and HAND) have been **resampled** to the resolution of the higher resolution raster (DW), and **aligned** with respect to the DW layer.

**Finally**, in order to include the layer of protected areas in the raster analysis, it has been **rasterized**, generating a raster aligned with the rest and with the same spatial resolution.

1. **Reclassification and weighting of variables**

The GIS analysis consisted in the **reclassification** of the raster layers and their **multiplication**. The reclassification consisted of re-evaluating the variables of the layers with numerical values between **0** (**forbidden** **zone**, not suitable at all) and **5** (**suitable** **zone**). The layers have been reclassified according to the suitability of each variable (layer) in the cultivation of the genus *Peltophorum dubium*, as shown below:

* 1. **Land Cover map (DW1):**

|  |  |
| --- | --- |
| **Dynamic Worl Classification** | **Assigned Weight** |
| 0 Water  1 Trees  2 Grass  3 Flooded vegetation  4 Crops  5 Shrub & Scrub  6 Built area  7 Bare ground  8 Snow & Ice | **0**  **0**  **5**  **1**  **0**  **2**  **0**  **3**  **0** |

Thus, the DynamicWorld layer, which indicates the typology of the terrain, has given a **value of 0** to the categories of Water, Trees, Crops, Built Area and Snow-Ice areas, because they are not favourable areas. I consider that the best areas for the location of the areas of interest are the 'Grass' type (receiving a value of **5 points**), then the 'Bare ground' and, finally, the areas with 'Shrubs', since they could be considered for forestation.

Although areas in the 'Flooded vegetation' category are usually not suitable because they are likely to be below the minimum HAND level (1m), they could be interesting if they are above this level. Hence, they have been assigned the **value of 1**.

* 1. **Slope map (from ALOS):**

From the ALOS DMS, a **slope map** has been calculated. In the reclassification, cells with a slope value of less than 1 have received a **value of 0** (**Forbidden Area).** Flat areas have been rated better and areas with steeper slopes (from a value of 70% slope) have been rated worse.

|  |  |
| --- | --- |
| Slope (%) | Value assigned |
| <1  1-70  70-100  100-120  > 120% (45º) | 0  5  3  2  1 |

* 1. **HAND map:**

In this case, the HAND layer values are simply ranked according to whether they are above or below the **unit value**. In the first case, they receive a score of 0 (forbidden) and in the second case a score of 5 (ideal zone).

* 1. **Protected Areas:**

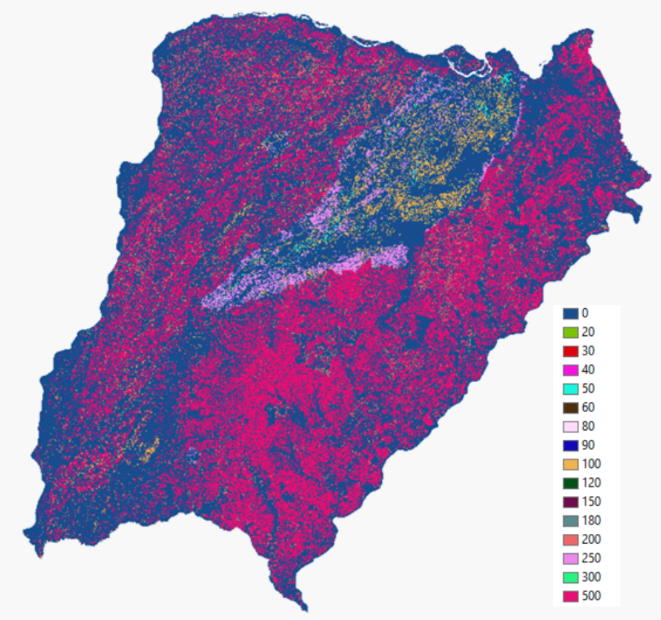
|  |  |
| --- | --- |
| **Type** | **Weighted Value** |
| Protected Wetland Area  Protected non-wetland area  Non-protected Area | 0  2  5 |

Instead of making a boolean classification (suitable, unsuitable) of the Protected Areas layer, a **categorical** classification into three classes has been preferred:

* + - The protected areas that are **wetlands** have been given a value of 0 because they are prohibited areas. The following have been included in these, because of their name: Lagunas y Esteros del Iberá, Humedales Chaco and Laguna Brava.
    - **Protected areas** that are **not wetlands** have been classified as acceptable areas (**value 2**), since there is a possibility that, even though they are protected areas, they could be suitable (legally and officially) for reforestation, which could be a beneficial practice for the maintenance of the area by slowing down processes such as erosion or desertification. The rest of the areas of the 'Protected Areas' layer included within the province of Corrientes have been included here.
    - **Non-protected** **areas** receive the best score (**5**), being considered Favorable areas.

The **reclassification** process was performed using the 'Reclass' tool of ArcMap. Subsequently, all layers have been **multiplied** to obtain a final raster ('Final\_layer.tif'), with pixels graded between 0 and 500 (figure below). The value of these pixels is in relation to the suitability to be declared as area of interest.

Multiplication and not the sum of the layers has been chosen as the layer combination tool. As zero is the product between any number and zero, whenever the pixel value of any layer is zero, the result always will be zero. Thus, even if an area may be suitable for almost all the variables (layers), it will remain as 'Forbidden Area' (value=0) as long as there is at least one variable with the value set to zero.



Finally, this layer has been vectorized for the generation of polygons.

1. **Map visualization:**

In this second phase of the exercise, the resulting vector layer containing all the polygons resulting from the analysis was imported into Python and categorized according to their value to generate a definitive map.

The steps taken in this phase of the exercise are collected in the jupyter notebook 'Part2.2.ipynb'. Since the vector layer containing all the polygons is very large (>1GB), the pyogrio library has been used to speed up its reading and operability.

**First**, the number of polygons in this layer has been **reduced** by eliminating polygons with an area **less than 100** ha and with a **value of 0** ('Forbidden Area'), creating a new lighter one called 'reduced\_final\_vector.shp'.

**Secondly**, the polygons have been **categorized** according to their value in the three areas indicated by the exercise statement:

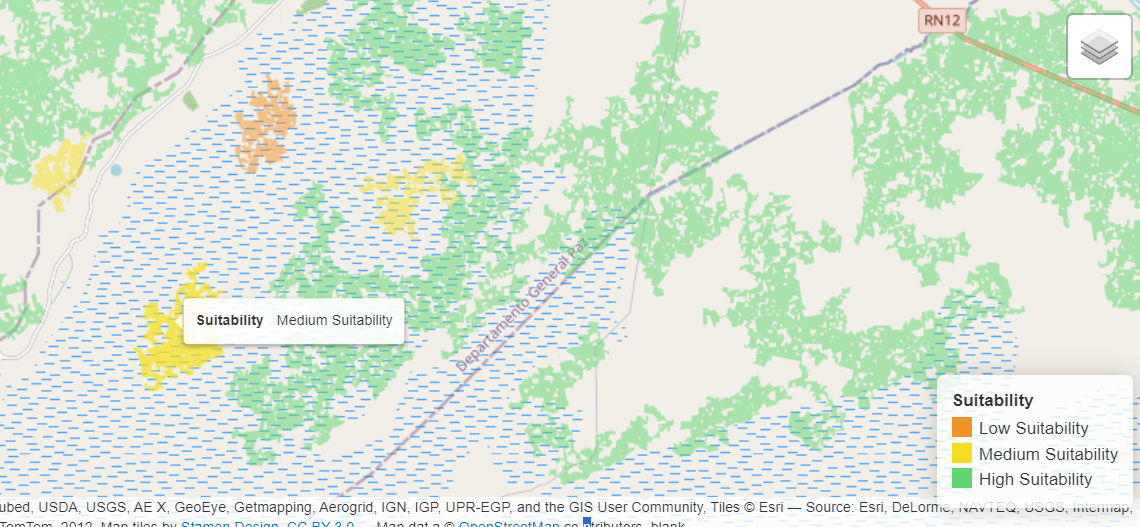
* **Low Suitability Area**: areas with values ranging among 1 and 150.
* **Medium Suitability Area**: with values ranging among 150 and 300.
* **High Suitability Area**: with values ranging among 300 and 500.

**Finally**, a **web map** has been designed using the folium library to visualize these areas. Since the study area used for the GIS analysis (province of Corrientes) is very large, the final map is very heavy (>500MB). It is located at the google Drive in [this link](https://drive.google.com/drive/folders/1SfHQhcZvCr05-QabxZvqyv_s_9TFQwE7), as an **html file** ('Map.html').

The complete visualization can be done by opening the file in a conventional browser, such as Google Chrome. In case it does not load, a small portion can be viewed in the 'Map\_small\_sample.html' file, or by running the last cell of the 'Part2.2.ipynb' script.

The map contains **other information** that may be of interest, such as ESRI-Image, ESRI-StreetMap and Stamen Terrain layers, which are satellite image, street map and digital surface model layers, respectively.

A small portion of this final map is shown below:



# **Additional questions**

* What kind of aspects would you take into account if you were assigned with the task of implementing this type of analysis at scale?

By scale I mean a study area of variable extension, from **large scales** (regional, national) to **small scales** (municipal, local). One of the important aspects that I would take into account depending on the scale would be the **reference system** to be used, in particular, its time zone when working on local/regional studies.

In the case of a very large scale, the **computational capacity** of my computer would come into play: check that my computer has enough resources to do this analysis (mainly, enough RAM and space). Otherwise, I would first try to optimize the resources of my computer and use appropriate tools for large datasets (e.g. pyogrio for Python spatial analysis) and, if I still do not succeed, I would consider an analysis methodology based on cloud services, such as Google Earth Engine, or studying the possibility of connecting to a remote server.

Another aspect that I would take into account in the analysis depending on the scale would be the **data source**. The data to be needed in regional/national studies are likely to be available in official/national databases. Whereas for local studies you may have to look for data in local councils or organizations.

Finally, I would also consider the **accuracy of the datasets**. Large scale analyses tend to be more generalist, while local analyses tend to require greater precision.

* How can you ensure that the output result is consistent over time and responds to the data quality business users expect?

First, before starting the GIS analysis, I would **check the dates** of the datasets I have available and see if I can get the datasets for the dates or date range I want.

In case of only finding deprecated datasets, I would consider the option of **creating them myself** if it would not be too complicated. For example, in the case of digital elevation models, I could easily generate them through LAStools (Qgis or ArcMap) if I had LiDAR data available.

As the layers never have the same dates, the map resulting from the analysis will always have been obtained from the data of a **period of time** (date\_range: date of the oldest layer - date of the most modern layer). I would then **make sure** that this time range is acceptable to perform the GIS analysis, or else I would have to change the methodology to achieve time consistency among layers.

Finally, I would always **inform the client** about the date range of the GIS analysis and the approximate validity date of the resulting map. For this, I would inform me about the usual rate of change/update of the layers used in the analysis.

* If one or more of the input data sources vary with time by their nature (e.g., land use change, deforestation, etc.), how would you approach this challenge? How can we compare the results of the analysis today vs a few months ago?

To be aware of possible changes in the data sources, first of all and for the case of maps that do not usually undergo frequent changes, I would try to be aware of the **communication channels** of the data providers. Generally, these providers usually notify through their official channels or **social networks** about the modifications and updates of their maps.

In case I did not find out about the modification of any data source or it was very changeable, I would try to **develop a script** that would allow me to know periodically if changes are occurring in the original data source. The script could access the **web** section in question and check if the same data exists. Maybe there is some accessible variable that reports the creation/modification date of the data offered.

I would also consider that if the analysis methodology is half- automatized, it might be easier and more efficient to **rerun** it with the new data sources than to generate a change comparison. In this way, the results of a GIS analysis would have a date (or time range) of validity. If the changes were periodic in the data sources we rely on, we could even establish a minimum frequency for updating our cartographic products.

**Secondly**, if changes have been made in the data sources used, I would devise some way to compare the results of the analysis today vs a few months ago. I would try to obtain some **algorithm** for change detection, either between raster or vector layers, or between other types of files. I would probably end up developing code for this algorithm, or glue coding for joining existing programs.

In the case of **comparisons between raster data,** I find them simpler. I could easily make a comparison between the pixels of each band and generate a map of changes as output. In the case of **shapefile data** I would find it more complicated. I would consider rasterizing the layer to apply the previous step, or always convert the geometry to points (with their attributes included in each of them) and make a geostatistical comparison between them, which would inform me both of changes in their location and in their typology (modified attributes). In this way, a new map of changes could be obtained that would allow me to identify the areas that have undergone the greatest changes.

In the case of **comparison of rasters**, this is something that some viewers already do, such as the IGN with the [PNOA photo comparator](https://visualizadores.ign.es/comparador_pnoa/), so I would try to study the way in which that tool has been developed. In case they are tools from public organizations, I would try to access the technical report related to the release of that tool. **Github** is always a good place to discover new tools or get inspired to create new methodologies.