

GIS, Python, PYQGIS



Parte del material obtenido de
[https://github.com/sebastianhohmann/gis_course/
tree/master/QGIS/research_course](https://github.com/sebastianhohmann/gis_course/tree/master/QGIS/research_course)

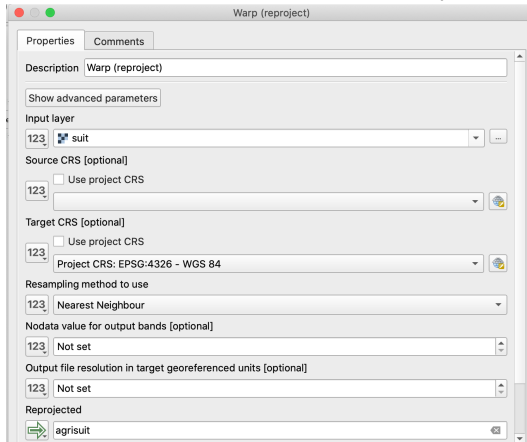
Herramientas Computacionales para Investigación

Modelador Gráfico

- Para que la investigación sea replicable (¡también para nosotros mismos!) queremos automatizar todo lo posible el flujo de trabajo del GIS.
- Buen puente hacia la automatización completa en Python.
- Ejemplo: construir un modelo para calcular la idoneidad agrícola media de todos los condados de EE.UU.
 - Descargar los condados de EE.UU. de <https://gadm.org/data.html>. Del ZIP que les descarga, usaremos **gadm41_USA_2.shp**
 - Descargar los datos rasterizados globales de idoneidad agrícola de <https://sage.nelson.wisc.edu/data-and-models/atlas-of-the-biosphere/mapping-the-biosphere/land-use/suitability-for-agriculture/>.

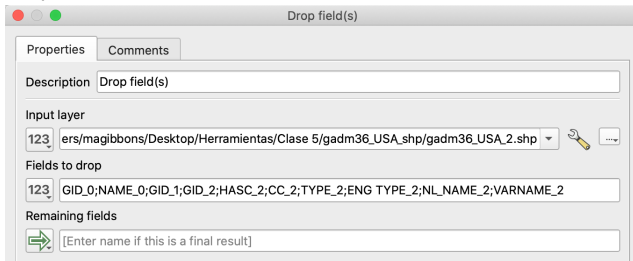
Modelador Gráfico

- Layer → Add Layer → Add Raster Layer: /suit/suit/hdr.adf
- Processing → Graphical Modeler ( →  → New Model)
- Algorithms → Raster Projections → WARP (reproject)



Modelador Gráfico

- Layer → Add Layer → Add Vector Layer:
gadm41_USA_2.shp
- Algorithms → Vector Table → Drop Fields
- Fields to drop:
GID_0;NAME_0;GID_1;GID_2;HASC_2;CC_2;TYPE_2;ENG
TYPE_2;NL_NAME_1;NL_NAME_2;VARNAME_2 (tiene que
ser introducido como una lista separada por punto y coma sin
espacios)

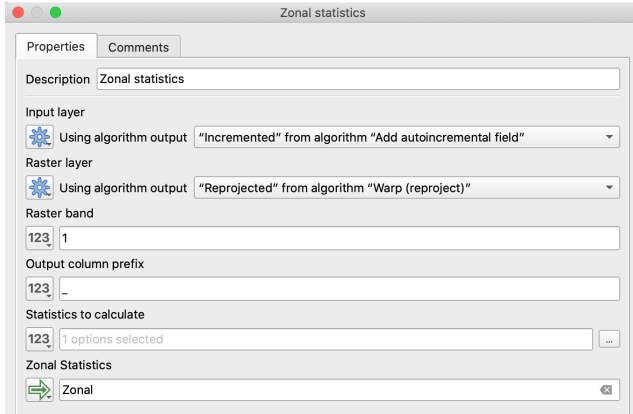


- Algorithms → Add autoincremental field

5 / 35



Modelador Gráfico

- Algorithms → Raster Analysis → Zonal statistics



- Guardar modelo
- Correr con el ícono de Play verde

Modelador Gráfico

- Exportar a Python: Model → Export → Export as Python Script ()
- Copiar código
- Abrir Python console (en Plugins de la ventana ppal.)
- Abrir Editor ()
- Pegar
- Guardar (también se puede guardar desde la ventana que se abre al exportar)
- Editar y agregarle comentarios para hacerlo más legible (ver `_1_agrisuit_us_clean.py`)

Entendiendo y limpiando código

- QGIS nos da el script de geoprocesamiento organizado como una **clase**, con todo el todo el geoprocesamiento dentro del **método** `processAlgorithm`
- También importa un montón de módulos
- Sólo mantendremos los pasos centrales del geoprocesamiento y sólo las importaciones de módulos necesarias
- Eliminar todas las importaciones, clases y definiciones de métodos antes del primer diccionario `alg_params`
- Antes de llegar a la definición de las variables locales, introducir (usar ruta propia en PATH) `maindir = PATH/gis_data`

Replicar Michalopoulos AER (2012)

- 1 Preparar el shapefile WLMS
- 2 Preparar el agricultural suitability raster
- 3 Loopear sobre los archivos raster
- 4 Generar variables - cantidad de idiomas por país, distancias, áreas
- 5 Crear países virtuales
- 6 Estadísticas
- 7 Crear celdas, obtener idiomas en cada celda, variables de control, raster, intersecar con países, vecinos

Inputs

- Languages: WLMS (World Language Mapping System)
<http://www.worldgeodatasets.com/language/> → langa.shp en carpeta
- Agricultural suitability: <https://sage.nelson.wisc.edu/data-and-models/atlas-of-the-biosphere/mapping-the-biosphere/land-use/suitability-for-agriculture/>
- Elevation: (no la descarguen)
http://topex.ucsd.edu/WWW_html/srtm30_plus.html
- Population density for different years
<https://www.pbl.nl/en/image/data>
- Country boundaries <http://www.naturalearthdata.com/downloads/10m-cultural-vectors/10m-admin-0-countries/>
- Coastline <http://www.naturalearthdata.com/downloads/10m-physical-vectors/10m-coastline/>
- Lakes <http://www.naturalearthdata.com/downloads/10m-physical-vectors/10m-lakes/>

1) Preparar el shapefile WLMS

- Usar Modelador Gráfico
- Add layer o localizar en PC **langa.shp** (descargado de worldgeodatasets.com/language cuando era gratis)
- Arreglar las geometrías para procesar el shapefile (siempre habría que hacerlo)
- Añadir el campo ID autoincremental para los países
- Field Calculator para limpiar variable
- Borrar columnas (keep GID, ID, lnm) :
ID_ISO_A3;ID_ISO_A2;ID_FIPS;NAM_LABEL;NAME_PROP;
NAME2;NAM_ANSI;CNT;C1;POP;LMP_POP1;G;LMP_CLASS;
FAMILYPROP;FAMILY;langpc_km2;length
- Output: vector wldsout
- Guardar clean.shp
- Guardar modelo
- Exportar como Python Script y guardar

Arreglar las geometrías

- Agregar layer langa.shp o buscarla en el algoritmo

Properties

Comments

Description

Fix geometries

Input layer

123  langa   

Fixed geometries

 fix_geo 

Dependencies

0 dependencies selected 

Añadir el campo ID autoincremental

Properties


Comments

Description

Add autoincremental field

Show advanced parameters

Input layer

 Using algorithm output "Fixed geometries" from algorithm "Fix geometries"

Field name

123

GID

Start values at [optional]

123

1

✕


↑

↓

Group values by [optional]

123

Incremented

 autoinc_id

✕

Dependencies

0 dependencies selected

...

Generar variable lmn con NAME_PROP de los que tienen menos de 10 caracteres

- Field Calculator para calcular largo variable NAME_PROP
 - Input: Usar anterior desde algorithm output.
 - Field Calculator - Field name: length, Result type: Integer, Expression: length(NAME_PROP)
- Feature filter para eliminar los que tienen más de 10
 - Algorithms → Vector Table → Feature filter
 - Output name: menor_a_11 - Filter Expression: length11 - tick en Final Output
 - Input Layer la anterior usando algorithm output
- Generar variable lmn con NAME_PROP
 - Field Calculator - Description: Field calculator clone - Input layer la anterior usando algorithm output - Field name: lmn - Result type: String - Result length: 10 - Expression: "NAME_PROP"
 - Calculated: field_calc

Borrar columnas


Properties

Comments

Description

Drop field(s)

Input layer




Using algorithm output "Calculated" from algorithm "Field calculator"

Fields to drop

123 ID_FIPS;NAM_LABEL;NAME_PROP;NAME2;NAM_ANSI;CNT;C1;POP;LMP_POP1;G;LMP_CLASS;FAMILYPROP;FAMILY;langpc_km2

Remaining fields



wldsout

Dependencies

0 dependencies selected

Preparar el shapefile WLMS

- Usar Modelador Gráfico (guiarse con *1cleanWLDS.py*)
- Arreglar las geometrías para procesar el shapefile
- Añadir el campo ID autoincremental para los países
- Field Calculator para limpiar variable
- Borrar columnas:
ID_ISO_A3;ID_ISO_A2;ID_FIPS;NAM_LABEL;NAME_PROP;
NAME2;NAM_ANSI;CNT;C1;POP;LMP_POP1;G;
LMP_CLASS;FAMILYPROP;FAMILY;langpc_km2
- Output: vector wldsout
- Guardar clean.shp
- Guardar modelo
- Exportar como Python Script y guardar

2) Preparar el agricultural suitability raster

- Usar Modelador Gráfico
- Add raster o localizar **suit/hdr.adf** (descargado antes de [este link](#))
- Usar *GDAL Warp reproject* para proyectar el raster en WGS 84.
 - Algorithms → Raster Projections → WARP (reproject)
- Usar *GDAL Extract projection* para crear una proyección permanente del raster.
 - Algorithms → GDAL → Raster Projections → Extract Projections
 - Input file: Using algorithm output
 - Create also .prj file: Yes
- Output: raster suitout
- Correr modelo y poner que se guarde como **landquality.gif** (lo vamos a usar después)
- Guardar modelo
- Export to Python

3) Archivos raster

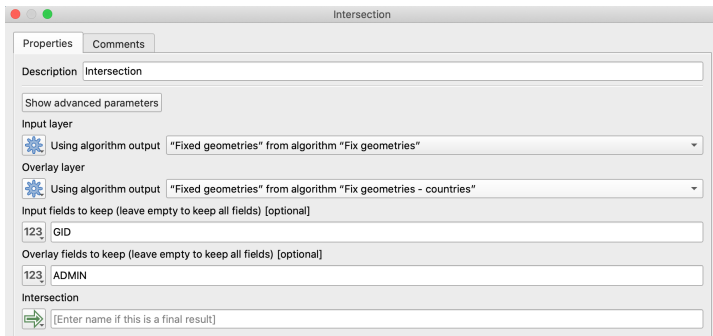
- Usar Modelador Gráfico
- Arreglar las geometrías del shapefile (siempre habría que hacerlo) **ne_10m_admin_0_countries.shp** (descargado de [este link](#)) → **fixgeo_3**
- Borrar columnas → **drop_fields_3** → keep: ADMIN, ISO_A3
- Add rasters: landqual, [popd1800](#), [popd1900](#), [popd2000](#)
- Zonal statistics para cada uno de los rasters (mean)
- Save vector features to file → Saved features → Click en la flecha verde → Values → Save to File → raster_stats.csv (o Click derecho en layer zonalstats → Export → Save Features As... → CSV)
- Guardar modelo y Exportar como Python Script, guardar

4) Otras variables

- **Número de idiomas en cada país:** Intersectar WLMS y países, y hacer estadísticas por categorías.
- **Distancia a la costa:** Encontrar los centroides de los países con *Centroids*, usar *GRASS v.distance* para encontrar la distancia a la costa desde el centroide del país.
- **Áreas de los países:** Reproyectar para tener la proyección adecuada para el cálculo. Usar la calculadora de campo con $\text{area}(\$geometry)/1000000$ como FÓRMULA.

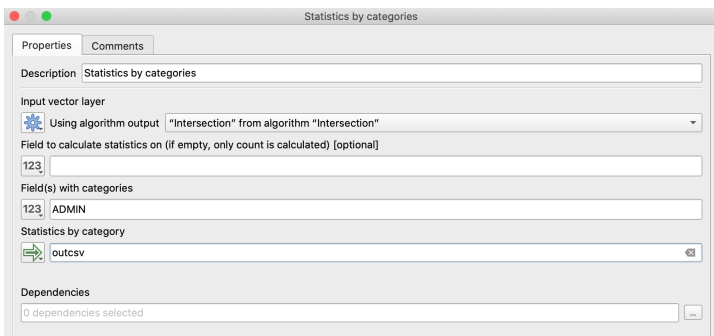
4a) Número de idiomas en cada país:

- Usar Modelador Gráfico
- Fix Geometries del shapefile **clean.shp** sacado de 1) → **fixgeo_wlds**
- Fix Geometries del shapefile **ne_10m_admin_0_countries.shp** → **fixgeo_countries**
- Algorithm → Vector overlay → Intersection



4a cont.) Número de idiomas en cada país:

- Algorithm → Vector analysis → Statistics by categories
- Output: layer **outcsv**
- Repetir y guardar como .csv



4b) Distancia a la costa - calcular centroides

- Usar Modelador Gráfico: modelo4b
- Fix Geometries de **ne_10m_coastline.shp** → **fixgeo_coast** (Agregar *coast* a Description)
- Fix Geometries de **ne_10m_admin_0_countries.shp** → **fixgeo_countries** (Agregar *countries* a Description)
- Algorithm → Vector geometry → Centroids → **country_centroids**
 - Create centroid for each part: No
- Agregar coordenadas a los centroides: Algorithm → Vector geometry → Add geometry attributes → **centroids_with_coordinates**
- Elimino columna de *fixgeo_coast* (*coast* en Description): scalerank → Output: **coastout**
- Eliminar columnas de *centroids_with_coordinates* (*centroids_w_c* en Description) → **centroidsout** → keep: ADMIN, ISO_A3, xcoord, ycoord

4b cont.) Distancia a la costa

- Seguimos en el mismo modelo
- Algorithm → GRASS → v.distance

v.distance

Properties Comments

Description v.distance

Show advanced parameters

'from' vector map

Using algorithm output "Remaining fields" from algorithm "Drop field(s) - centroids_w_c"

'to' vector map

Using algorithm output "Remaining fields" from algorithm "Drop field(s) - coast"

Maximum distance or -1.0 for no limit [optional]

123 -1.000000

Minimum distance or -1.0 for no limit [optional]

123 -1.000000

'upload': Values describing the relation between two nearest features

123 1 options selected

Column name(s) where values specified by 'upload' option will be uploaded

123 xcoord

Column name of nearest feature (used with upload=to_attr) [optional]

123

Nearest

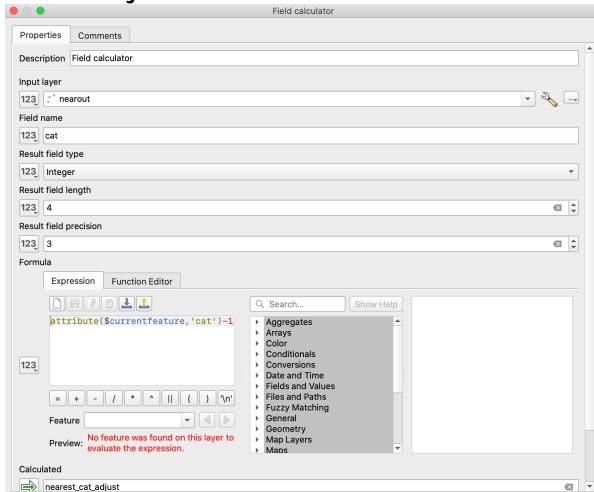
nearout

Distance

distout

4b cont.) Distancia a la costa - corrijo variable

- Seguimos en el mismo modelo
- Algorithm → Vector Table → Field Calculator → **nearest_cat_adjust**



4b cont.) Distancia a la costa - drop

- Seguimos en el mismo modelo
- Algorithm → Vector Table → Drop Fields: xcoord;ycoord → **nearest_cat_adjust_dropfields**

4b cont.) Distancia a la costa - merge

- Algorithm → Vector General → Join attributes by field value
- **centroids_nearest_coast_joined**

Join attributes by field value

Properties Comments

Description Join attributes by field value

Input layer
123 : centroidsout

Table field
123 ne_10m_adm

Input layer 2
123 : nearest_cat_adjust_dropfields

Table field 2
123 ne_10m_adm

Layer 2 fields to copy (leave empty to copy all fields) [optional]
123

Join type
123 Take attributes of the first matching feature only (one-to-one)

Discard records which could not be joined
123 No

Joined field prefix [optional]
123

Joined layer [optional]
centroids_nearest_coast_joined

Unjoinable features from first layer [optional]
[Enter name if this is a final result]

4b cont.) Distancia a la costa - drop y merge

- Drop Fields: ne_10m_adm_2;ADMIN_2;ISO_A3_2 → **centroids_nearest_coast_joined_dropfields**
- Algorithm → Vector General → Join attributes by field value

Join attributes by field value

Properties Comments

Description Join attributes by field value

Input layer
123 V distout

Table field
123 cat

Input layer 2
123 centroids_nearest_coast_joined_dropfields

Table field 2
123 cat

Layer 2 fields to copy (leave empty to copy all fields) [optional]
123

Join type
123 Take attributes of the first matching feature only (one-to-one)

Discard records which could not be joined
123 No

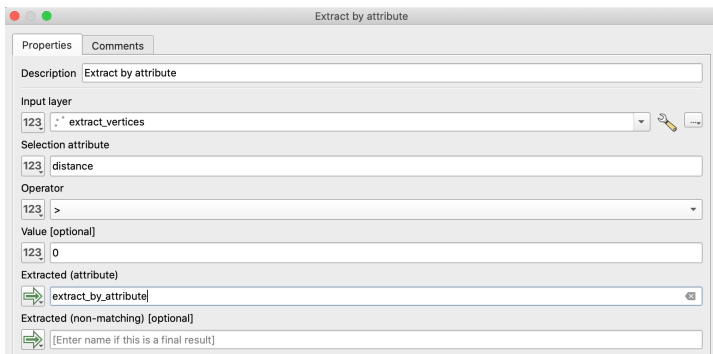
Joined field prefix [optional]
123

Joined layer [optional]
123 centroids_nearest_coast_distance_joined

Unjoinable features from first layer [optional]
123

4b cont.) Distancia a la costa - extraer vértices y filtrar

- Seguimos en el mismo modelo
- Algorithm → Vector Geometry → Extract vertices → `extract_vertices`
- Algorithm → Vector Selection → Extract by attribute



4b cont.) Distancia a la costa - variables de latitud y longitud del centroide

- Seguimos en el mismo modelo
- Field Calculator
 - Input: extract_by_attribute
 - Field name: cent_lat
 - Type: Float
 - Length: 10
 - Precision: 10
 - Formula: attribute(\$currentfeature,'ycoord')
 - Output **added_field_cent_lat**
- Field Calculator
 - Input: added_field_cent_lat
 - Field name: cent_lon
 - Mismo type, length, precision
 - Formula: attribute(\$currentfeature,'xcoord')
 - Output: **added_field_cent_lon**

4b cont) Distancia a la costa - drop, geometria

- Seguimos en el mismo modelo
- Drop Fields de **added_field_cent_lon**: xcoord;ycoord;fid_2;cat_2; vertex_index;vertex_part;vertex_part;_index;distance;angle (keep: ADMIN, ISO_A3, cent_lat, cent_lon) → **centroids_lat_lon_drop_fields**
- Algorithm → Vector Geometry → Add geometry attributes
 - Input: centroids_lat_lon_drop_fields
 - Layer CRS
 - Output: **add_geo_coast**

4b cont) Distancia a la costa - variables de latitud y longitud de la costa

- Seguimos en el mismo modelo
- Field Calculator
 - Input: add_geo_coast
 - Field name: coast_lat
 - Type: Float
 - Length: 10
 - Precision: 10
 - Formula: attribute(\$currentfeature,'ycoord')
 - Output added_field_coast_lat
- Field Calculator
 - Input: added_field_coast_lat
 - Field name: coast_lon
 - Mismo type, length, precision
 - Formula: attribute(\$currentfeature,'xcoord')
 - Output: added_field_coast_lon

Distancia a la costa - guardar como CSV

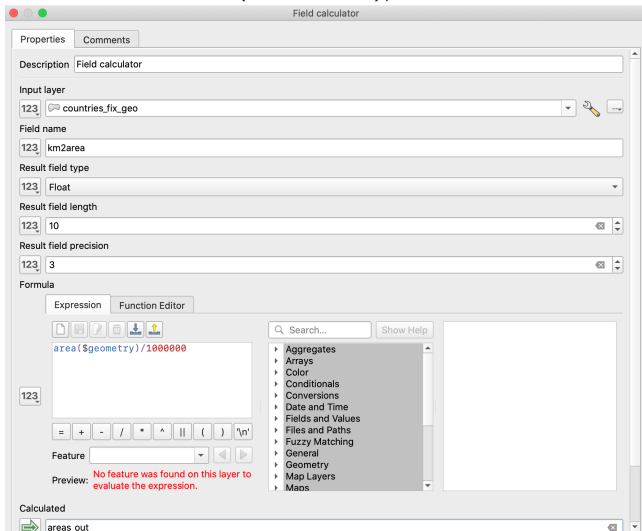
- Seguimos en el mismo modelo
- Drop Fields de **added_field_coast_lon**: xcoord;ycoord → **csvout**
- Guardar modelo
- Exportar como Python Script y guardar
- Save vector features to file → Saved features → Click en la flecha verde → Values → Save to File → dist_coast.**csv**

4c) Calcular área de los países

- Usar Modelador Gráfico: modelo4c
- Drop fields de **ne_10m_admin_0_countries.shp** → **countries_drop_fields**
 - keep ADMIN,ISO_A3
- Reproject Layer (vector, no raster)
 - Input: countries_drop_fields
 - Ir al ícono del mundito y buscar ESRI:54034 (defining world cylindrical equal area) → **countries_reprojected**
- Fix geometries → **countries_fix_geo**

4c) Calcular área de los países

- Field Calculator → $area(\$geometry)/1000000$



4c) Calcular área de los países - guardar como csv

- Save vector features to file
 - Vector features: areas_out
 - Save vector features to file → Saved features → Click en la flecha verde → Values → Save to File → areas_out.csv
 - Guardar Modelo
 - Exportar Python