# Workshop

## *An introduction to Raster GIS Data: Learning the Pythonic Way (Parte 1)*

**16/10/2015**

**Cayetano Benavent Viñuales**
GIS Analyst - Geographica
cayetano.benavent@geographica.gs

# Sumario de contenidos

## Antes de empezar...

Este Workshop no es una introducción general a los Sistemas de Información Geográfica (SIG).

Aquella persona que quiera introducirse en ese maravilloso mundo y tenga tiempo (¡y ganas!), le recomiendo la siguiente lectura:

***Curso de Introducción a QGIS***

Este curso, a través del software QGIS, introduce al alumno los conceptos básicos en materia de SIG.

## Antes de empezar...

Si alguien quiere avanzar un paso más, puede continuar la senda iniciada con este otro curso:

***Curso avanzado de QGIS***

Ahora sí, ¡empecemos!

# ¿Qué son los datos raster (contexto GIS)?

*"A data model in which geographic features are **represented using discrete cells**, generally squares, arranged as a (contiguous) **rectangular grid**.*

*A single grid is essentially the same as a **two-dimensional matrix**, but is typically **referenced from the lower left corner** rather than the norm for matrices, which are referenced from the upper left.*

*Raster files may have one or more values (attributes or **bands**) associated with each cell position or pixel."*

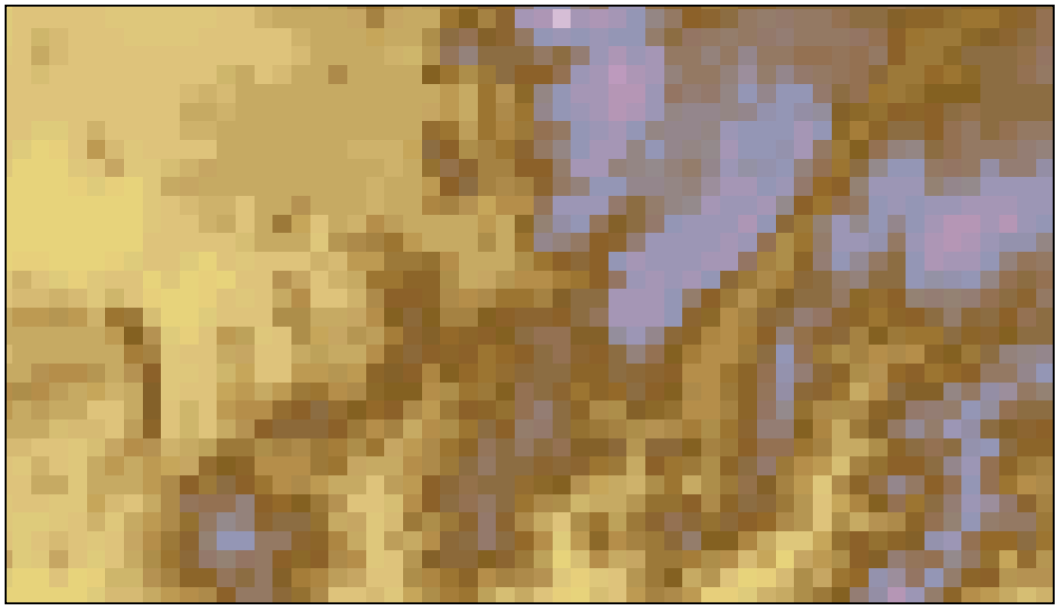
(M. de Smith, M. Goodchild and P. Longley, 2015)

## Formato ráster

- Representación **continua** de la realidad.

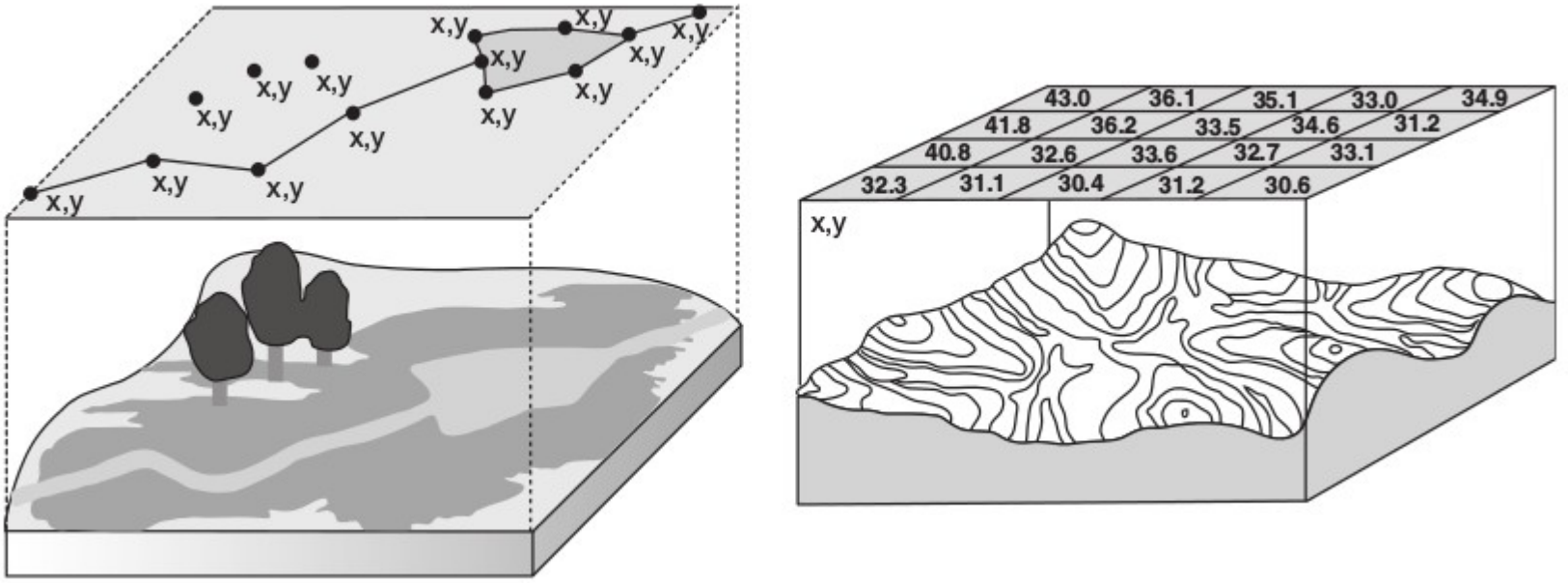
- El formato ráster **está basado en celdas** de igual tamaño (generalmente) que poseen un valor.

- El tamaño de la celda define el nivel de detalle de la información.



*La altimetría del terreno de la figura es un ejemplo de formato ráster.*

# GIS Data: Vector vs. Raster




(J. Albretch, 2007)

## Ventajas e inconvenientes de cada modelo

- Vectorial:
  - Estructura de datos más compleja, pero más compacta.
  - Relaciones topológicas muy potentes.
  - Muy adecuado para salidas cartográficas.
  - Geoprocesamiento requiere algoritmos muy complejos.

- Ráster:
  - Estructura de datos muy simple, pero de más consumo de recursos de almacenamiento.
  - Geoprocesamiento muy potente y fácil de implementar.
  - Relaciones topológicas pobres.
  - Menos adecuado para salidas cartográficas.

## ¿Qué modelo elegir: vectorial o ráster?

**Depende** completamente del objetivo del trabajo que estemos desarrollando. Habría que responder a varias preguntas para saberlo:

- ¿Es un fenómeno continuo en el espacio?
- ¿Importa la precisión de los límites?
- ¿Son necesarias las relaciones topológicas?
- ¿Qué tipo de análisis o cálculos voy a realizar sobre los datos?

La conclusión es que ambos modelos son complementarios. Depende del fin perseguido, usaremos uno u otro (normalmente los dos combinados).

# Leer y escribir datos raster: GDAL y Rasterio


- Leer y escribir datos GIS raster, en el mundo del software libre, es casi sinónimo de trabajar con GDAL: http://www.gdal.org/

- Desde la v1.3.2, GDAL y OGR se unen en un mismo proyecto (pertenecen al mismo “source tree”).

- GDAL es la capa de lectura/escritura raster de casi todo el software libre geo (¡y de mucho software no libre!):
  https://trac.osgeo.org/gdal/wiki/SoftwareUsingGdal

- GDAL suele ser la primera capa de cualquier desarrollo o flujo de geoprocesamiento que diseñemos para trabajar con datos raster.

# Leer y escribir datos raster: GDAL y Rasterio


- GDAL soporta actualmente (octubre 2015) unos 142 formatos:

  http://www.gdal.org/formats_list.html

- GDAL permite escritura en una gran parte de los formatos soportados (no todos).

- Muchos formatos requieren compilación específica de ciertos drivers (o librerías dependientes).

- Con cada versión nueva, se añaden multitud de formatos y nuevas capacidades.

# Leer y escribir datos raster: GDAL y Rasterio


- Con la versión actualmente estable (GDAL 2.0), la API cambia por completo. Su integración con OGR es ya muy elevada.

- Hay que tener en cuenta que la mayor parte del software libre que depende de GDAL para leer o escribir (casi todo!!), aún está en proceso de adptación al cambio de v1.X a v2.0.

- Para ver un resumen con las principales novedades, recomiendo leer la presentación de Even Roualt (actual PSC Chair) del FOSS4Geo2015:

http://download.osgeo.org/gdal/presentations/GDAL%202.0%20overview%20(FOSS4G-E%202015).pdf

# Leer y escribir datos raster: GDAL y Rasterio


- GDAL se distribuye bajo una licencia de estilo MIT/X:
  https://trac.osgeo.org/gdal/wiki/FAQGeneral#WhatlicensedoesGDALOGRuse

- Está escrita en C/C++, y su API principal, obviamente, se accede desde C/C++.

- Hay APIs en otros lenguajes. Las activas actualmente son:  C#, Java, Perl, and Python.

- Otras APIs a GDAL externas al proyecto (“outside of the GDAL source tree”): Go, Lua, Node.js, R, Tcl, and VB6.

# Leer y escribir datos raster: GDAL y Rasterio

- La API de Python es probablemente la más utilizada. La wiki oficial es:

  https://trac.osgeo.org/gdal/wiki/GdalOgrInPython


- La documentación de la API (generada automáticamente):

  http://gdal.org/python/

- GDAL provee un conjunto de utilidades de línea de comando muy potentes:

  http://www.gdal.org/gdal_utilities.html

- Estas utilidades, escritas en C/C++ o en Python, son excelentes ejemplos de cómo manejar la API de GDAL. Los test de GDAL están casi todos escritos en Python, por lo que estudiarlos es otra manera de aprender: https://svn.osgeo.org/gdal/trunk/autotest/

# Leer y escribir datos raster: GDAL y Rasterio


- Aunque abundante, la información sobre GDAL es farragosa de entender para usuarios no expertos del mundo GIS. La librería funciona a la perfección, pero es algo "barroca".

  Recomiendo por ello, acceder a lecturas externas más didácticas, como el Cookbook de Jared Erickson:

  http://pcjericks.github.io/py-gdalogr-cookbook/index.html

  y los cursos de Chris Garrard: http://www.gis.usu.edu/~chrisg/python/2009/

  Las últimas son un clásico, pero están algo desactualizadas. C. Garrard está a punto de publicar un libro, que habría que comprar: https://www.manning.com/books/geoprocessing-with-python

- En cualquier caso, estemos atentos, porque la v2.1 trae novedades cruciales. Leed este post de hace varios días de Even Roualt:

  http://erouault.blogspot.com.es/2015/10/gdal-and-ogr-utilities-as-library.html

# Leer y escribir datos raster: GDAL y Rasterio


- Antes de empezar a trabajar con GDAL, siempre debemos asegurarnos de dos cosas:

1. Versión de GDAL instalada:

```
$ gdalinfo --version
```

2. Formatos soportados en la versión que tenemos instalada:

```
$ gdalinfo --formats
```

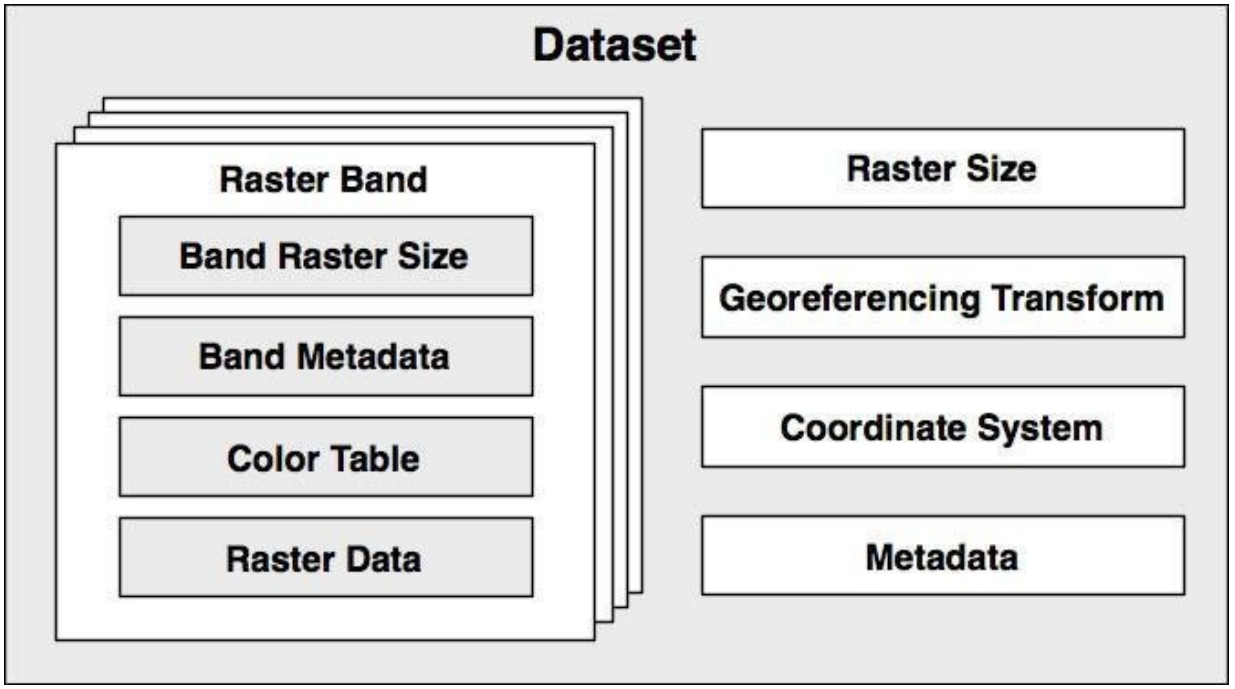
Ver detalles de formatos concretos:

```
$ gdalinfo --format GTIFF
```

# Leer y escribir datos raster: GDAL y Rasterio

- Aprendamos ahora a leer y escribir datos raster desde GDAL. En primer lugar, debemos conocer cómo se estructura el modelo de datos raster en GDAL:

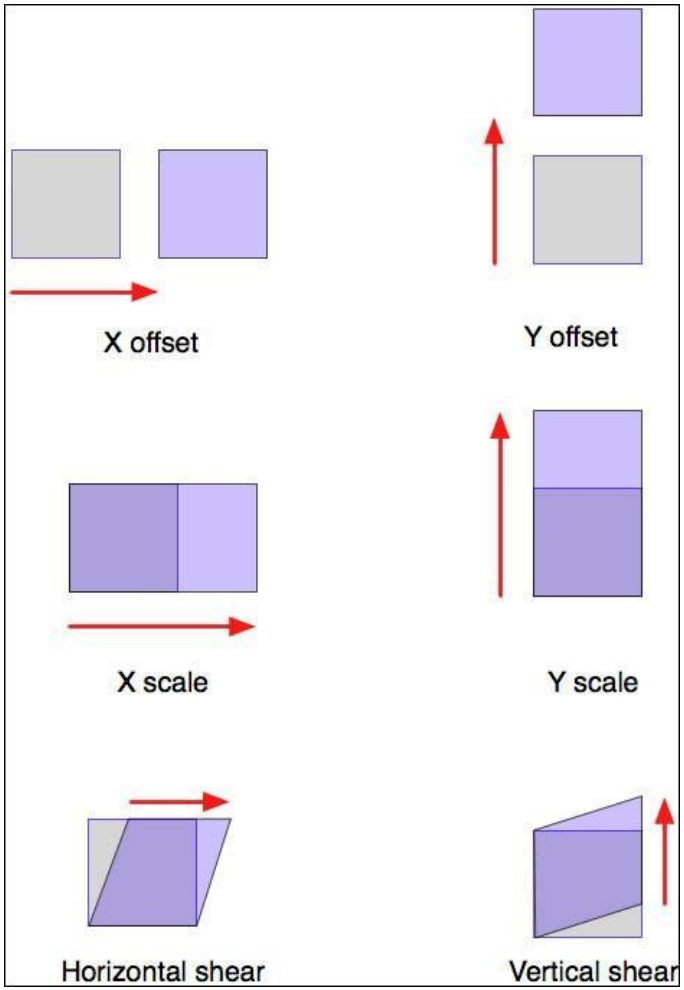
(Detalles en http://www.gdal.org/gdal_datamodel.html)




(E. Westra, 2014)

# Leer y escribir datos raster: GDAL y Rasterio

- La geotransformación se construye con los siguientes parámetros:




(E. Westra, 2014)

# Leer y escribir datos raster: GDAL y Rasterio

- Ejercicio 1: Leer datos raster con GDAL:

https://github.com/GeographicaGS/workshop_Raster_GIS_data/blob/master/code/raster_workshop_gdal_read.ipynb

https://github.com/GeographicaGS/workshop_Raster_GIS_data/blob/master/code/raster_workshop_gdal_read.py

- Ejercicio 2: Escribir datos raster con GDAL (conversión de formatos):

https://github.com/GeographicaGS/workshop_Raster_GIS_data/blob/master/code/raster_workshop_gdal_write.ipynb

https://github.com/GeographicaGS/workshop_Raster_GIS_data/blob/master/code/raster_workshop_gdal_write.py

# Leer y escribir datos raster: GDAL y Rasterio

- Rasterio es una gran alternativa para leer y escribir datos raster desde Python. Desarrollada por Sean Gilles de Mapbox.

- No es un wrapper al binding de Python. Construida sobre GDAL con Cython.

- Es más sencilla de manejar que el binding “oficial”, más rápida y mucho más Pythónica.

- Viene con una interfaz de línea de comando (Rio):

```
$ rio info data/mde/h10_1050_2-2/h10_1050_2-2.tif
```

Hay una alternativa paralelizada: `$ rio-mucho`

# Leer y escribir datos raster: GDAL y Rasterio

- Lecturas interesantes para aprender Rasterio:

http://sgillies.github.io/foss4g-2014-fiona-rasterio

https://github.com/sgillies/frs-cheat-sheet

*"- A Python package at the top.*
*- Extension modules (using Cython) in the middle.*
*- Fast loops, typed memoryviews, "nogil" blocks.*
*- GDAL shared library on the bottom."*

# Leer y escribir datos raster: GDAL y Rasterio

- Ejercicio 3: Leer y escribir datos raster con Rasterio (conversión de formatos):

https://github.com/GeographicaGS/workshop_Raster_GIS_data/blob/master/code/raster_workshop_rasterio.ipynb

https://github.com/GeographicaGS/workshop_Raster_GIS_data/blob/master/code/raster_workshop_rasterio.py

# Georreferenciar datos raster: Rasterio

- GDAL posee una herramienta muy potente para la georreferenciación de datos raster: GDALWARP

- La documentación de la API de GDALWARP:
  http://www.gdal.org/warptut.html

- La utilidad de la línea de comando GDALWARP:
  http://www.gdal.org/gdalwarp.html

- El binding de Python no tiene acceso todavía (en la v2.1 parece que sí!!) a esta herramienta.

- Rasterio, por el contrario, provee acceso directo a GDALWARP ¡y muy eficiente!

# Georreferenciar datos raster: Rasterio

- Ejercicio 4: Georreferenciando datos raster con Rasterio:

https://github.com/GeographicaGS/workshop_Raster_GIS_data/blob/master/code/raster_workshop_rasterio_reproject.ipynb

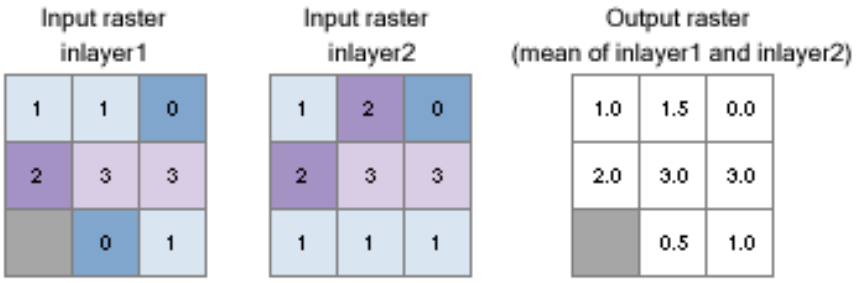
https://github.com/GeographicaGS/workshop_Raster_GIS_data/blob/master/code/raster_workshop_rasterio_reproj.py

# Álgebra de mapas: Rasterio + Numpy

Es el lenguaje de análisis ráster por excelencia.

Permite realizar análisis y modelado de la información geográfica a través de expresiones algebraicas (no hay que olvidar que una capa ráster está formada por celdas, cada una de las cuales tiene un valor numérico con el que podemos operar algebraicamente).

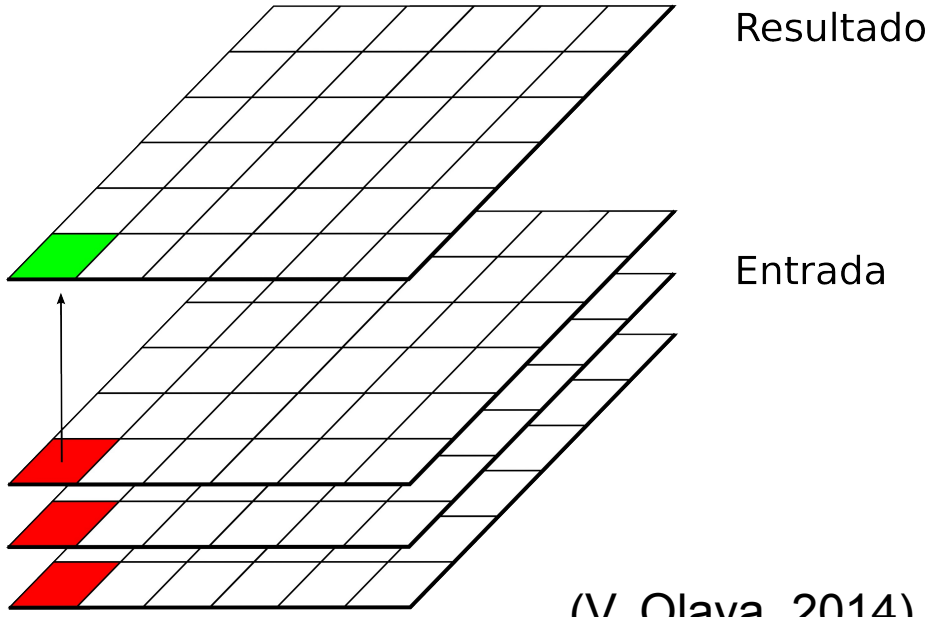
La expresión `([inlayer1]+[inlayer2])/2` podemos observar como se desarrolla en la figura adjunta



# Álgebra de mapas: Rasterio + Numpy

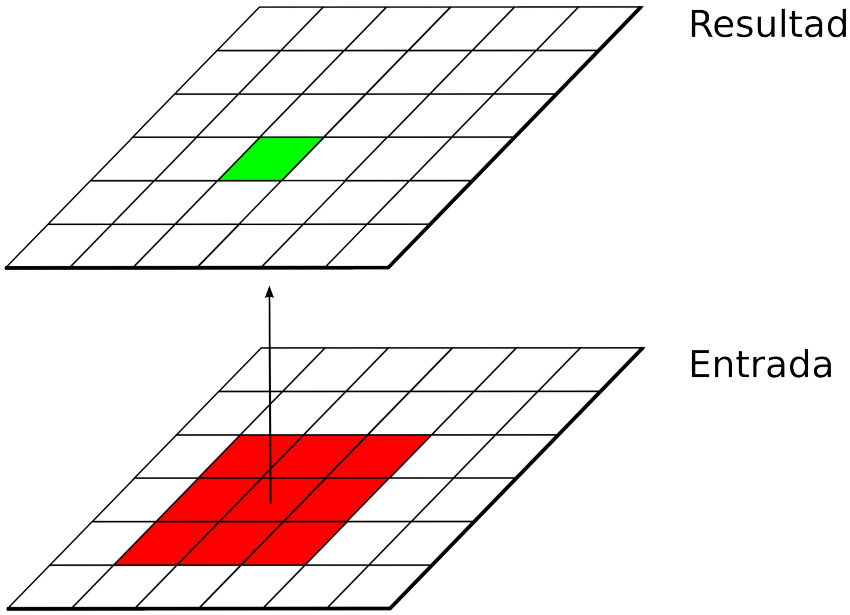
Podemos decir que hay 4 grandes tipos de álgebra de mapas:

**Local**. El valor en cada celda de la capa resultante es función únicamente de los valores en esa misma celda en las capas de partida.



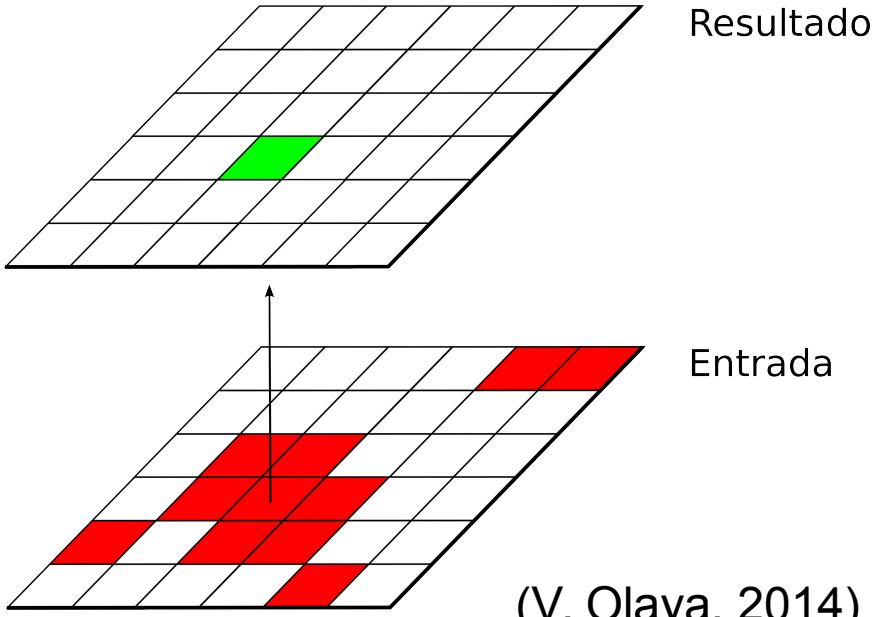

(V. Olaya, 2014)

**Focal.** El valor en cada celda de la capa resultante es función del valor en dicha celda y en las situadas en un entorno definido alrededor de la misma.



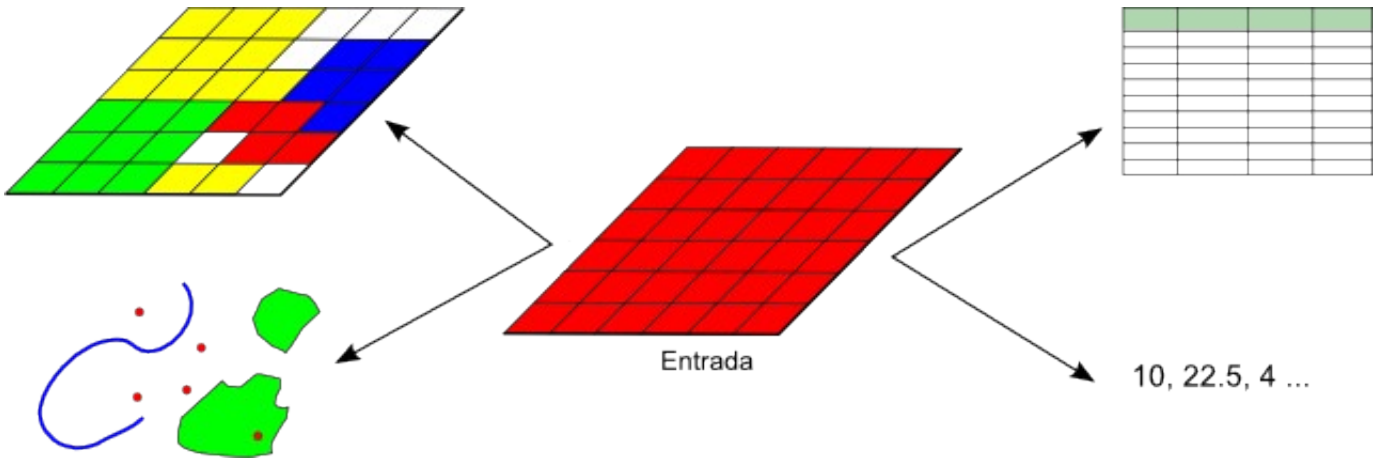

(V. Olaya, 2014)

**Zonal o regional**. El valor en cada celda de la capa resultante es función del valor de todas las celdas conectadas a esta que presentan un mismo valor para una de las capas de entrada (pertenecen a la misma clase que esta).




(V. Olaya, 2014)

**Global.** El valor resultante de la función es obtenido a partir de todas las celdas de la capa.




(V. Olaya, 2014)

# Álgebra de mapas: Rasterio + Numpy

Ejercicio 5: vamos a realizar un ejercicio sobre álgebra de mapas (Local). El problema a resolver es el siguiente: tenemos como datos de partida la modelización numérica de una predicción meteorológica de temperatura (unidad: k) - del modelo GFS (Global Forecast System) – en dos niveles verticales diferentes: 2 m y 80 m. Queremos obtener la modelización en grados centígrados, y calcular la diferencia de temperatura entre ambas predicciones.

```
Temperature:K (instant):regular_ll:heightAboveGround:level 80
   m:fcst time 6 hrs:from 201509030000
```

```
Temperature:K (instant):regular_ll:heightAboveGround:level 2
   m:fcst time 6 hrs:from 201509030000
```

# Álgebra de mapas: Rasterio + Numpy

- Ejercicio 5: Álgebra de mapas con Rasterio y Numpy:

https://github.com/GeographicaGS/workshop_Raster_GIS_data/blob/master/code/raster_workshop_mapalgebra.ipynb

https://github.com/GeographicaGS/workshop_Raster_GIS_data/blob/master/code/raster_workshop_mapalgebra.py

# Bibliografía


- Albrecht, J. (2007): “Key Concepts and Techniques in GIS”. SAGE.

- Longley, P., Goodchild, M., Maguire, D., Rhind, D. (1999): “Geographical Information Systems: Principles, Techniques, Management and Applications”, 2nd Edition. Wiley.

- Longley, P., Goodchild, M., Maguire, D., Rhind, D. (2005): “Geographical Information Systems and Science”, 2nd Edition. Wiley.

- Mitas, L., Mitasova, H. (1999): *Spatial Interpolation.* In Longley, P., Goodchild, M., Maguire, D., Rhind, D. (Eds.) “Geographical Information Systems: Principles, Techniques, Management and Applications”. Wiley.

- Smith, M., Goodchild, M., Longley, P. (2015): “Geospatial Analysis”, 5th Edition. The Winchelsea Press.

- Wang, J.F., Stein, A., Gao, B.B., Ge, Y., (2012): *A review of spatial sampling*. Spatial Statistics Vol.2, pp 1-14.