Introduction to QGIS practical

David Orme

This practical is a quick introduction to GIS and a few techniques for exploring and using geographic datasets.

Quantum GIS (QGIS)

We are going to be using QGIS, an open-source GIS program which has evolved in recent years to be a strong competitor to ArcGIS as a personal GIS platform. QGIS has a pretty decent user interface and it is also possible to script and automate analyses using QGIS. Note that many other GIS packages exist - some quite specialized, some quite broad - and all the techniques we will be looking at form the common core of GIS that will apply across particular platforms.

One of the strengths of QGIS is that – in addition to providing a GIS interface of its own – it also integrates tools from a range of other GIS programs. These include:

- The Geospatial Data Abstraction Library (GDAL, `http://www.gdal.org/`), which provides a suite of tools and programs for converting and manipulating spatial data
- GRASS GIS (`http://grass.osgeo.org/`), which is a complete GIS system in its own right, but has until recently been largely a command line application.

Installing QGIS

If you need to install QGIS on your own laptop, then official instructions for Windows, Mac and Linux are available from this link:

`https://www.qgis.org/en/site/forusers/download.html`

Practical structure

We will concentrate on using the QGIS graphical user interface in order to learn the basic techniques and structure of GIS data and tools. The text describes the general background to each technique and the actual practical steps are highlighted in green:

- This is a practical step – you need to follow these instructions.

Hopefully the practical should be fairly self explanatory but there is a lot of help online for QGIS, so if you need more help on a topic later on, try starting here:

`http://www.qgis.org/en/docs/index.html`

However, in addition to the main user interface and like many of the tools we will use across the Masters courses at Silwood, these GIS tools can be controlled from the command line, allowing you to build powerful GIS scripts to automate and replicate your analyses. The practical handout will also provide some comments on how to go about doing this. They will come in boxes like this and will assume that you have some familiarity with using command line tools and scripting.

Programming GIS

There are more detailed installation instructions for specific operating systems here:

`https://www.qgis.org/en/site/forusers/alldownloads.html`

On Windows, you will need to install the Open Source GIS for Windows (OSGeo4W) tools, available as the 'Advanced Users' link on QGIS Download page. The project home page is here:

`https://trac.osgeo.org/osgeo4w`

On a Mac, you may also need to install some other software frameworks from this page:

`http://www.kyngchaos.com/software/frameworks`

On Linux, the standard installation *should* ensure that all of the required tools are installed in sensible places so that they can be called from the command line.

Practical One

GIS data - a mess of formats

GIS data files are complicated. One GIS data set often consists of multiple linked files that contain different bits of information. We will start working with a folder of data downloaded from Blackboard.

- We will be creating and using some quite large files. If you are using a college Windows desktop computer, the default is to store everything on your network drive, which can make everything rather slow. To work on the local machine, create a new folder on the local hard drive" 'C:/Temp/GIS_practicals'.

  Do include the underscore not a space - spaces in filenames are often a problem when using scripts. You may get a warning when finding 'C:/Temp' that these are hidden files that you shouldn't need to see: ignore this and tell the computer to reveal the files.

  If you are using your own computer, save these files wherever you like!

- Open Blackboard and navigate to the GIS module page. Go to the Practicals folder and download the zip file 'QGIS_Practical_Files.zip' into the directory you just created.
- You now need to extract the files from the archive. The commands will vary if you are using Windows, Mac or Linux but double clicking or right clicking on the zip archive will probably work. Don't be fooled by Windows, which will allow you to look inside the archive without extracting the files.

The practical files consist of three folders containing files for three different regions. You need to keep GIS files very well organised as GIS tends to be a messy business with lots of files, possibly from lots of different locations if you have many projects.

GIS 'files' often contain more than one part – it is common for a particular GIS dataset to have a data file and then separate files describing the spatial part of the data. One common example is the 'shapefile' to contain vector data, which is actually a set of *at least* four files:

shp The coordinates of the GIS vector features. A shapefile will only contain one type of point, line or polygon data.

shx The extent (or limits) of the coordinates. These are used so often, it is convenient to have them stored in a separate file.

prj The projection system of the coordinates.

dbf A database file containing the attributes of each geographic feature.

Another example is the raster '.bil' format (which stands for *b*and *i*nter*l*eaved data). This actually consists of two files: the '.bil' file itself, which contains the data and a header file ('.hdr') which contains georeferencing information.

The key point here is that you have to keep all of the different parts of GIS files together for the data to work. Be careful when moving GIS files!

Projections

Because the earth is not flat, any map is a projection of the surface of the Earth onto a flat surface, whether it is a printed map or GIS data on a screen. Some of these (see below; Figure 1) are easy to envisage and others are much more mind-bending. There are hundreds of different projections of which three easily visualised ones are:

Azimuthal Imagine a sheet of paper held flat against the surface of the earth.

Cylindrical Imagine a sheet of paper rolled into a cylinder around the earth, again cut and flattened out.

Conical Imagine a sheet of paper rolled into a cone and sat on the earth, then cut and flattened out.

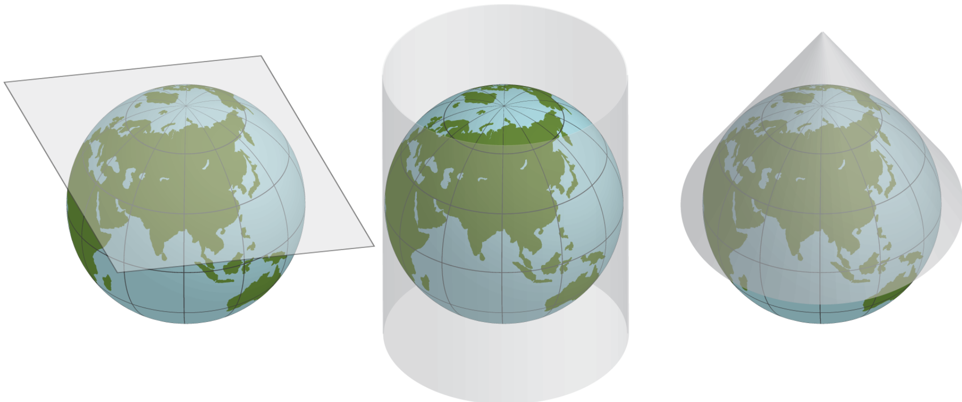

Figure 1: *Visualising projections from a sphere onto a planar surface*

This picture is complicated by the fact that there are really two components that define a projection.

1. The *geographic coordinate system* - the position of the points on the surface of the Earth as latitude and longitude. The geographic coordinates will vary depending on the model used to describe the surface of the earth. The model (or datum) consists of a spheroid — the shape of the earth in terms of radius and flattening — and this can be local, providing an extremely good local model of the Earth's surface but a poor global one, or global, providing a model for larger scale maps.

2. The *projected coordinate system*. The projected coordinates are the XY positions of latitude and longitude points on the flat surface.

What this boils down to is that if you want a working GIS database then you need to remember that datasets come with a projection. You must keep track of these projections or your data are not going to be where you think they are.

- Now start QGIS. You should see some menu ribbons at the top, panels called 'Browser' and 'Layers' on the left and an open white panel in the centre where the data will appear.
- In the Browser panel, navigate to your new practical folder and open the Global folder.
- Select the Tissot.shp, Cntry98.shp, Background.shp and bio1.tiff files (three vector datasets and a raster dataset of global mean annual temperature) and click on the Add button at the top of the browser pane.
- The datasets will appear in the Layer panel. Click and drag on the names to change the display order so that Tissot is at the top, Cntry98 and then bio1 below it and finally Background at the bottom.

You are now looking at a dataset of the countries of the world as they were in 1998. The Tissot file contains Tissot indicatrices: these are perfect circles drawn around a set of points on the surface of the earth. All points on the boundary of the circles are equidistant from the centre and they cover the same area but shape and distance are often distorted in map projections – the Tissot indicatrices help you see this distortion.

The data are currently displayed in equirectangular projection: the longitude and latitude are simply being treated as X and Y coordinates. This leads to some odd distortion – the Tissot circles show how much distortion is going on – and some very odd properties – the poles (90° N/S from any longitude) are stretched out into lines along the top and bottom of the map!

- The colours are probably very ugly. Right click on Tissot in the Layer panel and choose 'Properties' from the drop down menu. Go to the Style tab and change the display colour. Repeat this with the other two layers until you have something you are happy with.
- In order to see the temperature data underneath the countries, you will need click on the Simple Fill box for the Cntry98 style tab and change the Fill style to No Brush. You should end up with something like Figure 2.

Note that we *can* see through bits of the temperature raster map to see the background colour – this dataset only has values for terrestrial temperatures and the other raster cells are filled with a special code to show that there is *no data* for that location. We'll now look at changing the projection used to **display the data**: note that this does nothing at all to the data in the files.

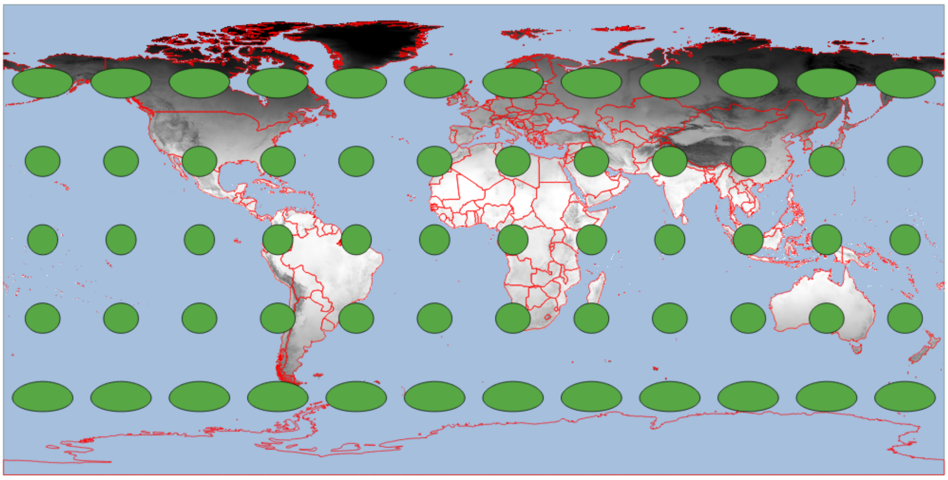

Figure 2: *Countries of the world and mean annual temperature in equirectangular projection*

- Move the cursor around over the map - note how the coordinate box below the map shows the cursor location in the current projection system.
- From the Project menu, choose Project Properties and then the CRS tab ('Coordinate Reference System'). Make sure the 'Enable on the fly CRS transformation' box is checked and then type 'America' into the Filter box. You will see a list of CRS's: select 'North_America_Equidistant_Conic' and click OK.

This will look very odd: the North Pole is a point in the centre, but the South Pole is now a circle around the outside of the map. There's also an odd wedge at the top where the cone of the projection has been 'cut' and spread out to flatten the map, stretching everything that crosses it. So - why would we want to use this?

- Select the zoom tool from the icons at the top () and zoom in on the USA. Notice how the Tissot circles here are pretty round – this projection provides a good basis for continental maps of North America but is bad for most other locations.
- Zoom back out () and try different CRS transformations:
  - 'WGS 84 / World Mercator' – the Tissot circles are the right shape but their area isn't the same;
  - 'WGS 84 / NSIDC EASE-Grid Global' – the Tissot circles are distorted but all have the same area.

You may find that the data doesn't draw properly or is shown as a series of blocks rather than the full detail of the coastline. Transformation is computationally intensive and this 'on-the-fly' transformation doesn't always work – you may need to zoom in to look at smaller areas to see the data in more detail.

Once you have decided which projection is appropriate for a project you are working on, you need to make sure that all your data is re-projected into that projection so that all of this can be avoided. That is what we will do now!

Reprojection and raster calculation

We're going to calculate NDVI and EVI from MODIS satellite data, using the files in the Borneo practical folder.

- Close your current QGIS Project and create a new one.
- From the Borneo folder, add the three rasters: MODIS_red_reflectance.tif, MODIS_NIR_reflectance.tif and MODIS_blue_reflectance.tif.
- You should see something like Figure 3. Look at the bottom right of your QGIS screen - you should see the box showing the projection is displaying something like USER: 100000.

These files are a product from the MODIS satellite sensor. The raw sensor data has been processed to georeference and calibrate it. It has also been converted from radiance to reflectance values:

- **Radiance**: how much the sensor is illuminated by light from the pixel.
- **Reflectance**: how much of the light hitting that area of the ground has been reflected back up to the sensor. This calculation involves complex corrections for atmospheric absorption, the incident angle of the light and a host of other factors.

This data has an unusual projection - one that has been chosen by NASA to distribute MODIS data. Figure 4 shows how the panels are arranged and numbered: we are looking at data from panel `h29v08`.

- Add the shapefile layer 'SAFE_Layout_UTM50N_WGS84.shp' to your project from the Borneo folder and then: right click on the layer name and choose 'Zoom to Layer'.

This shapefile is in the Universal Transverse Mercator projection, in Zone 50N, using the WGS 84 datum. It is the main projection we use for GIS at SAFE because it preserves shape and area well. This is the whole point of UTM: it defines a set of global zones that provide reasonable local projections for most locations:

`https://en.wikipedia.org/wiki/Universal_Transverse_Mercator_coordinate_system`

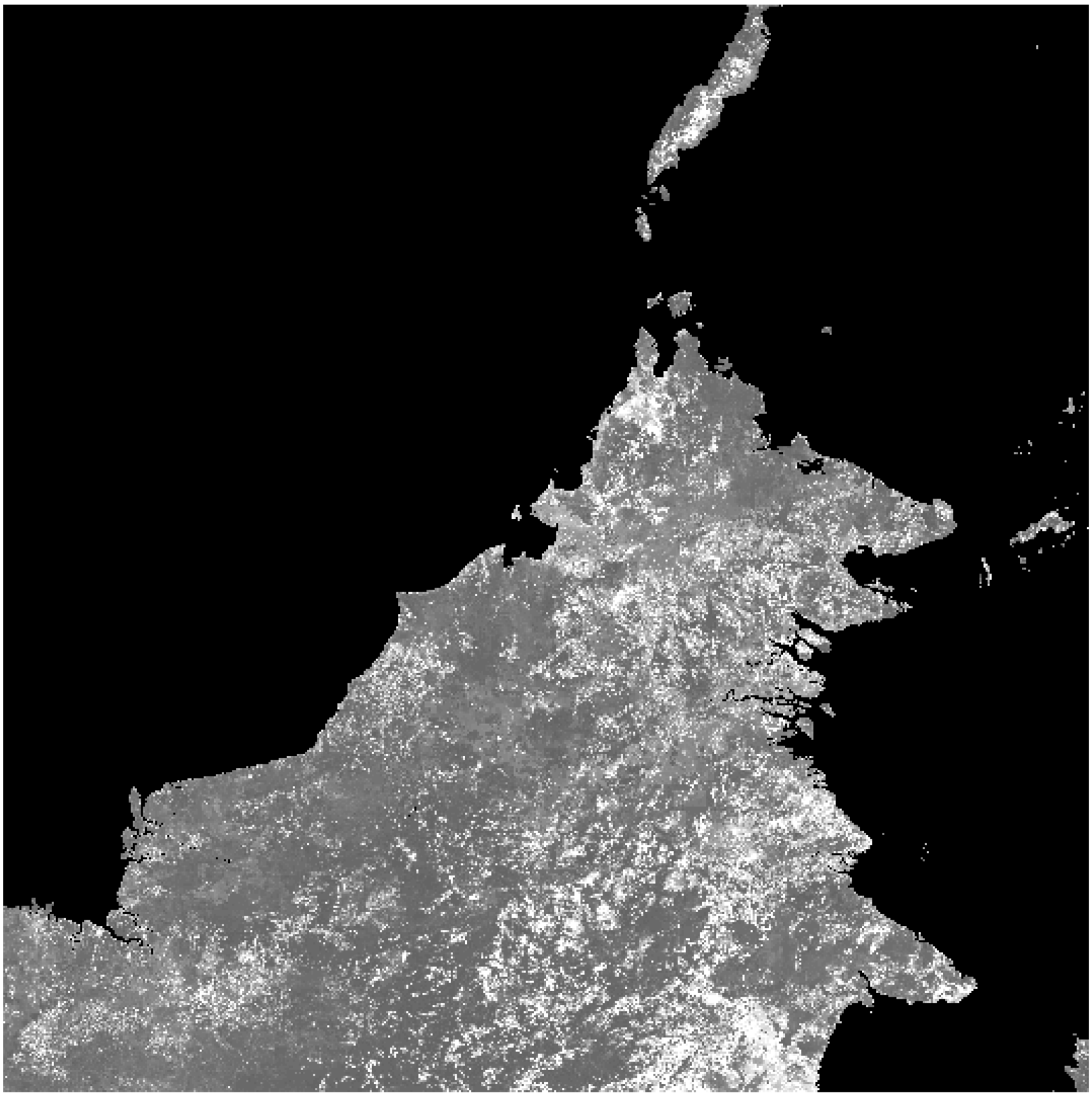

Figure 3: *MODIS Reflectance data*

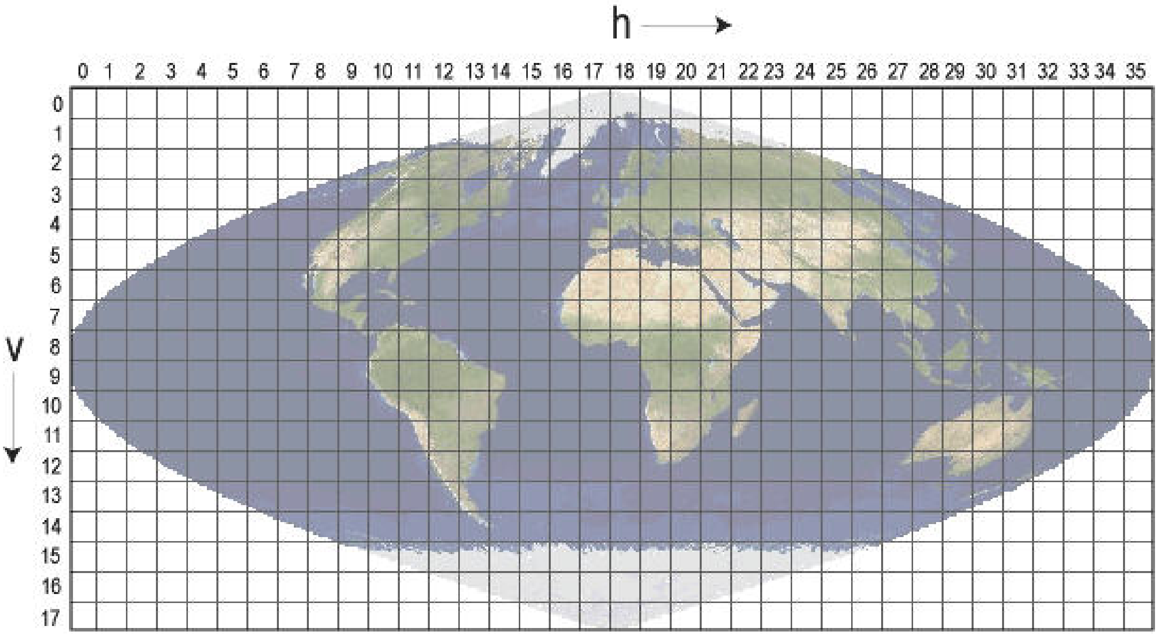

Figure 4: *MODIS Sinusoidal projection and panel labelling*

Obtaining MODIS data

This data is a download from a vast repository of satellite data products. You need to register for downloads but the data is available here:

`https://lpdaac.usgs.gov/`

It isn't necessarily easy to find the data you want though. For example, for MODIS data, this page describes the file naming convention and the different panes of GIS data that cover the globe (Figure 4). In some cases, you may literally have to search through online lists of filenames for what you are after. This, for instance is what you get when you look at the MODIS data pool:

`https://lpdaac.usgs.gov/data_access/data_pool` `http://e4ftl01.cr.usgs.gov/MOLT/`

You will also often find that you need to do battle with obscure file formats. MODIS data comes in Heirarchical Data Format (.hdf) files, which QGIS will open but are not easy to handle. I have used a JAVA tool provided by LPDAAC to extract the data to more friendly TIFF files.

`https://lpdaac.usgs.gov/tools/modis_reprojection_tool`

The SAFE experimental layout is currently looking very distorted - you can't tell but it is! We will want to reproject our satellite data into UTM so we can match it up neatly. So:

- Open the raster reprojection tool: Raster ▷ Projections ▷ Warp.
- In the dialog box, select the blue reflectance layer as the input layer and set the output to Modis_blue_reflectance_UTM50N.tif.
- Now set the Target SRS (spatial reference system): type EPSG:32650 into the filter box and it should bring up UTM50N. This is a reference code to a great resource: a global database of spatial reference systems and their definitions (`http://spatialreference.org/`.
- Click the resampling method checkbox and select 'cubic'. Reprojection creates new cells based on the surrounding data from the source layer - the resampling method selects how that the final value is interpolated from that data: in this case using a cubic polynomial.
- Click OK to run it. You should see the product added to your layers in QGIS and it should look slanted and odd.
- Repeat this process for the other two layers, remembering to change the output names!
- Now remove the original layers and change the project projection to use UTM50N. If you bring the SAFE layout back up to the top, you should now see something that looks like Figure 5: this is how the SAFE project is laid out on the ground.
- Right click on the SAFE layout layer name and select Open Attribute Table: this table shows what the different components of the SAFE experiment are. You can highlight rows in the table to highlight the different polygons in the map.

The gdalwarp tool

At the bottom of the dialog box, you'll see the text of a shell command to run the underlying `gdalwarp` command. This is a powerful tool for reprojecting rasters that actually has far more functionality than QGIS exposes: try entering `gdalwarp -h` at the command line to see more options.

These layers now show us the reflectance recorded by MODIS in three wavelengths: blue, red and near infrared (NIR). These are the bands used to calculate NDVI and EVI. First, we'll visualise the data as a false colour composite. To do this, we'll create a virtual raster file: this is a small text file that behaves like a multilayer raster but only really links to the existing files. In most cases, it is far better to do this than to make a real raster with all three bands and duplicate all the data:

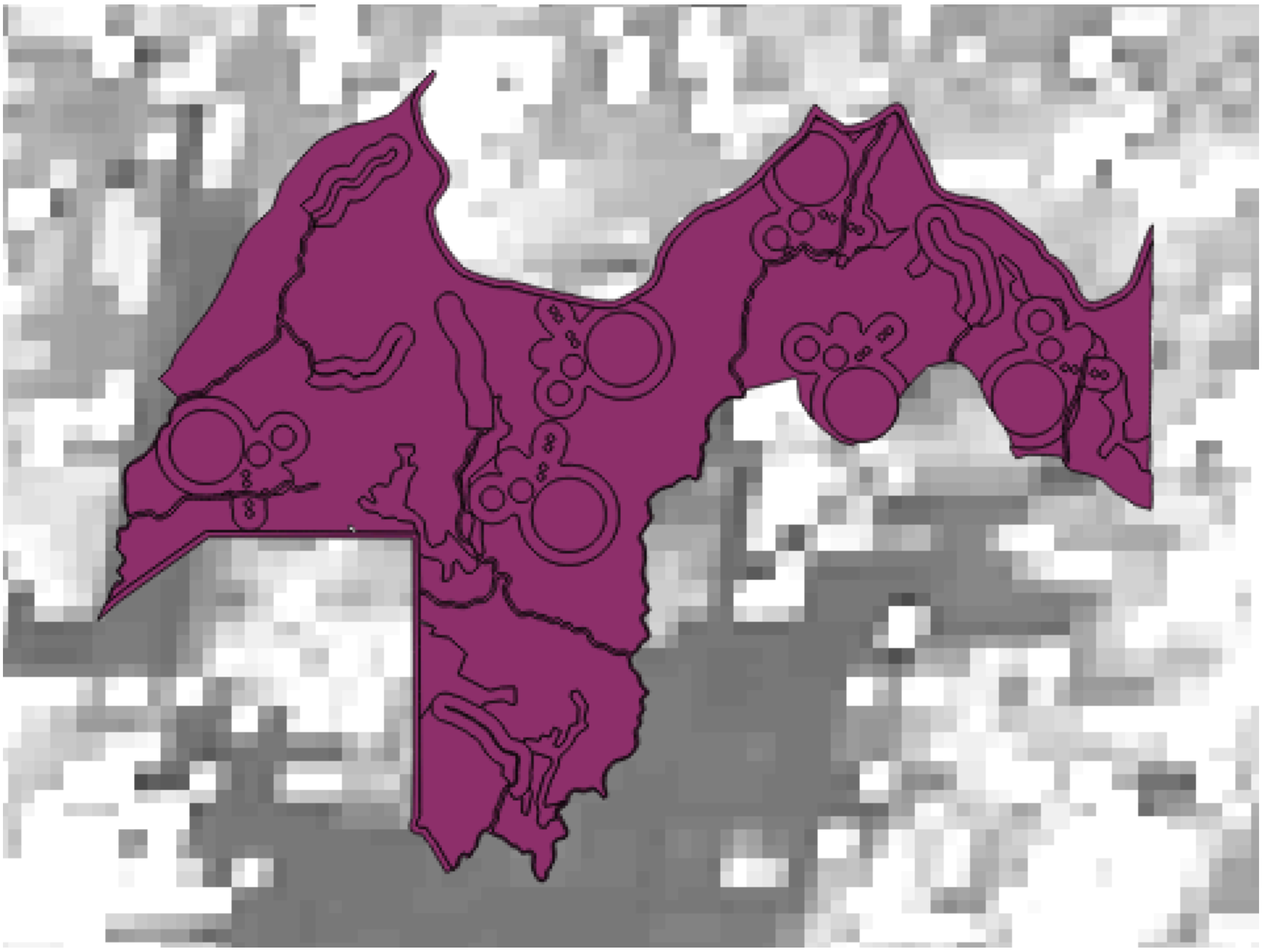

Figure 5: *SAFE layout in UTM 50N projection with MODIS reflectance data behind.*

- Choose the VRT tool: Raster ▷ Miscellaneous ▷ Build Virtual Raster.
- In the inputs selection, choose all three of the UTM 50N MODIS rasters. In the output, select a file in the Borneo folder called 'MODIS_reflectance.vrt'.
- Check the box marked 'Separate', otherwise you'll get an average across the layers and then click OK.
- You should now see a coloured image appear in your GIS window.
- If you right click on the new layer and choose properties and then the Style tab, you can change which bands are displayed using red, blue and green to give the false colour. The bands aren't labelled but were loaded in alphabetic order, so band 1 is blue, band 2 is NIR and band 3 is red.

The final thing we're going to do with this data is to use the Raster Calculator. This tool takes a set of aligned raster layers and creates a new layer with cell values calculated from the old ones. There are two things we can calculate easily from this reflectance data:

- The Normalised Difference Vegetation (NDVI):

$$\text{NDVI} = \frac{(\text{NIR} - \text{RED})}{(\text{NIR} + \text{RED})}$$

- The Enhanced Vegetation Index (EVI):

$$EVI = G \times \frac{(\text{NIR} - \text{RED})}{(\text{NIR} + C_1 \times \text{RED} - C_2 \times \text{Blue} + L)}$$

  Note that EVI has some constants in it that need to be set: for MODIS data, these are usually $G = 2.5, C_1 = 6.0, C_2 = 7.5$ and $L = 1$.

- Open the Raster Calculator tool: Raster ▷ Raster Calculator. You will see a list of the raster layers you have open on the left. Note that if a layer contains multiple bands (like our VRT layer), then each will be visible: `MODIS_VRT@1` is the first band in the virtual raster dataset.
- Enter the calculation for NDVI. You can click on the band names to insert them into the calculation box. It should look like this:

```
("MODIS_NIR_UTM50N@1" - "MODIS_red_UTM50N@1")/
("MODIS_NIR_UTM50N@1" + "MODIS_red_UTM50N@1")
```

- Set the output to create a TIF called MODIS_NDVI.tif in the Borneo folder and click go.
- Have a go with the EVI calculation!

The gdal_calc.py tool

Like `gdalwarp`, this operation can be achieved using the `gdal_calc.py` command from the shell. The GUI doesn't give you the code, but as an example, NDVI would be:

```
gdal_calc.py -A MODIS_NIR_UTM50N.tif -B MODIS_Red_UTM50N.tif \
             --outfile=MODIS_NDVI.tif --calc="(A-B)/(A+B)"+
```

Merging rasters, format conversion and zonal statistics

The grand aim in this section is to get a summary of the mean annual temperature and total precipitation within land cover classes in the UK.

To do this, we'll use the data in the EU folder in the practical files, which contains 5 raster datasets. Four of these are Bioclim data – the Bioclim variables are a suite of 19 climatic variables derived from monthly datasets of minimum and maximum temperature and of precipitation. The idea behind these variables is that they capture biologically more meaningful aspects of climate than a time series of monthly values. Variable 1 is annual mean temperature, variable 12 is total annual precipitation and the full suite of variables is described here:

`http://www.worldclim.org/bioclim`

Along with the temperature data you saw in the first session, these are 'current conditions' files (1950 - 2000 averages from interpolated weather station data) and were downloaded from the Worldclim website above. The rasters in the Global folder are at a 10 arc-minute resolution (60 arc-minutes to a degree, so $^1/_6$ th of a degree); the files in the EU folder are at 30 arc-second resolution (60 arc-seconds to an arc-minute, so $^1/_{120}$ th of a degree). High resolution files are very large, so the data here are two panels that cover the UK and some of the EU.

Remember that a degree is a measurement of angle not distance and that the length of a degree of longitude varies with latitude: think of the width of an orange segment. At the equator, an arc-minute of longitude is $^1/_{21600}$ th ($360° \times 60$) of the circumference of the Earth (radius = 6378137 m):

$$\frac{2 \times \pi \times 6378137.0}{360 \times 60} = 1855.325 \text{ metres} = 1 \text{ nautical mile}$$

In the UK (average latitude $\approx 54.0°$), an arc-minute of longitude is $\cos(54.0°)$ of this value, which is about 1090 metres. So, the raster cells in these images are roughly speaking 545 metres wide – but this varies across the image. Measurements of *latitude* are the same everywhere (excluding a tiny bit of flattening of the Earth), so raster grid cells in degrees aren't always square in other projections.

- In a new QGIS project, load the Bio1 layer and then add the Bio1_15 and Bio1_16 layers from the EU Folder. All files are in a WGS84 geographic coordinate system, so should appear as neat rectangles. Zoom in on the UK and hide the layers to see the difference in resolution.
- Remove the global layer and change to the British National Grid coordinate system. If you zoom out to see all of the data, you'll see it is now curved: the data are more compressed at the top, where the degree of longitude is shorter.

The last data set ('g250_06') is the 2006 CORINE land cover dataset for the EU: it contains land cover classes, so the integer values in the dataset are category codes. The 'clc_legend_qgis.txt' file contains a summary of the land cover classes and codes used in the map. This dataset is in an equal area projection ('Lambert Equal Area'), centred on Northern Germany in order to get a good, large scale coverage of the whole of the EU and is at a resolution of 250 metres. The dataset is freely available from here:

`http://www.eea.europa.eu/data-and-maps/data/corine-land-cover-2006-raster-3`

- In another new QGIS project, load the g250_06 layer and have a look at it.

So – given this data – how do we go about creating a dataset for the UK and Ireland where we can compare climatic conditions in different land cover classes? The datasets are at two different resolutions and in two different projections – the raster cell boundaries from each can be seen in Figure 6a.

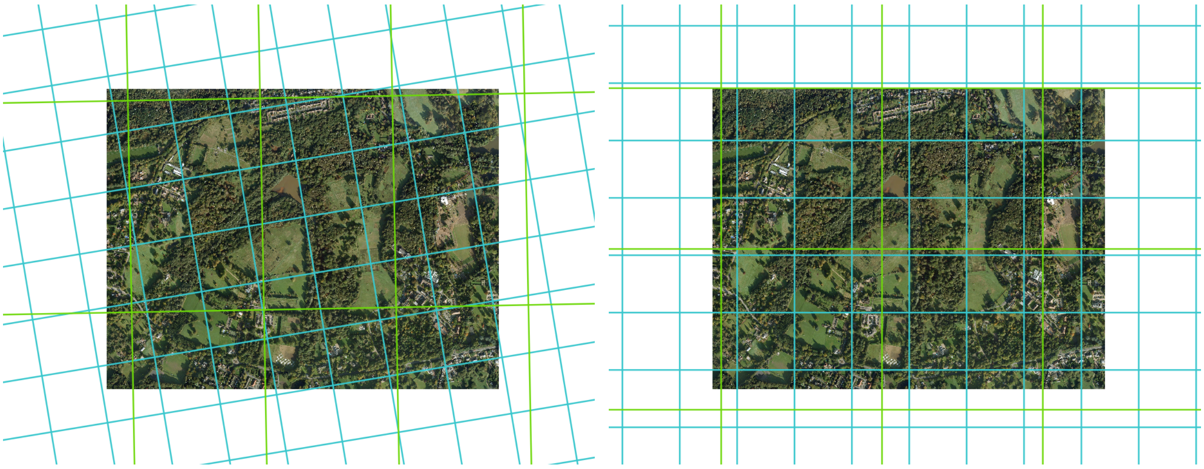

Figure 6: *Silwood Aerial photo, projected using the British National Grid, showing raster cell boundaries from a 30 arc-second resolution Bioclim layer (green) and the 250 metre resolution CORINE dataset (blue): (a) [left] shows the orientation of cell boundaries in their original projections; b) shows the orientation of reprojected cells without controlling the origin and resolution.*

We need to do three kinds of operations to do this: *merge* the two climate panels, *clip* the datasets to the region of interest, and then *reproject* the data from the original projections to our new projection. The reprojection step is the most computer intensive, so it makes sense to do this as the final step on the smaller subsets.

- With two different projections to handle, we need to be careful about reprojecting data in the QGIS display, so we'll handle the two data sources in their own projections rather than trying to view everything simultaneously.
- In a new project, load the 'bio1_15.bil' and 'bio1_16.bil' layers. Run the Merge tool (Raster ▹ Miscellaneous ▹ Merge) using the two Bioclim 'bio1' layers as inputs and 'bio1_merge.tif' as the output name. You will need to set QGIS to use -32768 as the no data value.
- Now run the Clipper tool (Raster ▹ Extraction ▹ Clipper) using 'bio1_merge.tif' as the input layer, 'bio1_UK.tif' as the output layer and the extent coordinates ($x_1 = -12, y_1 = 60$) and ($x_2 = 4, y_2 = 48$) – these are latitude and longitude values.
- Start a new project and load the 'g250_06.tif' layer. Run the Clipper tool again using 'g250_06.tif' as the input layer, 'g250_06_UK.tif' as the output layer. This time the extent coordinates are in the European projected coordinate system: use (2800000, 4200000) and (4000000, 3000000) as the extent.
- Add the 'bio1_UK.tif' layer – it should be reprojected automatically – and you'll see that we've cut out two rough blocks covering the area of interest.

Now we need to reproject these layers using the Warp tool. This is a slightly trickier problem than in the last section: as Figure 6a shows the values in the raster cells are a different resolutions and orientations so the warping process needs to establish a new grid and then assign sensible values given the local data in the original projection.

In order to get the finest control, we will have to add some options to the Warp tool. This is easy: the drop down menus set up the command (you'll see the text of the command at the bottom of the tool) but we can click on the pencil icon to edit the commands before running it. We need to know two extra options: `-tr xres yres`, which sets the *target resolution*, and `-te xmin ymin xmax ymax`, which sets the *target extent*. Together, we can use this extra info to force both maps to align neatly using a 2 kilometre resolution. Without these extra steps, QGIS would try and keep a similar resolution to the original files and the grid origins would differ, leading to a bit of a mess (see Figure 6b).

- Open the Warp tool (Raster ▹ Projections ▹ Warp) and set the following: 'bio1_UK.tif' is the input file, 'bio1_UK_BNG.tif' is the output file, the target SRS is the British National Grid (EPSG:27700), the no data value is -32768 and the resampling method is cubic.
- You'll see the command line instructions appear in the box at the bottom. Now click on the pencil button and add the following text at the end, leaving a space after the last file name:

  ```
  -tr 2000 2000 -te -220000 -10000 680000 1080000
  ```

- Now click OK to run the reprojection.
- Repeat this process for the 'g250_06_UK.tif' file, saving it to 'g250_06_UK_BNG.tif' and using the Near resampling method – this will assign new cell values from the nearest land cover class from the original data. You do not need to set a no data value for this file.

We now have the data aligned and in the same projection. We now need to extract values for the environmental conditions using a process called Zonal Statistics. This takes raster data and summarises the values found with a set of polygon zones, so we will first need to create a shapefile of polygons with one polygon for each land cover class.

- Use the Polygonize tool (Raster ▹ Conversion ▹ Polygonize) to create a polygon version (CORINE_2K.shp) of the 2 km CORINE land cover data that we just created. This might take a little while to run.
- If you look at the attribute table for this shapefile, you will see it is made up of thousands of small polygons. We want to group these so that all the small polygons for each class are in a single feature. Use the Vector ▹ Geometry Tools ▹ Singleparts to Multipart tool to create this file (CORINE_2K_Multi.shp).
- If it isn't already available (Raster ▹ Zonal Statistics ▹ Zonal Statistics), enable the Zonal Statistics plugin (Plugins ▹ Manage and Install plugins …).
- Open the Zonal Statistics tool and set the bio1_UK_BNG layer.tif as the raster layer, CORINE_2K_Multi.shp as the zone layer and 'bio1_' as the column prefix. Run the tool – again this could take a little while to run.
- If you look at the attribute table for CORINE_2K_Multi.shp, new columns have been added to give the number, sum and mean values of the temperature data within each zone.
- If you have time, try and add the precipitation data (the two 'bio12' panels) to this shapefile!

CMEE coursework

Your coursework for this week is to create a Python script that automates the whole of this section. It should start with only the five original EU input files and create a comma separated variable file of mean climatic values for both bio1 and bio12 within each land cover class. The output file should be called 'zonalstats.csv' and start with the header row 'LCC,bio1,bio12' and then have an integer class code and means to two decimal places for each class (e.g. 43,94.54,961.38). You should use relative pathnames – the script can expect to find the input files in the directory from which it is run.

The first part of this process is easiest to implement using the GDAL tools directly from the command line, so you can copy the commands from the QGIS tools and embed them in `os.system()` (or equivalent) commands. The zonal statistics tools do not have a command line alternative – they are using the QGIS system internally. So, you will need to reinvent the process within Python! Some hints though:

- Rasters are just two dimensional arrays and the numpy module has some powerful handling commands for those!
- The gdal package for Python provides a huge amount of functionality for bringing data into Python. See here for some details: `http://www.gdal.org/gdal_tutorial.html`. Specifically, you are going to want to get an array of raster data and any no data values from the file – the basic recipe is:

```
import gdal

# load the data into an array
layer = gdal.Open('myLayer.tif')
data = layer.ReadAsArray()

# get the no data value
band = layer.GetRasterBand(1)
noDataVal = band.GetNoDataValue()
```

- You're going to need to extract values from one array (the bioclim variables) where a value in another array (the CORINE data) matches a particular land cover class code. A simple way to do this (using some dummy example data) is:

```
import numpy

data = numpy.array(range(0,20))
data.shape = [4,5]

codeVals = [1,2,3]
codes = numpy.random.choice(codeVals, size=20)
codes.shape = [4,5]

for eachCode in codeVals:
    locations = numpy.where(codes == eachCode)
    vals = data[locations]
    print vals.mean()
```

Good luck!