WoSIS tutorial
Accessing soil profile data from QGIS and R

WoSIS (World Soil Information Service)[1] is ISRIC's data service, primarily for soil profile and soil samples data. WoSIS is a PostgreSQL DB with multiple data services running on top. The system currently serves about 80,000 soil profiles (with coordinates) covering some 60+ countries. For more info about WoSIS please refer to the **official documentation**.

Software installation

To access soil profile (point) data you can either use **QGIS** or similar GIS software that supports **Web Feature Service**. This document explains primarily how to access WoSIS points directly from R, but this assumes that you have already installed:

- **R** [http://cran.r-project.org/bin/windows/base/];
- **R packages**: plotKML, rgdal, aqp
- **Google Earth** or **Google Earth Pro** (grab a free key for GEP);
- **GDAL** [http://www.gisinternals.com/release.php] for Windows machines use e.g. "gdal-111-1800-x64-core.msi";
- **QGIS** (optional)

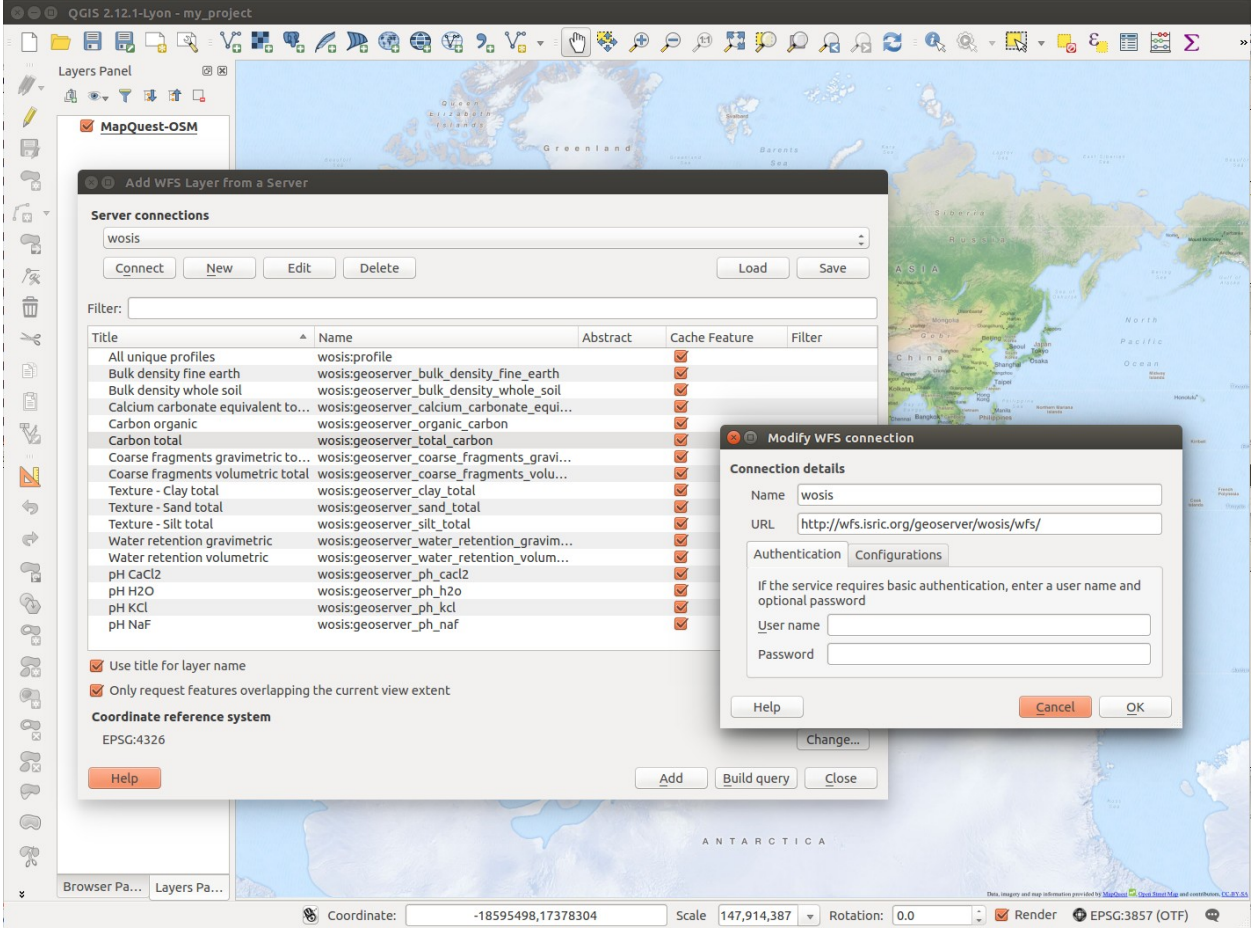
Look **here** to find out more about how to install R, GDAL and similar.

Accessing WoSIS from QGIS

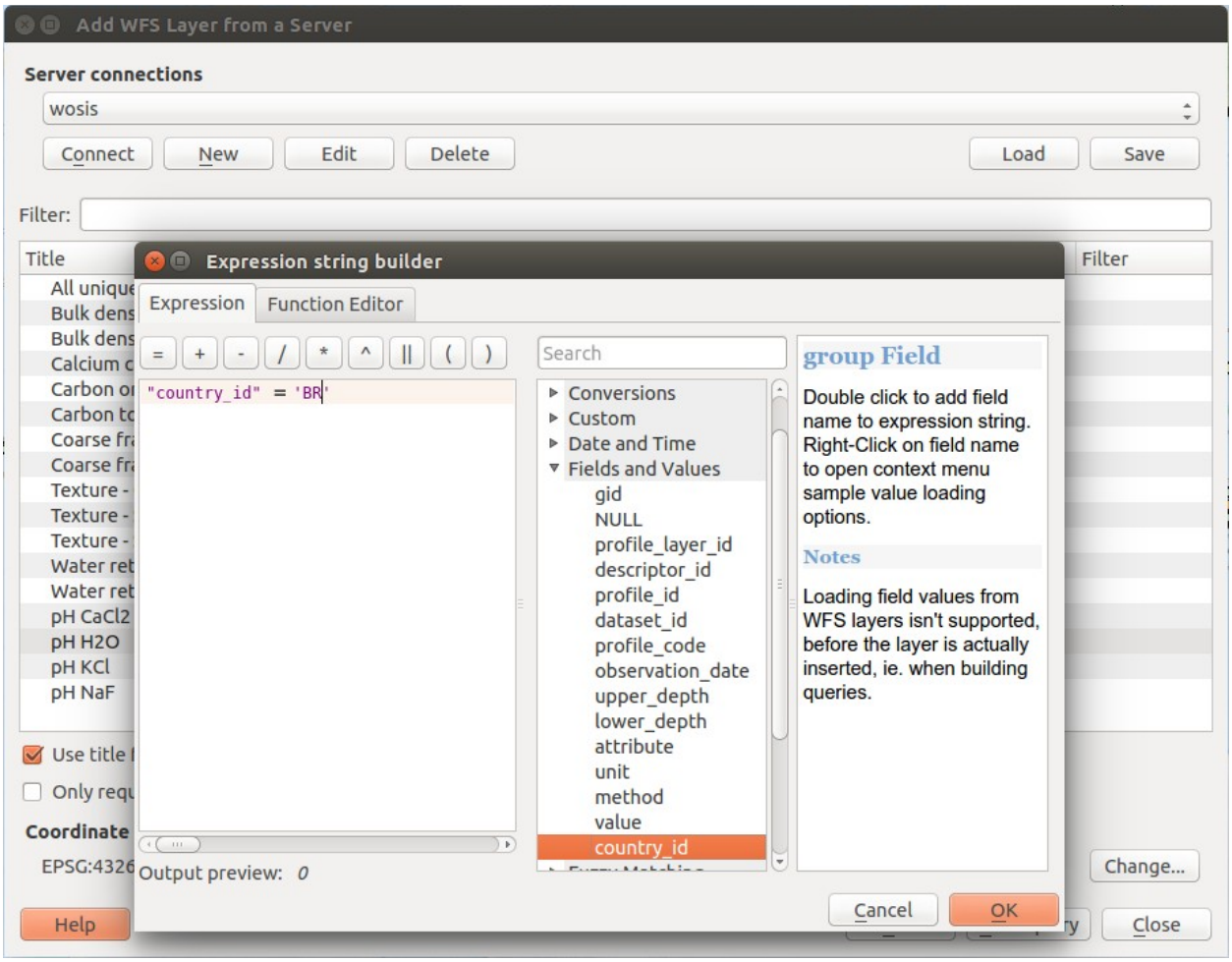
To open and view WoSIS points in QGIS, you need first to create a server WFS connection. To do this press the Add WFS layer button  Then press 'New', give it a connection name

'WoSIS' for example, in the URL field put 'http://wfs.isric.org/geoserver/wosis/wfs/', then press OK button. Press 'Connect' and all available layers will appear.

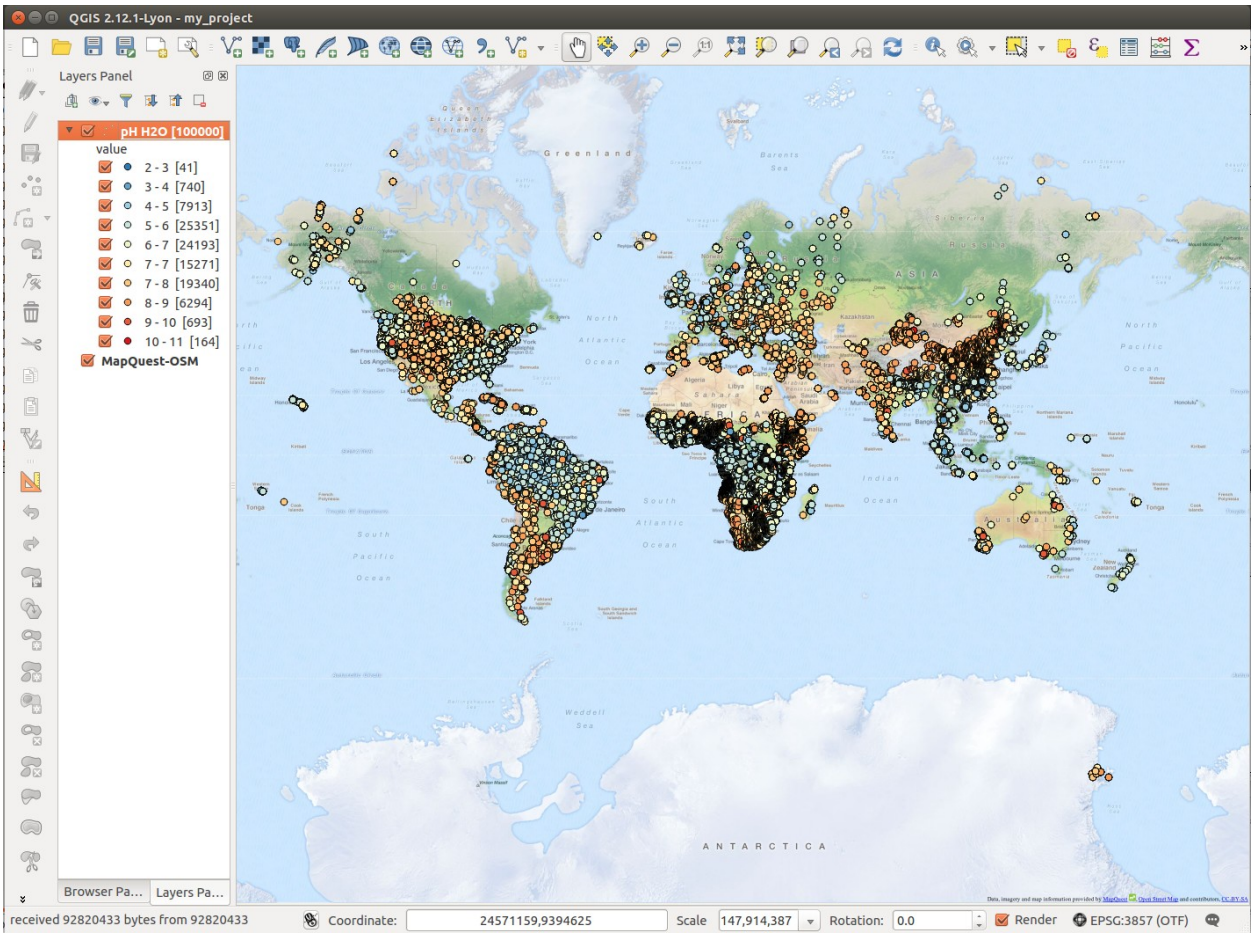
[1] http://www.isric.org/data/wosis



Next, select the layer of interest listed under "Title" and press 'Add'. Note, it might take even few minutes until you fetch all points for the global coverage (important: the maximum number of points that it can fetch is set to 100,000). So possibly a more efficient approach, if you require only a selection of points, is to first zoom to the area of interest and then tick on “Only request features overlapping the current view extent”. Another option, when adding the layer, is to double click on ‘Filter’ column, expand ‘Fields and Values’ and, for example, build the expression “country_id = ‘BR’ ”.



After you fetch point data, you can view it in QGIS, best overlaid over the country borders (you can download those from e.g. NaturalEarth website or install the 'OpenLayers plugin' under 'Plugins' menu). The global coverage map will look something like this:



Accessing WoSIS from R

A more flexible way to access WoSIS points is to use the GDAL functionality (see WFS driver documentation for GDAL). For this we can use OGR functions **ogrinfo** and **ogr2ogr** (basically a translation function). Before you can start, make sure you first install GDAL binaries (do not come with rgdal). Under Windows OS, this requires first locating GDAL executables. If GDAL is installed and path known, we can request information about the data on the server by using the ogrinfo:

```
gdal.dir <- shortPathName("C:/Program Files/GDAL")
ogr2ogr <- paste0(gdal.dir, "/ogr2ogr.exe")
ogrinfo <- paste0(gdal.dir, "/ogrinfo.exe")
system(paste(ogrinfo, '-ro WFS:\"http://wfs.isric.org/geoserver/wosis/wfs\"'))
INFO: Open of `WFS:http://wfs.isric.org/geoserver/wosis/wfs'
      using driver `WFS' successful.
1: wosis:profile (Point)
2: wosis:bulk_density_fine_earth (Point)
3: wosis:bulk_density_whole_soil (Point)
4: wosis:calcium_carbonate_equivalent_total (Point)
5: wosis:organic_carbon (Point)
6: wosis:total_carbon (Point)
7: wosis:coarse_fragments_gravimetric_total (Point)
8: wosis:coarse_fragments_volumetric_total (Point)
```

```
9: wosis:clay_total (Point)
10: wosis:sand_total (Point)
11: wosis:silt_total (Point)
12: wosis:water_retention_gravimetric (Point)
13: wosis:water_retention_volumetric (Point)
14: wosis:ph_cacl2 (Point)
15: wosis:ph_h2o (Point)
16: wosis:ph_kcl (Point)
17: wosis:ph_naf (Point)
```

This gives a list of layers currently available via WoSIS. For more info about the code names please refer to the **official documentation**.

Visualization of WoSIS data in Google Earth

Next, we would like to import points from WoSIS into R and then use them for analysis or visualize them in Google Earth. We can fetch only a subset of points i.e. clay content for a bounding box of 10 by 5 degrees (France) directly via the ogr2ogr:

```
system(paste(ogr2ogr, '-f \"ESRI Shapefile\" clay_total_sub.shp
WFS:\"http://wfs.isric.org/geoserver/wosis/wfs" clay_total -clipsrc 0 45 10 50'))
Warning 6: Normalized/laundered field name: 'profile_layer_id' to 'profile_la'
Warning 6: Normalized/laundered field name: 'descriptor_id' to 'descriptor'
Warning 6: Normalized/laundered field name: 'profile_code' to 'profile_co'
Warning 6: Normalized/laundered field name: 'observation_date' to 'observatio'
Warning 6: Normalized/laundered field name: 'upper_depth' to 'upper_dept'
Warning 6: Normalized/laundered field name: 'lower_depth' to 'lower_dept'
```

Other conversion possibilities using the WFS driver are explained **here**. The output of the operation above will fetch only few hundred points, which we can now import to R via rgdal package:

```
library(rgdal)
clay_total_sub <- readOGR("clay_total_sub.shp", "clay_total_sub")
OGR data source with driver: ESRI Shapefile
Source: "clay_total_sub.shp", layer: "clay_total_sub"
with 406 features
It has 15 fields
```

Note that these are in fact 3D points as they refer to different sampling depths (see: upper_dept, and lower_depth columns). We can visualize the points by using the plotKML package:

```
library(plotKML)
shape = "http://maps.google.com/mapfiles/kml/pal2/icon18.png"
kml(clay_total_sub, colour=value, shape=shape, points_names=clay_total_sub$value,
balloon=TRUE)
KML file opened for writing...
Writing to KML...
Closing  clay_total_sub.kml
kml_View("clay_total_sub.kml")
```

Probably more accurate thing to do with this data is to visualize it as 3D points, which is possible by adding the 3rd dimension into the kml function:

```
clay_total_sub$depth <- clay_total_sub$upper_dept + (clay_total_sub$lower_dept - clay_total_sub$upper_dept)/2
kml(clay_total_sub, file="clay_total_sub3D.kml", colour=value, shape=shape, points_names=clay_total_sub$value, balloon=TRUE, altitude=300-depth)
KML file opened for writing...
Writing to KML...
Closing  clay_total_sub3D.kml
kml_View("clay_total_sub3D.kml")
```

To further explore possibilities of processing and visualizing soil profile data, consider using the **aqp** package, which will allow you to produce soil depth plots such as the one shown in this gallery.