# Accessing WoSIS from R – 'Latest' Version

D G Rossiter

04-July-2022

## Table of Contents

## Introduction

This document shows how to access WoSIS "Latest" data from R. For access to WoSIS "Snapshot" data from R, see `WoSIS_Snapshot_with_R.Rmd` at https://git.wur.nl/Batje001/wosis/-/tree/master/R_scripts.

The "Latest" dynamic dataset contains the most recent version of standardised soil data served from WoSIS. Being dynamic, the dataset will grow once new point data are standardised, additional soil properties are considered, and/or when possible amendments are required.

For an overview of WoSIS, see https://www.isric.org/explore/wosis. This links to https://www.isric.org/explore/wosis/accessing-wosis-derived-datasets which explains the difference between snapshot and dynamic datasets, and how to access them.

The Procedures Manual describes how the WoSIS database is built.

## Packages

If you do not have these on your system, install with `install.packages(..., dependencies=TRUE)` or via the R Studio package manager.

The `gdalUtilities` package replaces the obsolete `gdalUtils` package, and as of end June 2022 must be installed from its author's github repository, using the `instll_github` function of the `devtools` package.

However, some functions (notably, `ogrinfo`) are not implemented there, and so `gdalUtils` must be used for these, and therefore your system must have a GDAL installation.

In a future version of these procedures we will find a way to replace `ogrinfo` with currently-supported packages.

```
# library(devtools)
# devtools::install_github("JoshOBrien/gdalUtilities")
library(gdalUtilities)      # wrappers for GDAL utility programs that could
be
                        #  called from the command line, but here via `sf`
# devtools::install_github("gearslaboratory/gdalUtils")
library(gdalUtils)      # wrappers for GDAL utility programs that could be

##
## Attaching package: 'gdalUtils'

## The following objects are masked from 'package:gdalUtilities':
##
##     gdal_grid, gdal_rasterize, gdal_translate, gdalbuildvrt, gdaldem,
##     gdalinfo, gdalwarp, nearblack, ogr2ogr

                        #  called from the command line,
library(sf)             # spatial data types -- Simple Features
```

```
## Linking to GEOS 3.10.2, GDAL 3.4.2, PROJ 8.2.1; sf_use_s2() is TRUE

##
## Attaching package: 'sf'

## The following object is masked from 'package:gdalUtils':
##
##     gdal_rasterize

## The following object is masked from 'package:gdalUtilities':
##
##     gdal_rasterize

library(stars)          # Spatiotemporal Arrays, Raster and Vector Data Cubes

## Loading required package: abind

library(dplyr)          # tidyverse data manipulation functions

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(ggplot2)        # gpplot graphics
library(maps)           # optional -- for boundary polygons
library(mapdata)
```

The `stars` and `sf` packages have built-in support for GDAL, the Geographic Data Abstraction Library. This is used to read, write and convert between formats. The `gdalUtilities` package provides direct access to GDAL via `sf::gdal_utils`.

Check the drivers supported by your installation, and their capabilities, for the three we will use: the OGC GeoPackages `GPKG`, the ESRI shapefile `ESRI`, and the CSV flat text file format `CSV`:

```
drivers <- sf::st_drivers()
# print(drivers)
ix <- grep("GPKG", drivers$name,  fixed=TRUE)
drivers[ix,]

##      name  long_name write copy is_raster is_vector  vsi
## GPKG GPKG GeoPackage  TRUE TRUE      TRUE      TRUE TRUE

ix <- grep("ESRI", drivers$name,  fixed=TRUE)
drivers[ix,]
```

```
##                          name          long_name write  copy is_raster
## ESRIC                   ESRIC Esri Compact Cache FALSE FALSE      TRUE
## ESRI Shapefile ESRI Shapefile     ESRI Shapefile  TRUE FALSE     FALSE
## ESRIJSON             ESRIJSON           ESRIJSON FALSE FALSE     FALSE
##                is_vector  vsi
## ESRIC               TRUE TRUE
## ESRI Shapefile      TRUE TRUE
## ESRIJSON            TRUE TRUE
```

```
ix <- grep("CSV", drivers$name,  fixed=TRUE)
drivers[ix,]
```

```
##     name                    long_name write  copy is_raster is_vector  vsi
## CSV  CSV Comma Separated Value (.csv)  TRUE FALSE     FALSE      TRUE TRUE
```

## WoSIS Web Feature Service

The "latest" (dynamic) version of WoSIS is provided via WFS (Web Feature Services). This is described in less technical terms in Wikipedia.

WFS allows you to incorporate geographic data into your own GIS projects, unlike WMS (Web Map Service), which only displays geographic data within a GIS. Here we use R as the GIS, since it can handle both geographic and feature-space information.

### Specify WoSIS WFS and list available layers

Specify the web address of the "latest" version of WoSIS:

```
wfs <- "WFS:https://maps.isric.org/mapserv?map=/map/wosis_latest.map"
```

List the layers in the WFS source with the `sf::st_layers` function.

```
(layers.info <- st_layers(wfs))
```

```
## Driver: WFS
## Available layers:
##                  layer_name geometry_type features fields
## 1    ms:wosis_latest_bdfi33                  78058     15
## 2    ms:wosis_latest_bdfiad                  21033     15
## 3    ms:wosis_latest_bdfifm                  14160     15
## 4    ms:wosis_latest_bdfiod                 122699     15
## 5    ms:wosis_latest_bdws33                 154203     15
## 6    ms:wosis_latest_bdwsod                  75306     15
## 7    ms:wosis_latest_cecph7                 294224     15
## 8    ms:wosis_latest_cecph8                  24118     15
## 9      ms:wosis_latest_cfgr                 202376     15
## 10     ms:wosis_latest_cfvo                 234848     15
## 11     ms:wosis_latest_clay                 630284     15
## 12     ms:wosis_latest_ecec                 132734     15
## 13   ms:wosis_latest_elco20                  44307     15
```

```
## 14   ms:wosis_latest_elco25                15658     15
## 15   ms:wosis_latest_elco50                90914     15
## 16   ms:wosis_latest_elcosp                73311     15
## 17   ms:wosis_latest_nitkjd               216352     15
## 18     ms:wosis_latest_orgc               492109     15
## 19     ms:wosis_latest_phaq               610294     15
## 20     ms:wosis_latest_phca               313668     15
## 21     ms:wosis_latest_phkc               152984     15
## 22     ms:wosis_latest_phnf                25325     15
## 23   ms:wosis_latest_phetb1                40401     15
## 24   ms:wosis_latest_phetm3                 7230     15
## 25   ms:wosis_latest_phetol                 8358     15
## 26   ms:wosis_latest_phprtn                23806     15
## 27   ms:wosis_latest_phptot                13997     15
## 28   ms:wosis_latest_phpwsl                 1241     15
## 29 ms:wosis_latest_profiles               217184     75
## 30     ms:wosis_latest_sand               512802     15
## 31     ms:wosis_latest_silt               596635     15
## 32     ms:wosis_latest_tceq               221850     15
## 33     ms:wosis_latest_totc               132189     15
## 34   ms:wosis_latest_wg0006                 3828     15
## 35   ms:wosis_latest_wg0010                12526     15
## 36   ms:wosis_latest_wg0033                94759     15
## 37   ms:wosis_latest_wg0100                 3360     15
## 38   ms:wosis_latest_wg0200                27780     15
## 39   ms:wosis_latest_wg0500                 1414     15
## 40   ms:wosis_latest_wg1500               182097     15
## 41   ms:wosis_latest_wv0010                 5213     15
## 42   ms:wosis_latest_wv0033                17567     15
## 43   ms:wosis_latest_wv0100                 2553     15
## 44   ms:wosis_latest_wv0500                 1758     15
## 45   ms:wosis_latest_wv1500                17371     15
```

There are 45 layers. Most of the names refer to soil properties in a fairly obvious way: the database name `ms:wosis_latest:wosis_latest_` and then a property name, e.g., `bdfi33`. These names are explained in the on-line documentation. For example `bdfi33` is "Bulk density of the fine earth fraction, equilibrated at 33 kPa." units are mg $kg^{-1}$.

## Display site properties

The layer `"ms:wosis_latest:wosis_latest_profiles"` contains the site information.

The `gdalUtils::ogrinfo` function displays information about a dataset accessible via GDAL. Here we see the metadata for the site information. Note the `so` "summary only" option. This suppresses listing of features, and shows only the summary information. We choose this because there are many profiles in the entire dataset.

The options for `ogrinfo` are listed at the GDAL on-line reference page.

```
profiles.info <-
  gdalUtils::ogrinfo(wfs, layer = "ms:wosis_latest_profiles",
                     ro = TRUE, so = TRUE, q = TRUE)
cat(profiles.info, sep="\n")
```

```
## Metadata:
##   ABSTRACT=Title: wosis_latest
##   PROVIDER_NAME=ISRIC - World Soil Reference
##   TITLE=Title: wosis_latest
##
## Layer name: ms:wosis_latest_profiles
## Metadata:
##   ABSTRACT=All profiles available in WoSIS latest with the soil
Classification according to specified edition (year) of the World Reference
Base for Soil Resources (WRB, up to qualifier level); FAO-Unesco Legend (up
to soil unit level); USDA Soil Taxonomy (up to subgroup level).
##
## ISRIC is developing a centralized and user-focused server database, known
as ISRIC World Soil Information Service (WoSIS). The aims are to:
##
## • Safeguard world soil data "as is"
##
## • Share soil point (i.e. profile) data upon their standardization, and
ultimately harmonization
##
## • Provide quality-assessed, geo-referenced soil data for a growing range
of environmental applications (e.g. SoilGrids250m).
##
## At present, wosis_latest contains standardized data for 217,000 profiles.
The number of measured data for each property varies between profiles and with
depth, generally depending on the purpose of the initial studies. Further, in
most source data sets, there are fewer data for soil physical as opposed to
soil chemical attributes and there are fewer measurements for deeper than for
superficial horizons. Generally, limited quality information is associated
with the various source data. Further information is provided in a
(https://doi.org/10.5194/essd-12-299-2020) and
(https://www.isric.org/sites/default/files/WOSISprocedureManual_2020nov17web.
pdf)
##   KEYWORD_1=Soil
##   KEYWORD_2=Standard
##   KEYWORD_3=Profiles
##   KEYWORD_4=WoSIS
##   KEYWORD_5=WoSIS latest
##   KEYWORD_6=Soil classification
##   TITLE=WoSIS latest - Profiles
```

This shows the metadata of the sites.

The `gdalUtils::ogrinfo` function also allows an optional limitation to a bounding box with the `spat` argument, as a four-element vector (`xmin, ymin, xmax, ymax`). These are the

coördinates in the layer's Coordinate Reference System (CRS), in this case geographic coördinates.

For example, all the profile (site) information from a 2°×2° tile in central Europe.

Note: the `q=FALSE` ("not quiet") option lists the geometry and feature count.

```
central.eu.profiles.info <-
  gdalUtils::ogrinfo(wfs, ro=TRUE, so=TRUE, q=FALSE,
                     layer="ms:wosis_latest_profiles",
                     spat=c(6, 48, 8, 50),
                     verbose = FALSE)
head(central.eu.profiles.info, 8)
```

```
## [1] "INFO: Open of
`WFS:https://maps.isric.org/mapserv?map=/map/wosis_latest.map'"
## [2] "      using driver `WFS' successful."
## [3] "Metadata:"
## [4] "  ABSTRACT=Title: wosis_latest"
## [5] "  PROVIDER_NAME=ISRIC - World Soil Reference"
## [6] "  TITLE=Title: wosis_latest"
## [7] ""
## [8] "Layer name: ms:wosis_latest_profiles"
```

Show the number of features, the spatial extent, and the Coordinate Reference System (CRS):

```
ix.f <- grep("Feature Count", central.eu.profiles.info)
central.eu.profiles.info[ix.f]
```

```
## [1] "Feature Count: 224"
```

```
ix.e <- grep("Extent", central.eu.profiles.info)
central.eu.profiles.info[ix.e]
```

```
## [1] "Extent: (6.052500, 48.074444) - (7.966667, 50.000000)"
```

```
ix.g <- grep("GEOGCRS", central.eu.profiles.info)
cat(paste(central.eu.profiles.info[ix.g:(ix.g+17)], collapse="\n"))
```

```
## GEOGCRS["WGS 84",
##     DATUM["World Geodetic System 1984",
##         ELLIPSOID["WGS 84",6378137,298.257223563,
##             LENGTHUNIT["metre",1]]],
##     PRIMEM["Greenwich",0,
##         ANGLEUNIT["degree",0.0174532925199433]],
##     CS[ellipsoidal,2],
##         AXIS["geodetic latitude (Lat)",north,
##             ORDER[1],
##             ANGLEUNIT["degree",0.0174532925199433]],
##         AXIS["geodetic longitude (Lon)",east,
##             ORDER[2],
```

```
##             ANGLEUNIT["degree",0.0174532925199433]],
##     USAGE[
##         SCOPE["unknown"],
##         AREA["World"],
##         BBOX[-90,-180,90,180]],
##     ID["EPSG",4326]]
```

Here we see the number of profiles in this tile, the extent (a bit smaller than the bounding box we requested), and the CRS formatted as "Well-Known Text". The CRS corresponds to EPSG code `4326`, i.e., geographic coördinates on the WGS84 datum.

Show the data field names and their data types. These are listed after the `Geometry Column` line of the metadata:

```
ix.p <- grep("Geometry Column", central.eu.profiles.info)
n <- length(central.eu.profiles.info)
central.eu.profiles.info[ix.p+1:n]
```

```
##   [1] "gml_id: String (0.0) NOT NULL"
##   [2] "profile_id: Integer (0.0)"
##   [3] "dataset_id: String (0.0)"
##   [4] "continent: String (0.0)"
##   [5] "country_id: String (0.0)"
##   [6] "country_name: String (0.0)"
##   [7] "geom_accuracy: Real (0.0)"
##   [8] "latitude: Real (0.0)"
##   [9] "longitude: Real (0.0)"
##  [10] "dsds: Integer (0.0)"
##  [11] "cfao_version: Integer (0.0)"
##  [12] "cfao_major_group_code: String (0.0)"
##  [13] "cfao_major_group: String (0.0)"
##  [14] "cfao_soil_unit_code: String (0.0)"
##  [15] "cfao_soil_unit: String (0.0)"
##  [16] "cwrb_version: Integer (0.0)"
##  [17] "cwrb_reference_soil_group_code: String (0.0)"
##  [18] "cwrb_reference_soil_group: String (0.0)"
##  [19] "cwrb_prefix_qualifier: String (0.0)"
##  [20] "cwrb_suffix_qualifier: String (0.0)"
##  [21] "cstx_version: Integer (0.0)"
##  [22] "cstx_order_name: String (0.0)"
##  [23] "cstx_suborder: String (0.0)"
##  [24] "cstx_great_group: String (0.0)"
##  [25] "cstx_subgroup: String (0.0)"
##  [26] "bdfi33: Integer (0.0)"
##  [27] "bdfiad: Integer (0.0)"
##  [28] "bdfifm: Integer (0.0)"
##  [29] "bdfiod: Integer (0.0)"
##  [30] "bdws33: Integer (0.0)"
##  [31] "bdwsad: Integer (0.0)"
##  [32] "bdwsfm: Integer (0.0)"
##  [33] "bdwsod: Integer (0.0)"
```

```
##  [34] "cecph7: Integer (0.0)"
##  [35] "cecph8: Integer (0.0)"
##  [36] "cfgr: Integer (0.0)"
##  [37] "cfvo: Integer (0.0)"
##  [38] "clay: Integer (0.0)"
##  [39] "ecec: Integer (0.0)"
##  [40] "elco20: Integer (0.0)"
##  [41] "elco25: Integer (0.0)"
##  [42] "elco50: Integer (0.0)"
##  [43] "elcosp: Integer (0.0)"
##  [44] "nitkjd: Integer (0.0)"
##  [45] "orgc: Integer (0.0)"
##  [46] "orgm: Integer (0.0)"
##  [47] "phaq: Integer (0.0)"
##  [48] "phba: Integer (0.0)"
##  [49] "phca: Integer (0.0)"
##  [50] "phetb1: Integer (0.0)"
##  [51] "phetm3: Integer (0.0)"
##  [52] "phetol: Integer (0.0)"
##  [53] "phkc: Integer (0.0)"
##  [54] "phnf: Integer (0.0)"
##  [55] "phprtn: Integer (0.0)"
##  [56] "phptot: Integer (0.0)"
##  [57] "phpwsl: Integer (0.0)"
##  [58] "sand: Integer (0.0)"
##  [59] "silt: Integer (0.0)"
##  [60] "tceq: Integer (0.0)"
##  [61] "totc: Integer (0.0)"
##  [62] "wg0006: Integer (0.0)"
##  [63] "wg0010: Integer (0.0)"
##  [64] "wg0033: Integer (0.0)"
##  [65] "wg0100: Integer (0.0)"
##  [66] "wg0200: Integer (0.0)"
##  [67] "wg0500: Integer (0.0)"
##  [68] "wg1500: Integer (0.0)"
##  [69] "wv0006: Integer (0.0)"
##  [70] "wv0010: Integer (0.0)"
##  [71] "wv0033: Integer (0.0)"
##  [72] "wv0100: Integer (0.0)"
##  [73] "wv0200: Integer (0.0)"
##  [74] "wv0500: Integer (0.0)"
##  [75] "wv1500: Integer (0.0)"
##  [76] NA
##  [77] NA
##  [78] NA
##  [79] NA
##  [80] NA
##  [81] NA
##  [82] NA
##  [83] NA
```

```
##  [84] NA
##  [85] NA
##  [86] NA
##  [87] NA
##  [88] NA
##  [89] NA
##  [90] NA
##  [91] NA
##  [92] NA
##  [93] NA
##  [94] NA
##  [95] NA
##  [96] NA
##  [97] NA
##  [98] NA
##  [99] NA
## [100] NA
## [101] NA
## [102] NA
## [103] NA
## [104] NA
## [105] NA
## [106] NA
## [107] NA
## [108] NA
## [109] NA
## [110] NA
## [111] NA
## [112] NA
## [113] NA
## [114] NA
## [115] NA
## [116] NA
## [117] NA
## [118] NA
## [119] NA
## [120] NA
## [121] NA
## [122] NA
## [123] NA
## [124] NA
## [125] NA
## [126] NA
```

It seems that many fields marked as `NA` are reserved for future expansion of the soil properties.

These codes are found in the Procedures Manual.

The `gdalUtils::ogrinfo` function also allows SQL (Structured Query Language queries to limit the extent of the search, by using the optional `where` argument. To use this we need to know the field names, which we saw in the previous output.

For example, the profiles from India:

```
india.profiles.info <-
  ogrinfo(wfs, ro=TRUE, so=TRUE, q=FALSE,
          layer="ms:wosis_latest_profiles",
          where="country_name='India'")
```

Show the number of records and geographic extent:

```
ix.f <- grep("Feature Count", india.profiles.info)
india.profiles.info[ix.f]

## [1] "Feature Count: 199"

ix.e <- grep("Extent", india.profiles.info)
india.profiles.info[ix.e]

## [1] "Extent: (69.800000, 8.483333) - (94.050000, 32.000000)"
```

There are only Feature Count: 199 profiles from India. Of course many more have been described, but despite the aim of WoSIS to be a complete world database, up till now it has proved impossible to develop a data-sharing arrangement with the NBSSLUP.

## Display layer properties

We need to know the layer's meaning, format, extent etc. before we read it into R.

Select the first property in the information list above. Again call `ogrinfo` but this time also specifying a layer within the WFS. Show only the summary, not the features, with the `so=TRUE` optional argument.

Note: the `q=FALSE` option lists the geometry and feature count.

```
property.info <- gdalUtils::ogrinfo(wfs, layer = "ms:wosis_latest_bdfi33",
                                    ro = TRUE, so = TRUE, q = FALSE)
cat(property.info, sep="\n")

## INFO: Open of
`WFS:https://maps.isric.org/mapserv?map=/map/wosis_latest.map'
##       using driver `WFS' successful.
## Metadata:
##   ABSTRACT=Title: wosis_latest
##   PROVIDER_NAME=ISRIC - World Soil Reference
##   TITLE=Title: wosis_latest
##
## Layer name: ms:wosis_latest_bdfi33
## Metadata:
##   ABSTRACT=Bulk density of the fine earth fraction*, equilibrated at 33
```

```
kPa (kg/dm³).
##
## ISRIC is developing a centralized and user-focused server database, known
as ISRIC World Soil Information Service (WoSIS). The aims are to:
##
## • Safeguard world soil data "as is"
##
## • Share soil point (i.e. profile) data upon their standardization, and
ultimately harmonization
##
## • Provide quality-assessed, geo-referenced soil data for a growing range
of environmental applications (e.g. SoilGrids250m).
##
## At present, wosis_latest contains standardized data for 217,000 profiles.
The number of measured data for each property varies between profiles and with
depth, generally depending on the purpose of the initial studies. Further, in
most source data sets, there are fewer data for soil physical as opposed to
soil chemical attributes and there are fewer measurements for deeper than for
superficial horizons. Generally, limited quality information is associated
with the various source data. Further information is provided in a
(https://doi.org/10.5194/essd-12-299-2020) and
(https://www.isric.org/sites/default/files/WOSISprocedureManual_2020nov17web.
pdf).
##
## * The fine earth fraction is generally defined as being less than 2 mm.
However, an upper limit of 1 mm was used in the former Soviet Union  and its
sattelite states (Katchynsky scheme). This has been indicated in the
database.
##   KEYWORD_1=Soil
##   KEYWORD_2=Standard
##   KEYWORD_3=WoSIS
##   KEYWORD_4=latest
##   KEYWORD_5=Bulk density fine earth - 33 kPa
##   TITLE=WoSIS latest - Bulk density fine earth - 33 kPa
## Geometry: Unknown (any)
## Feature Count: 78058
## Extent: (-171.927505, -77.848663) - (161.600617, 76.228333)
## Layer SRS WKT:
## GEOGCRS["WGS 84",
##     DATUM["World Geodetic System 1984",
##         ELLIPSOID["WGS 84",6378137,298.257223563,
##             LENGTHUNIT["metre",1]]],
##     PRIMEM["Greenwich",0,
##         ANGLEUNIT["degree",0.0174532925199433]],
##     CS[ellipsoidal,2],
##         AXIS["geodetic latitude (Lat)",north,
##             ORDER[1],
##             ANGLEUNIT["degree",0.0174532925199433]],
##         AXIS["geodetic longitude (Lon)",east,
##             ORDER[2],
```

```
##             ANGLEUNIT["degree",0.0174532925199433]],
##     USAGE[
##         SCOPE["unknown"],
##         AREA["World"],
##         BBOX[-90,-180,90,180]],
##     ID["EPSG",4326]]
## Data axis to CRS axis mapping: 2,1
## Geometry Column = msGeometry
## gml_id: String (0.0) NOT NULL
## profile_id: Integer (0.0)
## profile_layer_id: Integer (0.0)
## country_name: String (0.0)
## upper_depth: Integer (0.0)
## lower_depth: Integer (0.0)
## layer_name: String (0.0)
## litter: Integer (0.0)
## bdfi33_value: String (0.0)
## bdfi33_value_avg: Real (0.0)
## bdfi33_method: String (0.0)
## bdfi33_date: String (0.0)
## bdfi33_dataset_id: String (0.0)
## bdfi33_profile_code: String (0.0)
## bdfi33_licence: String (0.0)
```

```
ix.f <- grep("Feature Count", property.info)
property.info[ix.f]
```

```
## [1] "Feature Count: 78058"
```

This gives an explanation of ISRIC and WoSIS, and then the keywords showing this property.

We see this is "Bulk density of the fine earth fraction* equilibrated at 33 kPa." Refer to the procedures manual for how each property (here, bulk density) was determined. The footnote * gives the definition of this term for this dataset. See also the feature count – this is how many layers have a bulk density value.

The `gdalUtils::ogrinfo` function allows SQL (Structured Query Language queries to limit the extent of the search, by using the optional `where` argument. To use this we need to know the field names, which we saw in the previous output.

For example, all the subsoil bulk densities from India:

```
bd.india.info <- gdalUtils::ogrinfo(wfs, layer="ms:wosis_latest_bdfi33",
                                    where="country_name='India' AND
upper_depth > 100",
                                    ro=TRUE, so=TRUE, q=FALSE)
ix.f <- grep("Feature Count", bd.india.info)
bd.india.info[ix.f]
```

```
## [1] "Feature Count: 32"
```

```
(n.records <- as.numeric(strsplit(bd.india.info[ix.f],
                                  split=": ", fixed=TRUE)[[1]][2]))
```

```
## [1] 32
```

Here there are only 32 records

## Import WoSIS datasets to the client system

There seems to be no way to directly import from the WFS to an R workspace object, so there must first be an intermediate step: download the WFS layer in an appropriate format to a local directory, and then import as usual for GIS layers.

The `gdalUtilities::ogr2ogr` function reads from one format on the server and writes to another in the client, i.e., your local files. The default output file format is an ESRI Shapefile; other formats can be specified with the (optional) `f` argument.

The possible formats are listed here. Here we explain how to access three of them: * OGC Geopackages. * ESRI Shapefiles * CSV files

The `where` and `spat` arguments can also be used here. You will often want to limit the size of the object with one or both of these. The `ogr2ogr` function also allows an (optional) transformation to any EPSG-defined CRS with the `t_srs` argument. See `?ogr2ogr` for more options.

Set up a directory on the local file system to receive the downloaded files:

```
wosis.dir.name <- "./wosis_latest"
if (!file.exists(wosis.dir.name)) dir.create(wosis.dir.name)
```

Note that **import via `ogr2ogr` can be quite slow**, because it depends on the network and the remote server, which may have a speed limitation to avoid overloads. Many of these downloads may take 15-20 minutes clock time, while only requiring less than a minute of local computer time.

Therefore, the codes that use `ogr2ogr` first check if the file has already been downloaded, and if so, skips the download. If you want to make sure to get the latest version of the files, delete any you have previously downloaded; this will force a download from the server.

### OGC Geopackage

WoSIS WFS is available as a OGC GeoPackage, using `f="GPKG` argument to `ogr2ogr`.

For example, the profiles as a GeoPackage:

```
layer.name <- "wosis_latest_profiles"
(dst.target.name <- paste0(wosis.dir.name,"/", layer.name, ".gpkg"))
```

```
## [1] "./wosis_latest/wosis_latest_profiles.gpkg"
```

```
if (!file.exists(dst.target.name)) {
system.time(
  gdalUtilities::ogr2ogr(src=wfs,
          dst=dst.target.name,
          layer=layer.name,
          f = "GPKG",
          overwrite=TRUE,
          skipfailures=TRUE)
)
}
file.info(dst.target.name)$size/1024/1024

## [1] 56.26953
```

This Geopackage is about 52 Mb.

### Reading imported profiles into R

Once the Geopackage has been downloaded to local storage, it can be read into R with the `st_read` function of the `sf` "Simple Features" package.

```
profiles.gpkg <- st_read(dst.target.name)

## Reading layer `ms:wosis_latest_profiles' from data source
##
`/Users/rossiter/data_noCloud/ISRIC_WoSIS/wosis_latest/wosis_latest_profiles.
gpkg'
##   using driver `GPKG'
## Simple feature collection with 217184 features and 75 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: -172.3633 ymin: -77.84866 xmax: 178.5 ymax: 79.9
## Geodetic CRS:  WGS 84

class(profiles.gpkg)

## [1] "sf"         "data.frame"

dim(profiles.gpkg)

## [1] 217184     76

names(profiles.gpkg)

##  [1] "gml_id"                        "profile_id"
##  [3] "dataset_id"                    "continent"
##  [5] "country_id"                    "country_name"
##  [7] "geom_accuracy"                 "latitude"
##  [9] "longitude"                     "dsds"
## [11] "cfao_version"                  "cfao_major_group_code"
## [13] "cfao_major_group"              "cfao_soil_unit_code"
## [15] "cfao_soil_unit"                "cwrb_version"
```

```
## [17] "cwrb_reference_soil_group_code" "cwrb_reference_soil_group"
## [19] "cwrb_prefix_qualifier"          "cwrb_suffix_qualifier"
## [21] "cstx_version"                   "cstx_order_name"
## [23] "cstx_suborder"                  "cstx_great_group"
## [25] "cstx_subgroup"                  "bdfi33"
## [27] "bdfiad"                         "bdfifm"
## [29] "bdfiod"                         "bdws33"
## [31] "bdwsad"                         "bdwsfm"
## [33] "bdwsod"                         "cecph7"
## [35] "cecph8"                         "cfgr"
## [37] "cfvo"                           "clay"
## [39] "ecec"                           "elco20"
## [41] "elco25"                         "elco50"
## [43] "elcosp"                         "nitkjd"
## [45] "orgc"                           "orgm"
## [47] "phaq"                           "phba"
## [49] "phca"                           "phetb1"
## [51] "phetm3"                         "phetol"
## [53] "phkc"                           "phnf"
## [55] "phprtn"                         "phptot"
## [57] "phpwsl"                         "sand"
## [59] "silt"                           "tceq"
## [61] "totc"                           "wg0006"
## [63] "wg0010"                         "wg0033"
## [65] "wg0100"                         "wg0200"
## [67] "wg0500"                         "wg1500"
## [69] "wv0006"                         "wv0010"
## [71] "wv0033"                         "wv0100"
## [73] "wv0200"                         "wv0500"
## [75] "wv1500"                         "msGeometry"
```

Many of these names have no content at the whole-profile (site) level.

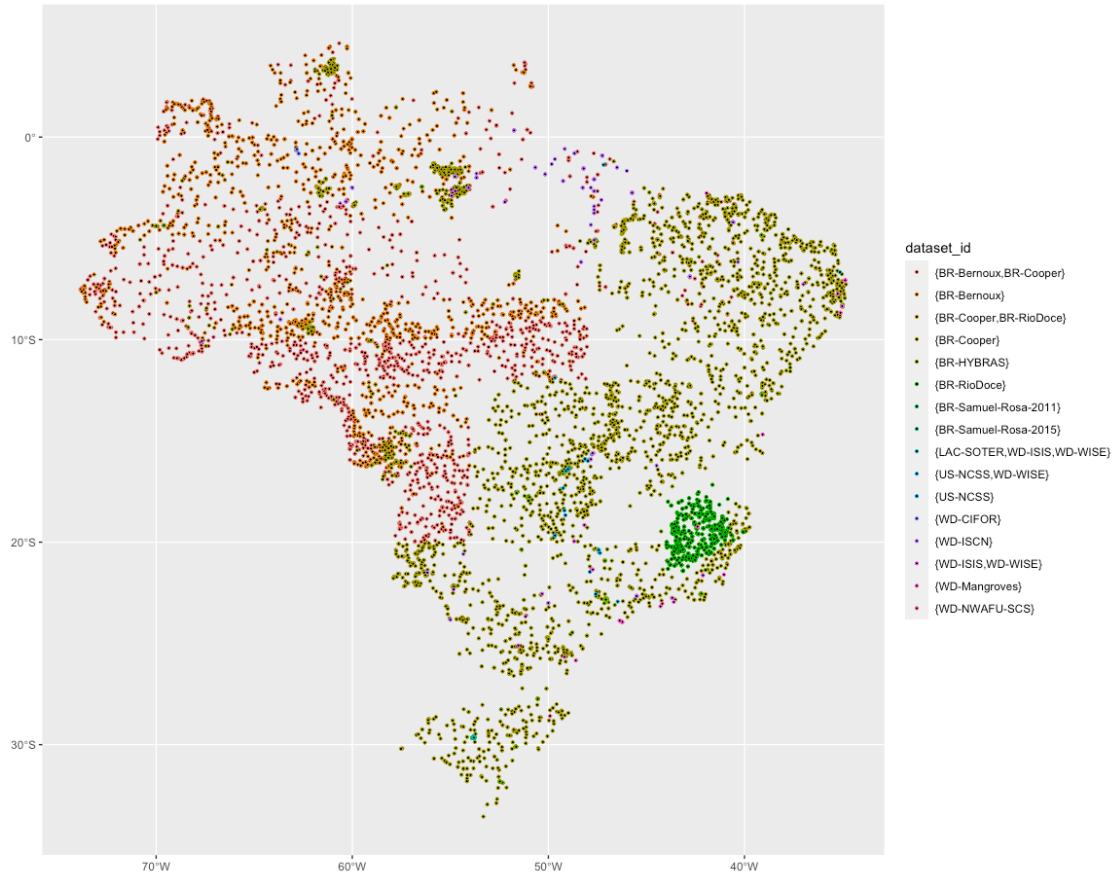
Which countries have the most profiles?

```
head(sort(table(profiles.gpkg$country_name), decreasing=TRUE))
```

```
##
## United States of America                Australia                    Chile
##                    56372                    42731                    13686
##              Switzerland                   Brazil                   Canada
##                    11073                     9242                     8499
```

These have georeference and so can be mapped. Here is an example of the profiles from Brazil, showing their source datasets:

```
ggplot(data=profiles.gpkg[(profiles.gpkg$country_name=="Brazil"), ]) +
  aes(col=dataset_id) +
  geom_sf(shape=21, size=0.8, fill="black")
```




## ESRI shapefiles

### WoSIS profiles as ESRI Shapefiles

WoSIS WFS is available as ESRI point shapefiles, using `f="ESRI Shapefile"` argument to `ogr2ogr`.

Download the profile information for the whole world. This is a very large file.

```
layer.name <- "ms_wosis_latest_profiles"
(dst.target.name <- paste0(wosis.dir.name,"/", layer.name, ".shp"))

## [1] "./wosis_latest/ms_wosis_latest_profiles.shp"

if (!file.exists(dst.target.name)) {
system.time(
  gdalUtilities::ogr2ogr(src=wfs,
```

```
          dst=wosis.dir.name,
          layer=layer.name,
          f = "ESRI Shapefile",
          overwrite=TRUE,
          skipfailures=TRUE)
)
}
file.info(dst.target.name)$size/1024/1024

## [1] 5.799534
```

The number of profiles can be restricted with a `spat` spatial extent or a `where` SQL query. For example, to download just the Indian profiles, into a subdirectory:

```
wosis.dir.name.india <- "./wosis_latest/india"
if (!file.exists(wosis.dir.name.india)) dir.create(wosis.dir.name.india)
layer.name <- "ms_wosis_latest_profiles"
(dst.target.name <- paste0(wosis.dir.name,"/", layer.name, ".shp"))

## [1] "./wosis_latest/ms_wosis_latest_profiles.shp"

if (!file.exists(dst.target.name)) {
  system.time(
  gdalUtilities::ogr2ogr(src=wfs,
          dst=wosis.dir.name.india,
          layer=layer.name,
          f = "ESRI Shapefile",
          where="country_name='India'",
          overwrite=TRUE,
          skipfailures=TRUE)
)
}
file.info(dst.target.name)$size/1024/1024

## [1] 5.799534
```

### Reading imported profiles into R

Now read the downloaded shapefile of profiles into an R `sp` object. For shapefiles, the directory and layer names must be specified separately as two arguments, `dsn` ("data set name") and `layer`. Notice how the server name of this layer which begins with `ms:` has been changed to `ms_` during import, due to restrictions on file names on the local file system.

```
layer.name <- "ms_wosis_latest_profiles"
profiles <- sf::st_read(dsn=wosis.dir.name, layer=layer.name,
                stringsAsFactors = FALSE)

## Reading layer `ms_wosis_latest_profiles' from data source
##   `/Users/rossiter/data_noCloud/ISRIC_WoSIS/wosis_latest' using driver
`ESRI Shapefile'
## Simple feature collection with 217184 features and 75 fields
## Geometry type: POINT
```

```
## Dimension:     XY
## Bounding box:  xmin: -172.3633 ymin: -77.84866 xmax: 178.5 ymax: 79.9
## Geodetic CRS:  WGS 84
```

```
class(profiles)
```

```
## [1] "sf"         "data.frame"
```

```
dim(profiles)
```

```
## [1] 217184     76
```

```
names(profiles)
```

```
##  [1] "gml_id"     "profile_id" "dataset_id" "continent"  "country_id"
##  [6] "country_na" "geom_accur" "latitude"   "longitude"  "dsds"
## [11] "cfao_versi" "cfao_major" "cfao_maj_1" "cfao_soil_" "cfao_soi_1"
## [16] "cwrb_versi" "cwrb_refer" "cwrb_ref_1" "cwrb_prefi" "cwrb_suffi"
## [21] "cstx_versi" "cstx_order" "cstx_subor" "cstx_great" "cstx_subgr"
## [26] "bdfi33"     "bdfiad"     "bdfifm"     "bdfiod"     "bdws33"
## [31] "bdwsad"     "bdwsfm"     "bdwsod"     "cecph7"     "cecph8"
## [36] "cfgr"       "cfvo"       "clay"       "ecec"       "elco20"
## [41] "elco25"     "elco50"     "elcosp"     "nitkjd"     "orgc"
## [46] "orgm"       "phaq"       "phba"       "phca"       "phetb1"
## [51] "phetm3"     "phetol"     "phkc"       "phnf"       "phprtn"
## [56] "phptot"     "phpwsl"     "sand"       "silt"       "tceq"
## [61] "totc"       "wg0006"     "wg0010"     "wg0033"     "wg0100"
## [66] "wg0200"     "wg0500"     "wg1500"     "wv0006"     "wv0010"
## [71] "wv0033"     "wv0100"     "wv0200"     "wv0500"     "wv1500"
## [76] "geometry"
```

```
head(profiles)
```

```
## Simple feature collection with 6 features and 75 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: -6.4167 ymin: 6.5875 xmax: 2.1525 ymax: 58.3333
## Geodetic CRS:  WGS 84
##                          gml_id profile_id               dataset_id
continent
## 1 wosis_latest_profiles.598717     598717               {EU-SPADE}
Europe
## 2 wosis_latest_profiles.598723     598723               {EU-SPADE}
Europe
## 3 wosis_latest_profiles.598856     598856               {EU-SPADE}
Europe
## 4 wosis_latest_profiles.598916     598916               {EU-SPADE}
Europe
## 5 wosis_latest_profiles.598976     598976               {EU-SPADE}
Europe
## 6  wosis_latest_profiles.47431      47431 {AF-AfSP,BJSOTER,WD-WISE}
Africa
```

```
##   country_id     country_na geom_accur latitude longitude dsds cfao_versi
## 1         GB United Kingdom      1e-06  58.1333   -4.5667   95       1997
## 2         GB United Kingdom      1e-06  57.2167   -5.8000  100       1997
## 3         GB United Kingdom      1e-06  57.3167   -6.4167   45       1997
## 4         GB United Kingdom      1e-06  57.0667   -4.8167   42       1974
## 5         GB United Kingdom      1e-06  58.3333   -3.5333   92       1997
## 6         BJ           Benin      1e-04   6.5875    2.1525  150       1974
##   cfao_major cfao_maj_1 cfao_soil_ cfao_soi_1 cwrb_versi cwrb_refer
cwrb_ref_1
## 1         HS   Histosol          d    Dystric         NA       <NA>
<NA>
## 2         HS   Histosol          d    Dystric         NA       <NA>
<NA>
## 3         CM   Cambisol          d    Dystric         NA       <NA>
<NA>
## 4          U     Ranker       <NA>       <NA>         NA       <NA>
<NA>
## 5         HS   Histosol          d    Dystric         NA       <NA>
<NA>
## 6          W   Planosol          d    Dystric       2006         PL
Planosol
##   cwrb_prefi cwrb_suffi cstx_versi cstx_order cstx_subor cstx_great
cstx_subgr
## 1       <NA>       <NA>         NA       <NA>       <NA>       <NA>
<NA>
## 2       <NA>       <NA>         NA       <NA>       <NA>       <NA>
<NA>
## 3       <NA>       <NA>         NA       <NA>       <NA>       <NA>
<NA>
## 4       <NA>       <NA>         NA       <NA>       <NA>       <NA>
<NA>
## 5       <NA>       <NA>         NA       <NA>       <NA>       <NA>
<NA>
## 6       <NA>       <NA>         NA       <NA>       <NA>       <NA>
<NA>
##   bdfi33 bdfiad bdfifm bdfiod bdws33 bdwsad bdwsfm bdwsod cecph7 cecph8
cfgr
## 1      0      0      0      0      0      0      0      0      0      0
0
## 2      0      0      0      0      0      0      0      0      0      0
0
## 3      0      0      0      0      0      0      0      0      0      0
0
## 4      0      0      0      0      0      0      0      0      0      0
0
## 5      0      0      0      0      0      0      0      0      0      0
0
## 6      0      0      0      0      0      0      0      0      0      0
0
##   cfvo clay ecec elco20 elco25 elco50 elcosp nitkjd orgc orgm phaq phba
```

```
phca
## 1    0    0    0      0      0      0      0      0    0    0    0    0
0
## 2    0    0    0      0      0      0      0      0    0    0    0    0
0
## 3    0    0    0      0      0      0      0      0    0    0    0    0
0
## 4    0    0    0      0      0      0      0      0    0    0    0    0
0
## 5    0    0    0      0      0      0      0      0    0    0    0    0
0
## 6    0    0    0      0      0      0      0      0    0    0    0    0
0
##   phetb1 phetm3 phetol phkc phnf phprtn phptot phpwsl sand silt tceq totc
## 1      0      0      0    0    0      0      0      0    0    0    0    0
## 2      0      0      0    0    0      0      0      0    0    0    0    0
## 3      0      0      0    0    0      0      0      0    0    0    0    0
## 4      0      0      0    0    0      0      0      0    0    0    0    0
## 5      0      0      0    0    0      0      0      0    0    0    0    0
## 6      0      0      0    0    0      0      0      0    0    0    0    0
##   wg0006 wg0010 wg0033 wg0100 wg0200 wg0500 wg1500 wv0006 wv0010 wv0033
wv0100
## 1      0      0      0      0      0      0      0      0      0      0
0
## 2      0      0      0      0      0      0      0      0      0      0
0
## 3      0      0      0      0      0      0      0      0      0      0
0
## 4      0      0      0      0      0      0      0      0      0      0
0
## 5      0      0      0      0      0      0      0      0      0      0
0
## 6      0      0      0      0      0      0      0      0      0      0
0
##   wv0200 wv0500 wv1500                geometry
## 1      0      0      0 POINT (-4.5667 58.1333)
## 2      0      0      0    POINT (-5.8 57.2167)
## 3      0      0      0 POINT (-6.4167 57.3167)
## 4      0      0      0 POINT (-4.8167 57.0667)
## 5      0      0      0 POINT (-3.5333 58.3333)
## 6      0      0      0   POINT (2.1525 6.5875)
```

The current database has 76 profiles. Quite a resource!

The Indian profiles:

```
profiles.india <- sf::st_read(dsn=wosis.dir.name.india, layer=layer.name,
                stringsAsFactors = FALSE)

## Reading layer `ms_wosis_latest_profiles' from data source
##   `/Users/rossiter/data_noCloud/ISRIC_WoSIS/wosis_latest/india'
```

```
##   using driver `ESRI Shapefile'
## Simple feature collection with 199 features and 75 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 69.8 ymin: 8.483333 xmax: 94.05 ymax: 32
## Geodetic CRS:  WGS 84

dim(profiles.india)

## [1] 199  76

names(profiles.india)

##  [1] "gml_id"     "profile_id" "dataset_id" "continent"  "country_id"
##  [6] "country_na" "geom_accur" "latitude"   "longitude"  "dsds"
## [11] "cfao_versi" "cfao_major" "cfao_maj_1" "cfao_soil_" "cfao_soi_1"
## [16] "cwrb_versi" "cwrb_refer" "cwrb_ref_1" "cwrb_prefi" "cwrb_suffi"
## [21] "cstx_versi" "cstx_order" "cstx_subor" "cstx_great" "cstx_subgr"
## [26] "bdfi33"     "bdfiad"     "bdfifm"     "bdfiod"     "bdws33"
## [31] "bdwsad"     "bdwsfm"     "bdwsod"     "cecph7"     "cecph8"
## [36] "cfgr"       "cfvo"       "clay"       "ecec"       "elco20"
## [41] "elco25"     "elco50"     "elcosp"     "nitkjd"     "orgc"
## [46] "orgm"       "phaq"       "phba"       "phca"       "phetb1"
## [51] "phetm3"     "phetol"     "phkc"       "phnf"       "phprtn"
## [56] "phptot"     "phpwsl"     "sand"       "silt"       "tceq"
## [61] "totc"       "wg0006"     "wg0010"     "wg0033"     "wg0100"
## [66] "wg0200"     "wg0500"     "wg1500"     "wv0006"     "wv0010"
## [71] "wv0033"     "wv0100"     "wv0200"     "wv0500"     "wv1500"
## [76] "geometry"

head(profiles.india)

## Simple feature collection with 6 features and 75 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 76.83334 ymin: 10.78333 xmax: 88.95738 ymax: 21.8277
## Geodetic CRS:  WGS 84
##                         gml_id profile_id     dataset_id continent
country_id
## 1 wosis_latest_profiles.622261     622261 {WD-Mangroves}      Asia
IN
## 2 wosis_latest_profiles.622883     622883 {WD-Mangroves}      Asia
IN
## 3 wosis_latest_profiles.623762     623762 {WD-NWAFU-SCS}      Asia
IN
## 4 wosis_latest_profiles.622603     622603 {WD-Mangroves}      Asia
IN
## 5 wosis_latest_profiles.622649     622649 {WD-Mangroves}      Asia
IN
## 6 wosis_latest_profiles.623297     623297 {WD-Mangroves}      Asia
IN
```

```
##   country_na geom_accur latitude longitude dsds cfao_versi cfao_major
## 1      India      1e-06 20.38793  86.77038   30         NA       <NA>
## 2      India      1e-06 20.39427  86.73617   30         NA       <NA>
## 3      India      1e-06 10.78333  76.83334  100         NA       <NA>
## 4      India      1e-06 21.66687  88.33126   25         NA       <NA>
## 5      India      1e-06 21.68509  88.95738   40         NA       <NA>
## 6      India      1e-06 21.82770  88.84857   40         NA       <NA>
##   cfao_maj_1 cfao_soil_ cfao_soi_1 cwrb_versi cwrb_refer cwrb_ref_1
cwrb_prefi
## 1       <NA>       <NA>       <NA>         NA       <NA>       <NA>
<NA>
## 2       <NA>       <NA>       <NA>         NA       <NA>       <NA>
<NA>
## 3       <NA>       <NA>       <NA>         NA       <NA>       <NA>
<NA>
## 4       <NA>       <NA>       <NA>         NA       <NA>       <NA>
<NA>
## 5       <NA>       <NA>       <NA>         NA       <NA>       <NA>
<NA>
## 6       <NA>       <NA>       <NA>         NA       <NA>       <NA>
<NA>
##   cwrb_suffi cstx_versi cstx_order cstx_subor cstx_great cstx_subgr bdfi33
## 1       <NA>         NA       <NA>       <NA>       <NA>       <NA>      0
## 2       <NA>         NA       <NA>       <NA>       <NA>       <NA>      0
## 3       <NA>         NA       <NA>       <NA>       <NA>       <NA>      0
## 4       <NA>         NA       <NA>       <NA>       <NA>       <NA>      0
## 5       <NA>         NA       <NA>       <NA>       <NA>       <NA>      0
## 6       <NA>         NA       <NA>       <NA>       <NA>       <NA>      0
##   bdfiad bdfifm bdfiod bdws33 bdwsad bdwsfm bdwsod cecph7 cecph8 cfgr cfvo
clay
## 1      0      0      0      0      0      0      0      0      0    0    0
0
## 2      0      0      0      0      0      0      0      0      0    0    0
0
## 3      0      0      0      0      0      0      0      0      0    0    0
0
## 4      0      0      0      0      0      0      0      0      0    0    0
0
## 5      0      0      0      0      0      0      0      0      0    0    0
0
## 6      0      0      0      0      0      0      0      0      0    0    0
0
##   ecec elco20 elco25 elco50 elcosp nitkjd orgc orgm phaq phba phca phetb1
## 1    0      0      0      0      0      0    0    0    0    0    0      0
## 2    0      0      0      0      0      0    0    0    0    0    0      0
## 3    0      0      0      0      0      0    0    0    0    0    0      0
## 4    0      0      0      0      0      0    0    0    0    0    0      0
## 5    0      0      0      0      0      0    0    0    0    0    0      0
## 6    0      0      0      0      0      0    0    0    0    0    0      0
##   phetm3 phetol phkc phnf phprtn phptot phpwsl sand silt tceq totc wg0006
```

```
## 1      0      0    0    0      0      0      0    0    0    0    0      0
## 2      0      0    0    0      0      0      0    0    0    0    0      0
## 3      0      0    0    0      0      0      0    0    0    0    0      0
## 4      0      0    0    0      0      0      0    0    0    0    0      0
## 5      0      0    0    0      0      0      0    0    0    0    0      0
## 6      0      0    0    0      0      0      0    0    0    0    0      0
##   wg0010 wg0033 wg0100 wg0200 wg0500 wg1500 wv0006 wv0010 wv0033 wv0100
wv0200
## 1      0      0      0      0      0      0      0      0      0      0
0
## 2      0      0      0      0      0      0      0      0      0      0
0
## 3      0      0      0      0      0      0      0      0      0      0
0
## 4      0      0      0      0      0      0      0      0      0      0
0
## 5      0      0      0      0      0      0      0      0      0      0
0
## 6      0      0      0      0      0      0      0      0      0      0
0
##   wv0500 wv1500                   geometry
## 1      0      0 POINT (86.77038 20.38793)
## 2      0      0 POINT (86.73617 20.39427)
## 3      0      0 POINT (76.83334 10.78333)
## 4      0      0 POINT (88.33126 21.66687)
## 5      0      0 POINT (88.95738 21.68509)
## 6      0      0  POINT (88.84857 21.8277)
```

Here there are only 76 profiles, as we saw with `ogrinfo`, above.

### WoSIS layers as ESRI Shapefiles

Download all records for a single property, here, bulk density, and save it a subdirectory of the current path, and with the default format. The directory must be first created if does not already exist.

Note `f=` argument to specify the format, here ESRI shapefiles.

```
layer.name <- "ms_wosis_latest_bdfi33"
(dst.target.name <- paste0(wosis.dir.name,"/", layer.name, ".shp"))
```

```
## [1] "./wosis_latest/ms_wosis_latest_bdfi33.shp"
```

```
if (!file.exists(dst.target.name)) {
  system.time(
  gdalUtilities::ogr2ogr(src=wfs,
          dst=wosis.dir.name,
          layer=layer.name,
          f = "ESRI Shapefile",
          overwrite=TRUE,
          skipfailures=TRUE)
```

```
)
}
file.info(dst.target.name)$size/1024/1024

## [1] 2.084469
```

Because the destination format is `"ESRI Shapefile"` many of the field names have to be shortened, as shown by warnings such as "Warning 6: Normalized/laundered field name: 'profile_layer_id' to 'profile_la'."

Note that `spat` is not applicable in this query because there is no spatial information in the attribute tables. However, it is possible to include an SQL query with `where` to limit the size of the download:

```
layer.name <- "ms_wosis_latest_bdfi33"
(dst.target.name <- paste0(wosis.dir.name.india,"/", layer.name, ".shp"))

## [1] "./wosis_latest/india/ms_wosis_latest_bdfi33.shp"

if (!file.exists(dst.target.name)) {
  system.time(
    gdalUtilities::ogr2ogr(src=wfs,
                           dst=wosis.dir.name.india,
                           layer=layer.name,
                           f = "ESRI Shapefile",
                           where="country_name='India'",
                           overwrite=TRUE,
                           skipfailures=TRUE)
  )
}
file.info(dst.target.name)$size/1024/1024

## [1] 0.003219604
```

### Reading imported layers into R

Now read the downloaded shapefiles into an R `sf` object.

For shapefiles, the directory and layer names must be specified separately as two arguments, `dsn` ("data set name") and `layer`. Notice how the server name of this layer which begins with `ms:` has been changed to `ms_` during import, due to restrictions on file names on the local file system.

```
# here strings are just that, not to be interpreted as R factors
layer.name <- "ms_wosis_latest_bdfi33"
bd33 <- st_read(dsn=wosis.dir.name, layer=layer.name,
                stringsAsFactors = FALSE)

## Reading layer `ms_wosis_latest_bdfi33' from data source
##   `/Users/rossiter/data_noCloud/ISRIC_WoSIS/wosis_latest' using driver
`ESRI Shapefile'
## Simple feature collection with 78058 features and 15 fields
```

```
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: -171.9275 ymin: -77.84866 xmax: 161.6006 ymax:
76.22833
## Geodetic CRS:  WGS 84

class(bd33)

## [1] "sf"         "data.frame"

names(bd33)

##  [1] "gml_id"     "profile_id" "profile_la" "country_na" "upper_dept"
##  [6] "lower_dept" "layer_name" "litter"     "bdfi33_val" "bdfi33_v_1"
## [11] "bdfi33_met" "bdfi33_dat" "bdfi33_d_1" "bdfi33_pro" "bdfi33_lic"
## [16] "geometry"

head(bd33)

## Simple feature collection with 6 features and 15 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 29.31667 ymin: -3.366667 xmax: 29.31667 ymax: -
3.366667
## Geodetic CRS:  WGS 84
##                        gml_id profile_id profile_la country_na upper_dept
## 1 wosis_latest_bdfi33.597709      47145     597709    Burundi          0
## 2 wosis_latest_bdfi33.597710      47145     597710    Burundi         15
## 3 wosis_latest_bdfi33.597711      47145     597711    Burundi         40
## 4 wosis_latest_bdfi33.597712      47145     597712    Burundi         70
## 5 wosis_latest_bdfi33.597713      47145     597713    Burundi        110
## 6 wosis_latest_bdfi33.597714      47145     597714    Burundi        170
##   lower_dept layer_name litter                                      bdfi33_val
## 1         15          A      0 {1:1.41,2:1.41,3:1.08,4:1.08,5:1.12,6:1.12}
## 2         40        Bho      0 {1:1.04,2:1.04,3:0.99,4:0.99,5:1.36,6:1.36}
## 3         70        Bo1      0 {1:1.25,2:1.25,3:0.95,4:0.95,5:0.97,6:0.97}
## 4        110        Bo2      0 {1:1.14,2:1.14,3:1.12,4:1.12,5:1.43,6:1.43}
## 5        170        Bo3      0 {1:1.42,2:1.42,3:1.12,4:1.12,5:1.14,6:1.14}
## 6        210         Bt      0 {1:1.29,2:1.29,3:1.28,4:1.28,5:1.64,6:1.64}
##   bdfi33_v_1
## 1       1.20
## 2       1.13
## 3       1.06
## 4       1.23
## 5       1.23
## 6       1.40
##
bdfi33_met
## 1 {"1:calculation = not specified, corrections = not specified,
measurement condition = equilibrated at 33 kPa (~1/3 bar), sample type = not
specified","2:calculation = not specified, corrections = not specified,
```

```
measurement condition = equilibrated at 33
## 2 {"1:calculation = not specified, corrections = not specified,
measurement condition = equilibrated at 33 kPa (~1/3 bar), sample type = not
specified","2:calculation = not specified, corrections = not specified,
measurement condition = equilibrated at 33
## 3 {"1:calculation = not specified, corrections = not specified,
measurement condition = equilibrated at 33 kPa (~1/3 bar), sample type = not
specified","2:calculation = not specified, corrections = not specified,
measurement condition = equilibrated at 33
## 4 {"1:calculation = not specified, corrections = not specified,
measurement condition = equilibrated at 33 kPa (~1/3 bar), sample type = not
specified","2:calculation = not specified, corrections = not specified,
measurement condition = equilibrated at 33
## 5 {"1:calculation = not specified, corrections = not specified,
measurement condition = equilibrated at 33 kPa (~1/3 bar), sample type = not
specified","2:calculation = not specified, corrections = not specified,
measurement condition = equilibrated at 33
## 6 {"1:calculation = not specified, corrections = not specified,
measurement condition = equilibrated at 33 kPa (~1/3 bar), sample type = not
specified","2:calculation = not specified, corrections = not specified,
measurement condition = equilibrated at 33
##
bdfi33_dat
## 1 {1:1983-10-3,2:1983-10-3,3:1983-10-3,4:1983-10-3,5:1983-10-3,6:1983-10-
3}
## 2 {1:1983-10-3,2:1983-10-3,3:1983-10-3,4:1983-10-3,5:1983-10-3,6:1983-10-
3}
## 3 {1:1983-10-3,2:1983-10-3,3:1983-10-3,4:1983-10-3,5:1983-10-3,6:1983-10-
3}
## 4 {1:1983-10-3,2:1983-10-3,3:1983-10-3,4:1983-10-3,5:1983-10-3,6:1983-10-
3}
## 5 {1:1983-10-3,2:1983-10-3,3:1983-10-3,4:1983-10-3,5:1983-10-3,6:1983-10-
3}
## 6 {1:1983-10-3,2:1983-10-3,3:1983-10-3,4:1983-10-3,5:1983-10-3,6:1983-10-
3}
##   bdfi33_d_1 bdfi33_pro
## 1    US-NCSS    84P0286
## 2    US-NCSS    84P0286
## 3    US-NCSS    84P0286
## 4    US-NCSS    84P0286
## 5    US-NCSS    84P0286
## 6    US-NCSS    84P0286
##                                                      bdfi33_lic
## 1 U.S. Public Domain http://www.usa.gov/publicdomain/label/1.0/
## 2 U.S. Public Domain http://www.usa.gov/publicdomain/label/1.0/
## 3 U.S. Public Domain http://www.usa.gov/publicdomain/label/1.0/
## 4 U.S. Public Domain http://www.usa.gov/publicdomain/label/1.0/
## 5 U.S. Public Domain http://www.usa.gov/publicdomain/label/1.0/
## 6 U.S. Public Domain http://www.usa.gov/publicdomain/label/1.0/
##                      geometry
```

```
## 1 POINT (29.31667 -3.366667)
## 2 POINT (29.31667 -3.366667)
## 3 POINT (29.31667 -3.366667)
## 4 POINT (29.31667 -3.366667)
## 5 POINT (29.31667 -3.366667)
## 6 POINT (29.31667 -3.366667)
```

Each record has some identification:

- `gml_id` the attribute name + `profile_la` (see below)
- `profile_id` profile internal ID
- `profile_la` profile + layer internal ID
- `country_na` country name
- `upper_dept` upper limit of layer from soil surface (excluding litter layer). cm
- `lower_dept` lower limit of layer from soil surface (excluding litter layer), cm
- `layer_name` layer name as assigned during original profile description
- `litter` whether the layer is a litter layer (0 = no, 1 = yes)

Each attribute has several names, with the following extensions:

- `value` – one or more values, in the format {1:value; 2:value…}, which are duplicate measurements
- `value_avg` – the average of the values
- `method` – text description of the analytical method
- `date` – one or more values, in the format {1:yyyy-mm-dd; 2:yyyy-mm-dd…}, which are the dates each of th duplicate measurements was added to the database (not the original measurement date, nor the field sampling date)
- `dataset_id` – text code of original database
- `profile_code` – text code of profile from original database
- `licence` – text string of the Creative Commons[1] license for this value, e.g. `CC-BY-NC`

So for example the attribute `bdfi33` has the following fields, shortened when input to a shapefile to the first ten characters; if these would be duplicated the name is further manipulated:

- `bdfi33_value` → `bdfi33_val`
- `bdfi33_value_avg` → `bdfi33_v_1`
- `bdfi33_method` → `bdfi33_met`
- `bdfi33_date` → `bdfi33_dat`
- `bdfi33_dataset_id` → `bdfi33_d_1`
- `bdfi33_profile_code` → `bdfi33_pro`
- `bdfi33_licence` → `bdfi33_lic`

---

[1] https://creativecommons.org/licenses/

The shapefile has been imported as a `SpatialPointsDataFrame`, a class within the `sp` package, with the correct CRS, as we saw from the `ogrinfo`.

Examine the format of the attribute, this is in field `*_val`:

```
head(bd33$bdfi33_val)
```

```
## [1] "{1:1.41,2:1.41,3:1.08,4:1.08,5:1.12,6:1.12}"
## [2] "{1:1.04,2:1.04,3:0.99,4:0.99,5:1.36,6:1.36}"
## [3] "{1:1.25,2:1.25,3:0.95,4:0.95,5:0.97,6:0.97}"
## [4] "{1:1.14,2:1.14,3:1.12,4:1.12,5:1.43,6:1.43}"
## [5] "{1:1.42,2:1.42,3:1.12,4:1.12,5:1.14,6:1.14}"
## [6] "{1:1.29,2:1.29,3:1.28,4:1.28,5:1.64,6:1.64}"
```

The format is `{seq:val[,seq:val]}` where the `seq` is an integer on `[1...]` indicating which measurement number – note that there can be more than one measurement per property, e.g., repeated lab. measurements, and `val` is the numeric value.

But the average value for a layer has its own field, so if we only want the average, it is prepared for us. We see an example here, from six rows chosen to show several profiles with their layers:

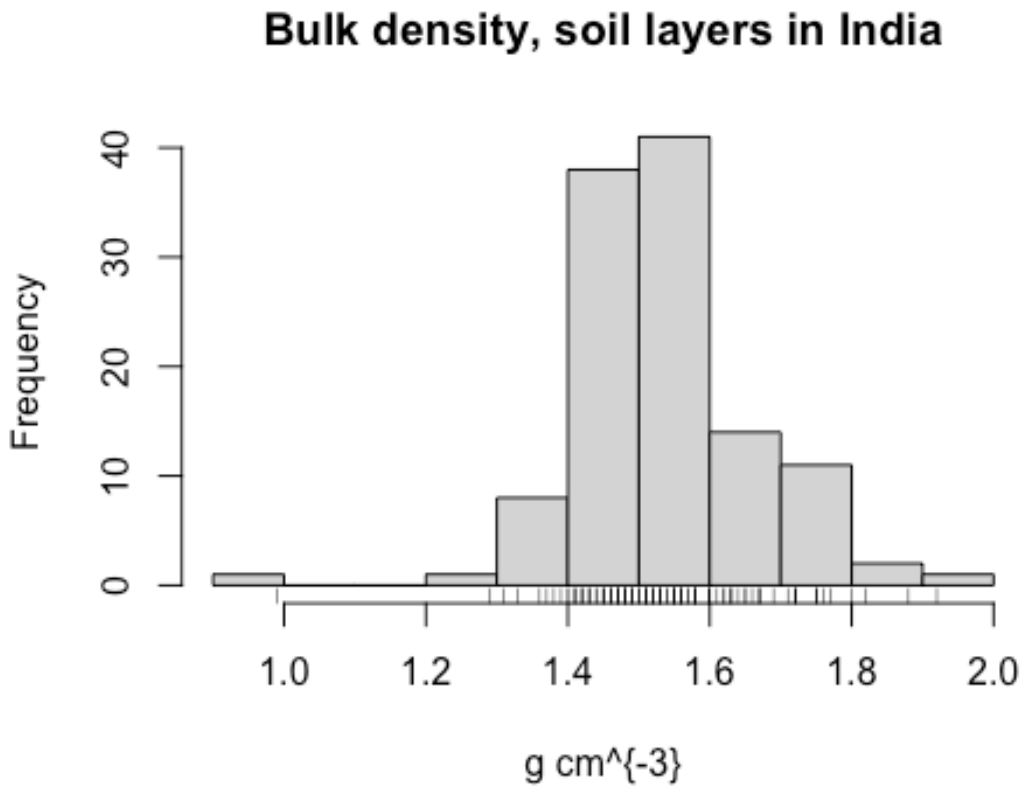
```
bd33[75:80,
c("profile_id","upper_dept","lower_dept","bdfi33_val","bdfi33_v_1")]
```

```
## Simple feature collection with 6 features and 5 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 23.53333 ymin: -19.84944 xmax: 23.53583 ymax: -
19.84556
## Geodetic CRS:  WGS 84
##    profile_id upper_dept lower_dept
bdfi33_val
## 75      48231        130        160
{1:0.95,2:0.95,3:1.65,4:1.65,5:1.95,6:1.95}
## 76      48234          0         16
{1:0.78,2:0.78,3:1.01,4:1.01,5:0.69,6:0.69}
## 77      48234         16         55
{1:1.45,2:1.45,3:1.83,4:1.83,5:1.67,6:1.67}
## 78      48234         55         94
{1:1.73,2:1.73,3:1.94,4:1.94,5:1.61,6:1.61}
## 79      48234         94        127
{1:1.58,2:1.58,3:1.92,4:1.92,5:1.70,6:1.70}
## 80      48234        127        184
{1:1.70,2:1.70,3:1.92,4:1.92,5:1.61,6:1.61}
##    bdfi33_v_1                   geometry
## 75       1.52 POINT (23.53333 -19.84944)
## 76       0.83 POINT (23.53583 -19.84556)
## 77       1.65 POINT (23.53583 -19.84556)
## 78       1.76 POINT (23.53583 -19.84556)
```

```
## 79       1.73 POINT (23.53583 -19.84556)
## 80       1.74 POINT (23.53583 -19.84556)
```

Here is the India example. We make a histogram of the representative values.

```
layer.name <- "ms_wosis_latest_bdfi33"
bd33.india <- st_read(dsn=wosis.dir.name.india, layer=layer.name,
                stringsAsFactors = FALSE)

## Reading layer `ms_wosis_latest_bdfi33' from data source
##   `/Users/rossiter/data_noCloud/ISRIC_WoSIS/wosis_latest/india'
##   using driver `ESRI Shapefile'
## Simple feature collection with 117 features and 15 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 75 ymin: 9 xmax: 79.98333 ymax: 28.58333
## Geodetic CRS:  WGS 84

class(bd33.india)

## [1] "sf"         "data.frame"

dim(bd33.india)

## [1] 117  16

(profile.id.india <- unique(bd33.india$profile_id))

##  [1]  66490  66492  67280  67281  67282  67283  67294  67295  67296  67782
## [11] 170991 170992 170993 170994 170995 170996 170997 170998 170999 171000

names(bd33.india)

##  [1] "gml_id"     "profile_id" "profile_la" "country_na" "upper_dept"
##  [6] "lower_dept" "layer_name" "litter"     "bdfi33_val" "bdfi33_v_1"
## [11] "bdfi33_met" "bdfi33_dat" "bdfi33_d_1" "bdfi33_pro" "bdfi33_lic"
## [16] "geometry"

hist(bd33.india$bdfi33_v_1, main="Bulk density, soil layers in India",
     xlab="g cm^{-3}")
rug(bd33.india$bdfi33_v_1)
```




Here there are 117 records in 20 profiles. These profiles can be found in the profiles database.

### Mapping soil properties

The location of the profiles is given in the "wosis_latest_profiles" table, whereas the soil properties are in separate tables, for example "ms_wosis_latest_bdfi33" for the bulk density. Both have a profile_id field. So to map a property, the profile ID must be used for a left join, to add the coordinates to the properties table.

First, select the bulk densities for a specified depth, here 30 cm:

```
bd30cm <- bd33.india %>%
  select(profile_id, upper_dept, lower_dept, bdfi33_v_1) %>%
  filter((lower_dept > 30) ) %>%
  group_by(profile_id) %>%    # may be more than one layer deeper than this
  arrange(upper_dept) %>%     # select the shallowest
  filter(row_number() == 1) %>%
  ungroup()
glimpse(bd30cm)
```

```
## Rows: 19
## Columns: 5
## $ profile_id <int> 67782, 67295, 170994, 67281, 66492, 170997, 67280,
```

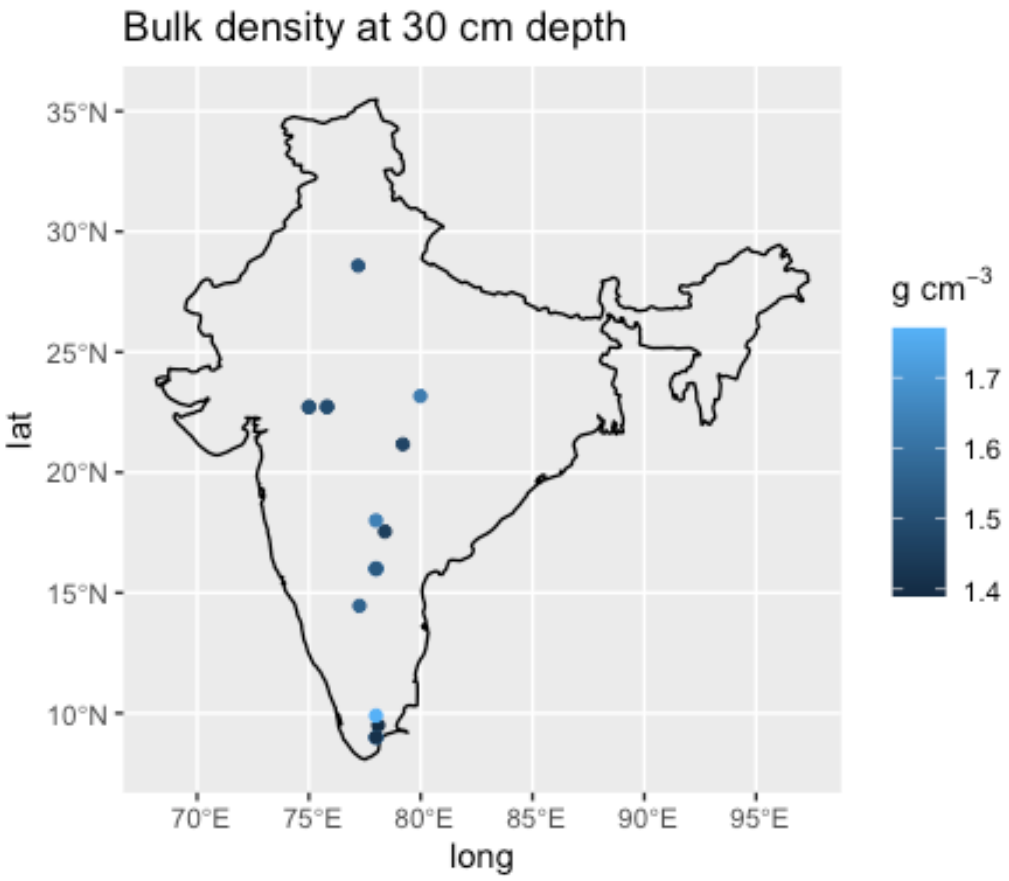
```
170992, …
## $ upper_dept <int> 11, 12, 12, 15, 18, 19, 20, 20, 20, 20, 22, 23, 23, 23,
26,…
## $ lower_dept <int> 38, 38, 38, 36, 46, 55, 42, 39, 39, 34, 43, 42, 34, 36,
48,…
## $ bdfi33_v_1 <dbl> 1.42, 1.72, 1.65, 1.63, 1.45, 1.55, 1.51, 1.64, 1.60,
1.39,…
## $ geometry   <POINT [°]> POINT (75 22.71667), POINT (78 16), POINT (78
16), PO…
```

Now join with the profile information. Note that the geometry information is duplicated, we only need it in the first-named object to be joined.

```
india.pts <- profiles.india %>%
  select(profile_id, geometry)
bd30cm.j <-
  left_join(india.pts, st_drop_geometry(bd30cm), by="profile_id")
print(bd30cm.j)

## Simple feature collection with 199 features and 4 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 69.8 ymin: 8.483333 xmax: 94.05 ymax: 32
## Geodetic CRS:  WGS 84
## First 10 features:
##    profile_id upper_dept lower_dept bdfi33_v_1                   geometry
## 1      622261         NA         NA         NA POINT (86.77038 20.38793)
## 2      622883         NA         NA         NA POINT (86.73617 20.39427)
## 3      623762         NA         NA         NA POINT (76.83334 10.78333)
## 4      622603         NA         NA         NA POINT (88.33126 21.66687)
## 5      622649         NA         NA         NA POINT (88.95738 21.68509)
## 6      623297         NA         NA         NA  POINT (88.84857 21.8277)
## 7      623769         NA         NA         NA    POINT (80.75 26.66667)
## 8      623283         NA         NA         NA POINT (88.19537 21.70742)
## 9      622421         NA         NA         NA POINT (79.79479 11.43068)
## 10     622650         NA         NA         NA POINT (88.80019 22.09143)
```

Now we can display the map, using the `geom_sf` geometry primitive in `ggplot`. The country boundary is obtained from the `mapdata` package.

```
IN <- map_data(map = 'world',
               region = 'India')
# remove named subregions, i.e., the islands
ix <- which(is.na(IN$subregion))
IN <- IN[ix,]
ggplot() +
  geom_path(aes(x = long, y = lat), data=IN) +
  geom_sf(data = bd30cm, aes(col = bdfi33_v_1)) +
  labs(title = "Bulk density at 30 cm depth", col = expression(paste(g," ",
cm)^-3))
```




Obviously, there is not much information for this soil property for India.

## CSV files

Another output format for `ogr2ogr` is the CSV 'comma-separated values' plain-text file. These typically have one header row giving the name of each column (field), and then one row (case, tuple) per observation

### WoSIS profiles as CSV files

WoSIS WFS is available as CSV files, using `f="CSV"` argument to `ogr2ogr`.

For example, read the profile information for the 2°×2° tile in central Europe.

```
wosis.dir.name.ceu <- "./wosis_latest/central_europe"
if (!file.exists(wosis.dir.name.ceu)) dir.create(wosis.dir.name.ceu)
src.layer.name <- "ms:wosis_latest_profiles"
dst.layer.name <- "wosis_latest_profiles_ceu"
(dst.target.name <- paste0(wosis.dir.name.ceu,"/",dst.layer.name,".csv"))

## [1] "./wosis_latest/central_europe/wosis_latest_profiles_ceu.csv"

if (!file.exists(dst.target.name)) {
gdalUtilities::ogr2ogr(src=wfs,
```

```
        dst=dst.target.name,
        layer=src.layer.name,
        f="CSV",
        spat=c(6, 48, 8, 50),
        overwrite=TRUE)
}
round(file.info(dst.target.name)$size/1024,1)
```

```
## [1] 73.1
```

This file is about 73 Kb.

### Read imported CSV-formatted profiles into R

The `read.csv` function reads from a CSV file into an R `data.frame`.

Read the profiles (sites) from central Europe:

```
layer.name <- "wosis_latest_profiles_ceu"
system.time(
  profiles.ceu <- read.csv(paste0(wosis.dir.name.ceu, "/",layer.name,".csv"),
                stringsAsFactors = FALSE)
)
```

```
##    user  system elapsed
##   0.006   0.000   0.008
```

```
names(profiles.ceu)
```

```
##  [1] "gml_id"                          "profile_id"
##  [3] "dataset_id"                      "continent"
##  [5] "country_id"                      "country_name"
##  [7] "geom_accuracy"                   "latitude"
##  [9] "longitude"                       "dsds"
## [11] "cfao_version"                    "cfao_major_group_code"
## [13] "cfao_major_group"                "cfao_soil_unit_code"
## [15] "cfao_soil_unit"                  "cwrb_version"
## [17] "cwrb_reference_soil_group_code" "cwrb_reference_soil_group"
## [19] "cwrb_prefix_qualifier"           "cwrb_suffix_qualifier"
## [21] "cstx_version"                    "cstx_order_name"
## [23] "cstx_suborder"                   "cstx_great_group"
## [25] "cstx_subgroup"                   "bdfi33"
## [27] "bdfiad"                          "bdfifm"
## [29] "bdfiod"                          "bdws33"
## [31] "bdwsad"                          "bdwsfm"
## [33] "bdwsod"                          "cecph7"
## [35] "cecph8"                          "cfgr"
## [37] "cfvo"                            "clay"
## [39] "ecec"                            "elco20"
## [41] "elco25"                          "elco50"
## [43] "elcosp"                          "nitkjd"
## [45] "orgc"                            "orgm"
```

```
## [47] "phaq"                          "phba"
## [49] "phca"                          "phetb1"
## [51] "phetm3"                        "phetol"
## [53] "phkc"                          "phnf"
## [55] "phprtn"                        "phptot"
## [57] "phpwsl"                        "sand"
## [59] "silt"                          "tceq"
## [61] "totc"                          "wg0006"
## [63] "wg0010"                        "wg0033"
## [65] "wg0100"                        "wg0200"
## [67] "wg0500"                        "wg1500"
## [69] "wv0006"                        "wv0010"
## [71] "wv0033"                        "wv0100"
## [73] "wv0200"                        "wv0500"
## [75] "wv1500"
```

### WoSIS layers as CSV files

Get the the bulk density for the layers in these profiles. Note that this query can not be limited by coordinates, since they are not included in this table. So we get all the layers and then limit to the profiles of interest.

```
src.layer.name <- "wosis_latest_bdfi33"
dst.layer.name <- "wosis_latest_bdfi33"
(dst.target.name <- paste0(wosis.dir.name,"/",dst.layer.name,".csv"))

## [1] "./wosis_latest/wosis_latest_bdfi33.csv"

if (!file.exists(dst.target.name)) {
gdalUtilities::ogr2ogr(src=wfs,
        dst=dst.target.name,
        layer=src.layer.name,
        f="CSV",
        overwrite=TRUE)
}
file.info(dst.target.name)$size/1024/1024

## [1] 29.34167
```

This is a very large file, about 29 Mb.

### Read imported layers into R

The bulk density per-layer.

```
layer.name <- "wosis_latest_bdfi33"
system.time(
  bd33.pts <- read.csv(file=paste0(wosis.dir.name,"/",layer.name,".csv"),
                stringsAsFactors = FALSE)
)
```

```
##    user  system elapsed
##   0.479   0.023   0.504
```

```
dim(bd33.pts)
```

```
## [1] 21599    15
```

```
names(bd33.pts)
```

```
##  [1] "gml_id"              "profile_id"          "profile_layer_id"
##  [4] "country_name"        "upper_depth"         "lower_depth"
##  [7] "layer_name"          "litter"              "bdfi33_value"
## [10] "bdfi33_value_avg"    "bdfi33_method"       "bdfi33_date"
## [13] "bdfi33_dataset_id"   "bdfi33_profile_code" "bdfi33_licence"
```

Notice that here we have the complete field names, not truncated as they were in the shapefiles. These can now be processed as above, i.e., the shapefiles.

## Spatial objects

Once the WoSIS profiles have been imported to R they can be converted to a spatial object in the `sf` package, by specifying the coördinates and the Coordinate Reference System (CRS), which we know from the WoSIS metadata is EPSG code 4326 (geographic coordinates on the WGS84 datum).

For example, here are the central Europe profiles as a spatial object:

```
profiles.ceu <- st_as_sf(profiles.ceu,
                         coords = c("longitude", "latitude"),
                         crs = 4326)
class(profiles.ceu)
```

```
## [1] "sf"         "data.frame"
```

Review some site information, e.g., the WRB Reference Soil Groups:

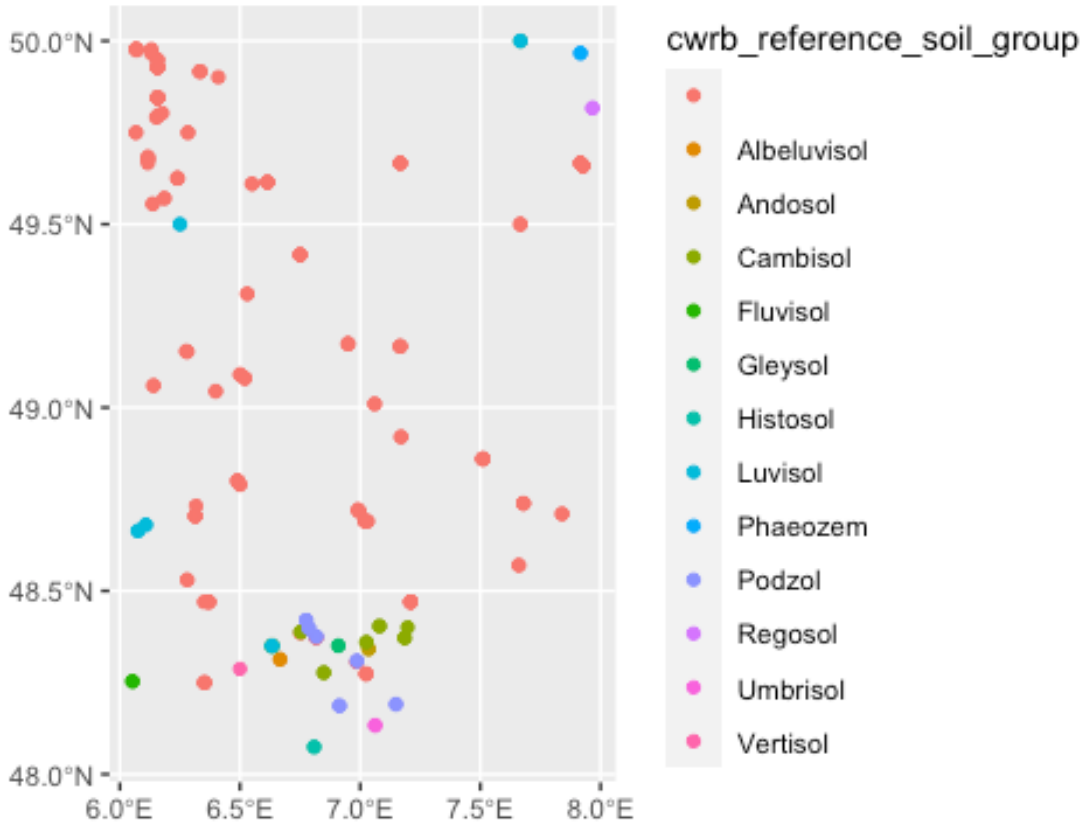
```
table(profiles.ceu$cwrb_reference_soil_group)
```

```
##
##             Albeluvisol     Andosol    Cambisol    Fluvisol     Gleysol
##         169           2           2          14           2           4
##    Histosol     Luvisol    Phaeozem      Podzol     Regosol    Umbrisol
##           2          10           2          12           2           2
##    Vertisol
##           1
```

Note that most of these profiles do not have a WRB classification.

Display a map of the profiles with their classification:

```
ggplot(data=profiles.ceu) +
  aes(col=cwrb_reference_soil_group) +
  geom_sf()
```




## Working with WoSIS as a `SoilProfileCollection`

The `aqp` "Algorithms for Quantitive Pedology" package (Beaudette, Roudier, and O'Geen 2013) defines data structures and functions specific to soil profile data, i.e., with site and linked layer information.

Load the package, and the `data.table` package on which it depends:

```
require(data.table)
```

```
## Loading required package: data.table
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
require(aqp)            # Algorithms for Quantitative Pedology

## Loading required package: aqp

## This is aqp 1.42

##
## Attaching package: 'aqp'

## The following objects are masked from 'package:dplyr':
##
##     combine, slice
```

Convert the bulk density `sf` object to a `SoilProfileCollection`, a data type defined in `aqp`. This data type has separate structures for the site (profile) and horizons. It does not have geometry, so we use `st_drop_geometry` to convert the `sf` object to a data frame.

The `aqp::depths` function initializes the SoilProfileCollection object. The formula has the field name of the profile on the left, and the the field names of the horizon boundaries on the right. These fields are in the WoSIS layer.

```
ds.aqp <- st_drop_geometry(bd33)
depths(ds.aqp) <- profile_id ~ upper_dept + lower_dept

## converting profile IDs from integer to character

is(ds.aqp)

## [1] "SoilProfileCollection"

slotNames(ds.aqp)

## [1] "idcol"        "hzidcol"      "depthcols"    "metadata"     "horizons"
## [6] "site"         "sp"           "diagnostic"   "restrictions"

str(ds.aqp@site)

## 'data.frame':    12888 obs. of  1 variable:
##  $ profile_id: chr  "135643" "135648" "135653" "135658" ...

str(ds.aqp@horizons)

## 'data.frame':    78058 obs. of  16 variables:
##  $ gml_id    : chr  "wosis_latest_bdfi33.605251"
"wosis_latest_bdfi33.605252" "wosis_latest_bdfi33.605253"
"wosis_latest_bdfi33.605254" ...
##  $ profile_id: chr  "135643" "135643" "135643" "135643" ...
##  $ profile_la: int  605251 605252 605253 605254 605255 605256 605262
605263 605264 605265 ...
##  $ country_na: chr  "United States of America" "United States of America"
"United States of America" "United States of America" ...
##  $ upper_dept: int  5 12 27 64 86 111 6 13 29 46 ...
##  $ lower_dept: int  12 27 64 86 111 157 13 29 46 77 ...
```

```
##  $ layer_name: chr  "BA" "Bw1" "Bw2" "2Cd1" ...
##  $ litter    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ bdfi33_val: chr  "{1:1.65,2:1.65,3:1.34,4:1.34,5:1.39,6:1.39}"
"{1:1.46,2:1.46,3:1.72,4:1.72,5:1.44,6:1.44}"
"{1:1.47,2:1.47,3:1.49,4:1.49,5:1.73,6:1.73}"
"{1:1.91,2:1.91,3:1.68,4:1.68,5:1.71,6:1.71}" ...
##  $ bdfi33_v_1: num  1.46 1.54 1.56 1.77 1.88 1.82 1.24 1.53 1.64 1.7 ...
##  $ bdfi33_met: chr  "{\"1:calculation = not specified, corrections = not
specified, measurement condition = equilibrated at 33 kPa ("| __truncated__
"{\"1:calculation = not specified, corrections = not specified, measurement
condition = equilibrated at 33 kPa ("| __truncated__ "{\"1:calculation = not
specified, corrections = not specified, measurement condition = equilibrated
at 33 kPa ("| __truncated__ "{\"1:calculation = not specified, corrections =
not specified, measurement condition = equilibrated at 33 kPa ("|
__truncated__ ...
##  $ bdfi33_dat: chr  "{1:1999-7-7,2:1999-7-7,3:1999-7-7,4:1999-7-7,5:1999-
7-7,6:1999-7-7}" "{1:1999-7-7,2:1999-7-7,3:1999-7-7,4:1999-7-7,5:1999-7-
7,6:1999-7-7}" "{1:1999-7-7,2:1999-7-7,3:1999-7-7,4:1999-7-7,5:1999-7-
7,6:1999-7-7}" "{1:1999-7-7,2:1999-7-7,3:1999-7-7,4:1999-7-7,5:1999-7-
7,6:1999-7-7}" ...
##  $ bdfi33_d_1: chr  "US-NCSS" "US-NCSS" "US-NCSS" "US-NCSS" ...
##  $ bdfi33_pro: chr  "00P0001" "00P0001" "00P0001" "00P0001" ...
##  $ bdfi33_lic: chr  "U.S. Public Domain
http://www.usa.gov/publicdomain/label/1.0/" "U.S. Public Domain
http://www.usa.gov/publicdomain/label/1.0/" "U.S. Public Domain
http://www.usa.gov/publicdomain/label/1.0/" "U.S. Public Domain
http://www.usa.gov/publicdomain/label/1.0/" ...
##  $ hzID      : chr  "1" "2" "3" "4" ...
```

```
head(ds.aqp@site)
```

```
##   profile_id
## 1     135643
## 2     135648
## 3     135653
## 4     135658
## 5     135663
## 6     135680
```

```
head(ds.aqp@horizons[c(2,5,6,7,9)],12)
```

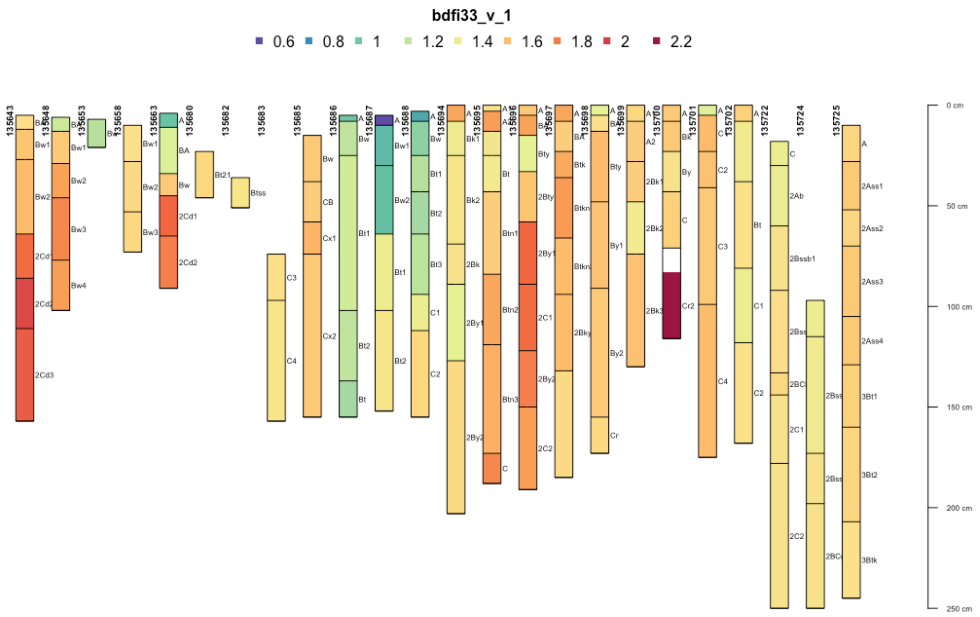
```
##    profile_id upper_dept lower_dept layer_name
## 1      135643          5         12         BA
## 2      135643         12         27        Bw1
## 3      135643         27         64        Bw2
## 4      135643         64         86       2Cd1
## 5      135643         86        111       2Cd2
## 6      135643        111        157       2Cd3
## 7      135648          6         13         BA
## 8      135648         13         29        Bw1
## 9      135648         29         46        Bw2
```

```
## 10     135648          46          77         Bw3
## 11     135648          77         102         Bw4
## 12     135653           7          21          Bw
##                                          bdfi33_val
## 1  {1:1.65,2:1.65,3:1.34,4:1.34,5:1.39,6:1.39}
## 2  {1:1.46,2:1.46,3:1.72,4:1.72,5:1.44,6:1.44}
## 3  {1:1.47,2:1.47,3:1.49,4:1.49,5:1.73,6:1.73}
## 4  {1:1.91,2:1.91,3:1.68,4:1.68,5:1.71,6:1.71}
## 5  {1:1.83,2:1.83,3:2.01,4:2.01,5:1.81,6:1.81}
## 6  {1:1.74,2:1.74,3:1.76,4:1.76,5:1.97,6:1.97}
## 7  {1:1.39,2:1.39,3:1.14,4:1.14,5:1.18,6:1.18}
## 8  {1:1.47,2:1.47,3:1.69,4:1.69,5:1.44,6:1.44}
## 9  {1:1.56,2:1.56,3:1.58,4:1.58,5:1.78,6:1.78}
## 10 {1:1.83,2:1.83,3:1.62,4:1.62,5:1.64,6:1.64}
## 11 {1:1.59,2:1.59,3:1.78,4:1.78,5:1.58,6:1.58}
## 12 {1:1.10,2:1.10,3:1.12,4:1.12,5:1.33,6:1.33}
```

Note how the horizons have been grouped into sites, in the `@site` slot, and the per-horizon (by depth) values are in the `@horizons` slot. Here we have 78058 horizons in 12888 profiles.

Now this `SoilProfileCollection` can be used for many `aqp` functions. For example, here is the depth distribution of average bulk density of the components for the first 24 listed profiles, labelled by genetic horizon:

```
plotSPC(ds.aqp[1:24,], name="layer_name", color='bdfi33_v_1')
```




Several layers in this set of profiles are missing bulk density.

## References

Beaudette, D. E., P. Roudier, and A. T. O'Geen. 2013. "Algorithms for Quantitative Pedology: A Toolkit for Soil Scientists." *Computers & Geosciences* 52 (March): 258–68. https://doi.org/10.1016/j.cageo.2012.10.020.