Soil Sensing: Measuring & Mapping soil properties

Titia Mulder

May 24, 2018

This tutorial consists of two parts. The first part shows how to estimate soil properties using a spectral library containing preprocessed VNIR-SWIR spectral data measured under laboratory conditions. The second part shows how satellite data can be used for mapping soil properties within a regional study area.

The processing steps are illustrated with a sample dataset from Morocco. The regional case study is located in Northern Morocco, centred at around 34.0N, -4.5W and covers an area of 15,000 km2. While the Rif mountains, an area of highlands, form the northern border, the Anti-Atlas mountain range is the southern border with areas of plateaus and intermountain valleys in between. Elevation ranges between +4 and +2350 m a.s.l. The climate is typically warm temperate with dry and hot summers. The main land use is dominated by a mosaic of vegetation and croplands, bare areas, sparse vegetation and open evergreen forest. For more details about the data collection see <https://doi.org/10.1016/j.jag.2012.07.004>.

# 0 Setting up an R session

Before starting scripting, it is recommended to empty the computer memory, clean up the workspace and load the required libraries. Note, you can download the *ithir* package either running the line: install.packages(“ithir”, repos=“<http://R-Forge.R-project.org>”) or download the package here <https://r-forge.r-project.org/R/?group_id=2000> and install the package and dependencies afterwards.

# empty memory and workspace  
gc()  
rm(list=ls())  
  
# load libraries  
require(sp)  
## Warning: package 'sp' was built under R version 3.4.2  
require(raster)  
## Warning: package 'raster' was built under R version 3.4.2  
require(rgdal)  
## Warning: package 'rgdal' was built under R version 3.4.2  
require(pls)  
## Warning: package 'pls' was built under R version 3.4.4  
require(ithir)  
require(MASS)  
## Warning: package 'MASS' was built under R version 3.4.2

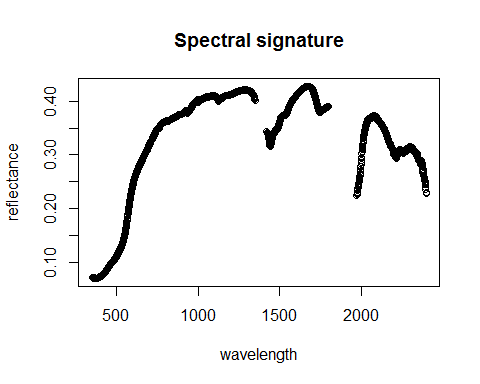
# 1 Estimating soil properties from a spectral library using Partial Least Squares Regression (PLSR)

A small dataset has been prepared for you. There are 64 samples from the study area which were analysed in the lab for pH, Soil Organic Matter (SOM%), Cation Exchange Capacity (CEC) and texture (clay, silt, sand (%)). In addition, laboratory spectral measurement were taken with an ASD Fieldspectrometer, covering the wavelength range 350-2500 micrometer. The spectral data were first smoothed using a Savitzky-Golay smoothing filter and next remaining noisy or irrelevant wavelength bands were removed. Consequently, the data ranges up to 2400 micrometer, excluding the wavelenght ranges with strong water absorption features. When you plot an example spectra you will see that the plot is not a continuous line.

## 1.1 Read the soil property point data and plot spectra

First, the whole dataset is loaded. Next, one object is created containing the soil site and soil data *sample\_data* and another object is created containing the spectral data *spectral\_data*. In order to plot the spectral data, also called a spectral signature, another object is created containing the wavelengths correponding to the measurements *wl*.

# load point data  
point <- read.csv('./point\_data.csv')  
sample\_data <- point[,c(1:10)]  
names(sample\_data) <- c("Site","Date","East","North", "pH\_KCl", "SOM", "CEC","Clay","Silt","Sand")   
  
spectral\_data <- point[,c(11:1823)]  
spectral\_data <- spectral\_data[complete.cases(spectral\_data),]  
wl <- read.csv('./wavelength.csv', header=T)  
  
plot(wl$x, spectral\_data[1,], main='Spectral signature',xlab='wavelength', ylab='reflectance', type='p')



## 1.2 How to fit a simple PLSR model

In this tutorial, a simple PLSR regression model is being used for the prediction of soil properties from a spectral libary. The basic idea, is that we fit a regression model between a soil property, e.g. SOM and the spectral data available. However, we have about 1823 spectral bands and thus 1823 predictors while we only have 64 soil samples. In order to reduce the dimensionality of the spectral data and obtain fewer predictors for the regression model, we apply a principle component analysis and use only those components that capture the majortiy of the spectral data variability. Typically, this number ranges between 4 and 15 for our dataset. This way, we can make use of the variability provided by the spectral data while reducing the number of predictor variables. The details of PLSR are described in the documentation ‘pls\_package\_paper.pdf’, which is a supporting paper for the package *pls* which we use for the PLSR modelling. Feel free to extent the models as given in the example code. Alternatively, machine-learning could be use to derive the statistical model, examples include Random Forest or Cubist or multivariate PLSR models with sample optimization for model calibration. However, this is outside the scope of this tutorial.

Inspect the function *plsr* and the different settings. Note the ncomp, validation and method which are choosen in the example code, feel free to change them and inspect the changes in output. The output can be assessed by the model summary. Here, you see how much of the variability is captured with each additional component and the amount of variability that is explained for the training data. In addition, the validation and prediction plots provide quickly insight in the model performance.

test\_plsr <- plsr(sample\_data$pH\_KCl ~ ., ncomp=15,data=spectral\_data, validation="LOO",  
 method="oscorespls",na.action=na.exclude)  
summary(test\_plsr)  
  
plot(test\_plsr, plottype='validation')  
plot(test\_plsr, plottype='prediction')  
plot(test\_plsr, plottype='correlation')  
plot(test\_plsr, plottype='biplot')

The example given relies on 15 components. When you inspect the validation plot, you may notice that the RMSEP increases after a certain additional component. For now, visually inspect the plot and configure the model is such a way that you obtain a low RMSEP while maximizing the number of components. **Try to fit several models for different soil properties and select one model to continue the tutorial with.**

Finally, derive several *Goodness of Fit* (GOOF) - metrics using the code below and inspect the output. Make sure you set the correct references to the data that you obtain from your final model. The example below refers to the model containing 10 components. Adjust your settings accordingly.

goof\_plsr <- goof(sample\_data$pH\_KCl, test\_plsr$fitted.values[,,10], type='spec')

## 1.3 How to estimate soil properties using a spectral library

In the previous section, you have derived a statistical model which can also be used for the prediction of unknown samples. This is ideally when you have a large spectral database available which provids you a robust prediction model. Now, this model can be used for the prediction of soil properties using only spectral data, meaning that you would no longer need expensive and time consuming chemical and physical lab analysis.

In this section, it is demonstrated how you can derive soil property information from a calibration model for unknown samples. However, the results may not be as reliable as we want since the test dataset is too small for the calibration of a robust prediction model.

First we need to create a dataset which can be used for the calibration of the PLSR model and another dataset which can be used for the prediction of soil properties of unknown samples.

# create a random subset of calibration and prediction data  
index <- sort(sample(nrow(sample\_data), size = 20))  
sample\_cal <- sample\_data[-index,]  
sample\_pred <- sample\_data[index,]  
  
spectra\_cal <- spectral\_data[-index,]  
spectra\_pred <- spectral\_data[index,]

Next, the PLSR model is fitted on the calibration dataset. Explore your settings and choose a soil property of interest. Make sure you always refer to the correct soil property and components that you use.

model\_plsr <- plsr(sample\_cal$CEC ~ ., ncomp=8,data=spectra\_cal, validation="LOO",  
 method="oscorespls",na.action=na.exclude)  
  
summary(model\_plsr)  
plot(model\_plsr, plottype='validation')  
plot(model\_plsr, plottype='prediction')

Now that you have a proper prediction model, you are going to use it for the prediction of unknown samples, using the function *predict*.

model\_cal <- predict(model\_plsr, spectra\_cal, comps=8)  
model\_pred <- predict(model\_plsr, spectra\_pred, comps=8)  
plot(sample\_pred$CEC, model\_pred[,1], main='Observed vs Predicted new values')

Typically, the prediction is less accurate since the new data was not included in the calibration dataset. Therefore, it is crucial to generate *Goodness of Fit* (GOOF) Metrics and asses the differences between the calibration and prediction performance. See the code below to generate a table with the metrics.

goof\_cal <- goof(sample\_cal$CEC, model\_plsr$fitted.values[,,8], type='spec')  
goof\_pred <- goof(sample\_pred$CEC, model\_pred[,1], type='spec')   
  
goof\_final <- rbind(goof\_cal, goof\_pred)  
goof\_final$dataset <- c("Calibration", 'Validation')

The GOOF metrics show that indeed the prediction was not (or less) robust in this case. However, as outlined before, the test case is actually too small to be used for this part of the tutorial. This was the last step, feel free in the future to explore other methods in a similar way. Alternatives to the PLSR are e.g. the Random Forest or Cubist models. Random Forest modelling will be discussed later this week. Another option is to use spectral data from other spectral regions, such as the Mid-Infrared, or use only parts of the spectrum that are relevant for the soil property of interest (e.g. mineralogy has important absortion features in the SWIR)

# 2 The power of satellite data for mapping soil properties

The second part of this tutorial shows how you can derive relevant soil information for Digital Soil Mapping (DSM) from satellite data. Typically in DSM, satellite-derived information about the greeness of vegetation (NDVI), soil moisture content or climate are being used, among others. However, this tutorial will focus on data that is related to parent material and the mineralogy of the study area. You will see that for the Moroccan dataset, a substantial part of the soil variability can already be explained by one satellite product only.

In this tutorial, you will model the statistical relation between the soil sample data and satellite data using stepwise multiple linear regression.

## 2.1 ASTER satellite data and the ASTER GEOSCIENCE product

The satellite data you will work with is Advanced Spaceborne Thermal Emission and Reflectance radiometer (ASTER) imagery. Compared to other multispectral sensors offering a global coverage, ASTER has a relative high spatial and spectral resolution and favourable positions of the spectral bands for the retrieval of soil properties. ASTER has been proven to be a useful data source for the retrieval of various soil properties and NDVI. The ASTER sensor records data in 15 spectral bands of which 4 in VNIR with a resolution of 15 m, 6 in the SWIR with a resolution of 30 m and 5 in the TIR region with a resolution of 90 m. A mosaic of these images was taken in the same season over the years 2005 and 2007. There are 14 different bands with information about the spectral reflectance at different wavelengths. More information about the satellite data can be found here: <https://lpdaac.usgs.gov/dataset_discovery/aster/aster_products_table/ast14dmo>

Moreover, the ASTER GEOSCIENCE product was prepared for you to work with in this tutorial. The ASTER GEOSCIENCE product consists of several spectral indices which are supposed to relate to various soil properties. You can read more about the method to calculate the ASTER GEOSCIENCE indices in the document *ASTER\_Geoscience\_product\_notes.pdf* and the code is provided in the R-script *R\_geoscience.R*.

Finally, it must be noted that the satellite data used here is typically suitable for DSM in arid and semi-arid regions with sparse vegetation cover. An exhaustive set of spatial covariates will be used later in the week, now it is only demonstrated how spectral data can be usefull for DSM.

## 2.2 Read the soil property point data

Start with loading the same soil dataset as used in part 1 of this tutorial. However, this time only use the site and soil information.

In order to link the site coordinates of the points to the satellite data, a SpatialPoint object is created *point\_xy*, containing the coordinates and the associated projection (WGS 84) system.

# load point data  
point <- read.csv('./point\_data.csv')  
sample\_data <- point[,c(1:10)]  
names(sample\_data) <- c("Site","Date","East","North", "pH\_KCl", "SOM", "CEC","Clay","Silt","Sand")   
point\_xy <- SpatialPoints(point[,c(3,4)], CRS("+init=epsg:4326"))

## 2.3 Read the satellite products

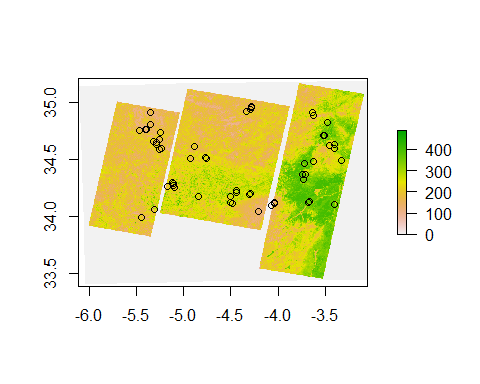
Load the ASTER satellite data and GEOSCIENCE product as a raster object. These are both geotif-files, containing several bands. In the code below, the data is loaded using the function ‘stack’ and the band names are given to each layer.

For the ASTER data, object *aster* with the names refer to the band numbers, e.g. band 1 is named ‘b1’.

For the GEOSCIENCE data, object *aster\_geo* the names refer to the indices that were derived (see the documentation). For example, GV refers to Greeness Vegetation, kao\_index refers to the kaolinit index and FI to Ferrous Iron. Furthermore, \_comp refers to composition and \_cont refers to content.

Before you continue, make sure that the points allign with the raster. It may happen that something went wrong with the settings of your coordinate system. Simply plot the raster and overlay the points, as detailed in the code below.

# Load ASTER satellite data  
aster <- stack('./aster.tif')  
names(aster) <- c('b1','b2','b3','b4','b5','b6','b7','b8','b9','b10','b11','b12','b13','b14')  
  
# Load ASTER GEOSCIENCE product  
aster\_geo <- stack('./aster\_geo.tif')  
names(aster\_geo) <- c('GV','FO\_cont', 'FO\_comp','FI\_index', 'OpI','AlOH\_index',   
 'AlOH\_comp', 'kao\_index', 'FeOh\_cont','MgOH\_cont', 'MgOH\_comp', 'mgoh\_fe' )  
  
# check allignment raster and point data  
plot(aster[[1]])  
points(point\_xy)



## 2.4 Prepare a dataset for stepwise multiple linear regression

Next, the satellite information can be extracted to the *SpatialPoints* object, following the code below. The result is a dataframe containing the raster data at the sample points.

These dataframes are merged together in one dataframe *data\_all*, containing all the data needed to fit a statistical model.

# extract data to points  
point\_aster <- extract(aster, point\_xy, df=TRUE)  
point\_aster\_geo <- extract(aster\_geo, point\_xy, df=TRUE)  
  
  
# create one dataset RS  
data\_rs <- data.frame(point\_aster, point\_aster\_geo[,-c(1)])  
  
# add data to soil information  
data\_all <- data.frame(sample\_data, data\_rs[,-c(1)])  
data\_all <- data\_all[complete.cases(data\_all),]

## 2.5 How to fit a stepwise multiple linear regression model

Before fitting models it is helpful to explore the dataset and get a better feeling for exisiting correlation within the data, e.g. the relations between soil properties and spectral data. One way is to calculate the correlation matrix and assess the correlations, see the code below. Another way is a principle component analysis, however this is outside the scope of this tutorial.

Make note of the highest correlations and try to explain them. For example, *Greeness of Vegetation* may have a high correlation with *Soil Organic Matter*, while the *MgOH indices* correlate better to *pH* and *CEC*.

# derive the correlation between variables  
correl <- cor(as.matrix(data\_all[,-c(1:4)]))

Next, the model can be calibrated. In pricipal, one uses a stepwise multiple linear regression model (stepwise MLR) to remove predictors (or covariate) from the MLR model which do not add to the overall explained variance and model performance. This usually gives a more robust model and avoids overfitting. In order to perform stepwise MLR, first a MLR model is fitted using all covariates. Next, iteratively (stepwise) covariates are either added or removed from the model until a certain criteria is met (e.g. the AIC).

See the code below and check the *stepAIC* function in the help, study the model arameter *direction* and change if deemed necessary.

The first model will only use the satellite aster bands. After you fitted the stepwise MLR, use the function *summary()* to see how well the model is capable to explain the variability of the soil property of interest and the ‘anova’ output to inspect how the model changed with everyt iteration (step).

Try several soil properties and choose your soil property of interest for the rest of this tutorial.

**How many and which variables were included in the final model, does that make sense to you? Do these variables also have high correlations with the soil property of interest?**

# model for SOM with aster bands only  
lm\_model\_aster\_0 <- lm(SOM ~ b1+b2+b3+b4+b5+b6+b7+b8+b9+b10+b11+b12+b13+b14, data=data\_all )  
lm\_model\_aster <- stepAIC(lm\_model\_aster\_0, direction='both')  
  
summary(lm\_model\_aster)  
lm\_model\_aster$anova

Similar, fit a stepwise MLR-model using only the GEOSCIENCE indices and check how well the model is capable to explain the variability of the soil property of interest.

**How many and which variables were included in the final model, does that make sense to you?**

# model for SOM with aster geoscience products only  
lm\_model\_geo\_0 <- lm(SOM ~ GV+FO\_cont+FO\_comp+FI\_index+OpI+AlOH\_index+AlOH\_comp+kao\_index+FeOh\_cont+MgOH\_cont+  
 MgOH\_comp+mgoh\_fe, data=data\_all )  
lm\_model\_geo <- stepAIC(lm\_model\_geo\_0, direction='both')  
  
summary(lm\_model\_geo)

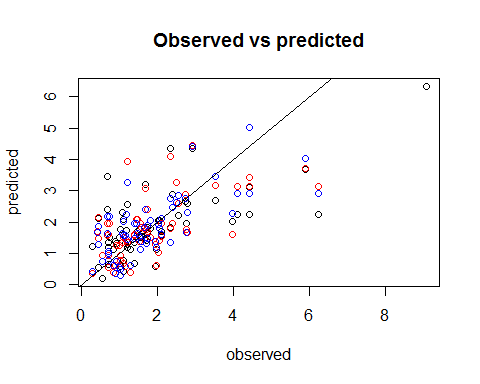
Similar, fit a stepwise MLR-model using all available covariates (combining all ASTER bands and GEOSCIENCE indices) and see if the model was better capable to explain the variability of the soil property of interest.

**How many and which variables were included in the final model, does that make sense to you?**

# model for SOM with all aster data  
lm\_model\_all\_0 <- lm(SOM ~ b1+b2+b3+b4+b5+b6+b7+b8+b9+b10+b11+b12+b13+b14+  
 GV+FO\_cont+FO\_comp+FI\_index+OpI+AlOH\_index+AlOH\_comp+kao\_index+FeOh\_cont+MgOH\_cont+MgOH\_comp+mgoh\_fe, data=data\_all )  
lm\_model\_all <- stepAIC(lm\_model\_all\_0, direction='both')  
  
summary(lm\_model\_all)

The summaries of the models do provide little information about the model performance. Therefore, it is better to make a plot with the observed versus the predicted values of the different models and further evaluate the model’s performance.

# plot observed versus predicted  
plot(data\_all$SOM, lm\_model\_aster$fitted.values, col='black', main='Observed vs predicted',  
 xlab='observed',ylab='predicted')  
points(data\_all$SOM, lm\_model\_geo$fitted.values, col= 'red')  
points(data\_all$SOM, lm\_model\_all$fitted.values, col='blue')  
abline(0,1)



In order to evaluate the model’s performance, derive several goodness of fit (GOOF) - metrics using the code below and inspect the output. Make sure you set the correct references to the data that you obtain from your final model.

**Which model for which soil property worked best, what can you conclude about the use of satellite data for estimating soil properties?**

The model that you now derived was based on all available data and so the GOOF metrics only indicate the goodness of the calibration model. Ideally, one uses a seperate calibration and validation dataset, similar as outlined in section 1.3. If you have time, you can rerun the code for a calibration and validation set, fit the models and derive the GOOF metrics.

############## Goodness of Fit calibration model  
  
goof\_aster <- goof(data\_all$SOM, lm\_model\_aster$fitted.values, type='spec')   
goof\_geo <- goof(data\_all$SOM, lm\_model\_geo$fitted.values, type='spec')   
goof\_all <- goof(data\_all$SOM, lm\_model\_all$fitted.values, type='spec')   
  
goof\_final <- rbind(goof\_aster, goof\_geo, goof\_all)  
goof\_final$data <- c("aster", 'geo', 'all')

## 2.6 How to apply Stepwise MLR to the full spatial extent

Finally, the calibrated model can be used to predict the soil property of interest at the full extent of the study area. Below you find the example code using the model calibrated on the satellite data (object aster). Feel free to explore your own final model. If that model relies on all the covariate layers, make sure they are compiled in one stack object (e.g. *raster\_all <- stack(aster, geo*).

map\_aster <- predict(aster, model=lm\_model\_aster, progress='text',  
 filename='map\_aster\_lm.tif', format='GTiff',datatype='FLT4S', overwrite=T)  
  
plot(map\_aster)