Statistical Methods:
Prediction of Soil Parameters through Near Infrared Spectroscopy

Kazimir Menzel
kazimir.menzel@me.com

Markus Pawellek
markuspawellek@gmail.com

Abstract

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

1 Introduction

Through soil analyses or soil testing one can get certain chemical and physical information about the used soil like the concentration of soil organic carbon (SOC) or the pH-value. With these Measurements, known as soil parameters, it is possible to optimize plant growth or to assist in solving soil-related problems.

However the direct measurement of soil parameters is very costly and error-prone. Therefore methods for fast and cheap determination of these parameters play a fundamental and vital role in particular fields like agriculture, geochemistry and ecology.

At the beginning of the 1960s Karl Norris for the first time in history used Near Infrared Spectroscopy (NIRS) to predict and calculate moisture content from seed extracts through a multivariate calibration approach. His work had a huge impact in agricultural an non-agricultural fields since NIRS proved to be a significant time-saver and cheap alternative to other methods.

In the last few years NIRS applications experienced a massive growth. This would have not been possible without better computing capabilities and progress in multivariate methods as the prediction of soil parameters out of a measured soil spectrum requires a large amount of statistical computations.

2 Fundamentals

2.1 Soil Parameters

Let A be any substance in a given soil sample with volume $V \in (0,\infty)$ and let $n_A \in (0,\infty)$ be the amount of A in the sample. Then the molar concentration c_A is given by

$$c_A \coloneqq \frac{n_A}{V}$$

Now let c_0 be the molar concentration of this whole sample. We define the amount-of-substance fraction (ASF) p_A of A as

$$p_A \coloneqq 100\% \cdot \frac{c_A}{c_0}$$

In this elaboration we concentrate on three main soil parameters that are given by existing NIRS-measurements. The first two parameters p_{SOC} and p_{N} are ASFs relating to soil organic carbon (SOC) and nitrogen in the soil sample. SOC is the carbon in this sample which is bound in an organic compound. The third parameter is the pH-value that is used to specify the acidity or basicity of an aqueous solution. It is based on the concentration of hydronium ions $c_{\mathrm{H_3O^+}}$.

$$\mathrm{pH} \coloneqq -\log_{10} c_{\mathrm{H_3O^+}} = -\frac{\ln c_{\mathrm{H_3O^+}}}{\ln 10}$$

2.2 NIRS

In NIRS one uses electromagnetic waves, also known as light, with a wavelength from $780\,\mathrm{nm}$ to $3000\,\mathrm{nm}$. This area is called the near infrared region and is the most energetic one of the infrared light.

An emitted light wave with a certain wavelength λ can interact with a soil sample in three ways. It can be reflected, absorbed or transmitted. For most soil samples measuring the transmittance of light waves is not sensible because thickness of these samples varies. So the measurement of the spectrum is reduced to the reflectance since absorptance cannot be directly determined.

The amount of reflection divides again in specular and diffuse reflection. For our purpose only the diffuse part shall be considered. This is the one which penetrates the soil sample most. As a consequence, diffuse reflected light is scattered among the hemisphere and contains information about the used soil sample. For a more detailed view on this topic please refer to [Tutorial].

The reflectance $\varrho(\lambda)$ referring to a wavelength λ of a light wave of a surface is given by the amount of radiation power P_r that is reflected from a surface divided by the initial power P_0 of this light wave.

$$\varrho(\lambda) := \frac{P_\mathrm{r}}{P_0}$$

This function ϱ is then called (in our case) the near infrared spectrum of the soil sample.

Absorptance itself originates from the existence of vibrational modes in molecules. A photon with a wavelength λ can only be absorbed if the appropriate frequency f exactly matches a multiple of the transition energy of the bond or group that vibrates. This is why the spectra of soil samples are formed of overtones and combinations bands.

As a matter of fact it is because of the similarity of diffuse reflected and transmitted light that we can use the Beer-Lambert law as a relation of the attenuation of light and the properties of samples. Let $n \in \mathbb{N}$ be the count of different species in a sample and c_i be the molar concentration of the ith species for $i \in \mathbb{N}, i \leq n$. Is again λ the wavelength of the used light then there exist certain coefficients $\varepsilon_i(\lambda)$ for all $i \in \mathbb{N}, i \leq n$ so that

$$-\ln \varrho(\lambda) = \sum_{i=1}^{n} \varepsilon_i(\lambda) c_i$$

2.3 Measured Data

For the prediction of soil parameters it is inevitable one already got some soil spectra with corresponding directly measured soil parameters. We are using a dataframe made by Dr. A. Don of the Heinrich von Thünen-Institut Braunschweig.

The dataframe is made up of 533 measured soil samples. For everyone of them the in section 2.1 discussed parameters $p_\mathrm{SOC}, p_\mathrm{N}$, pH and the near infrared spectrum are given. The soil spectra consists of 319 wavelengths ranging from 1400 nm to 2672 nm by a step of 4 nm. The reflectance $\varrho(\lambda)$ of a sample at a given wavelength λ is saved as

$$-\lg \varrho(\lambda) = -\frac{\ln \varrho(\lambda)}{\ln 10}$$

Figure 1 shows six randomly chosen soil spectra in a diagram.

2.4 Statistical Model

Let $n \in \mathbb{N}$ be the soil sample count and $k \in \mathbb{N}$ with $k \leq n$ the number of measured wavelengths. ϱ_i shall be defined as soil spectrum of the ith sample for every $i \in \mathbb{N}, i \leq n$. λ_j is the jth measured wavelength for every $j \in \mathbb{N}, j \leq k$, also called predictor. Then according to section 2.3 the measured reflectance values are x_{ij} with

$$x_{ij} := -\lg \varrho_i(\lambda_j)$$

for every $i, j \in \mathbb{N}, i \leq n, j \leq k$. Through the following matrix it is possible to get an easier notation.

$$X := (x_{ij}) \in \mathbb{R}^{n \times k}$$

Let $p_i^{(\mathrm{SOC})}, p_i^{(\mathrm{N})}, \mathrm{pH}_i$ be the measured soil parameters of the ith sample for every $i \in \mathbb{N}, i \leq n$, also known as observables. Then we define the n-dimensional vectors

$$\begin{aligned} p^{(\mathrm{SOC})} &:= \left(p_i^{(\mathrm{SOC})}\right) \\ p^{(\mathrm{N})} &:= \left(p_i^{(\mathrm{N})}\right) \\ \mathrm{pH} &:= (\mathrm{pH}_i) \end{aligned}$$

The Beer-Lambert law allows us to make assumptions to the relations of soil spectra and soil parameters. In section 2.2 we saw that the logarithmized reflectance can be written as a linear combination of molar concentrations. Hence, vice versa it makes sense to assume that an ASF can be represented by a linear combination of logarithmized reflectance values.

Now let $P^{(\mathrm{SOC})}, P^{(\mathrm{N})}$ be the appropriate random variables to the soil parameters $p^{(\mathrm{SOC})}$ and $p^{(N)}$. Then with the above assumption the expected values are given by

$$\begin{aligned} \mathbb{E}\, P^{(\mathrm{SOC})} &= X\beta_k^{(\mathrm{SOC})} + \beta_0^{(\mathrm{SOC})} =: \mathbb{X}\beta^{(\mathrm{SOC})} \\ \mathbb{E}\, P^{(\mathrm{N})} &= X\beta_k^{(\mathrm{N})} + \beta_0^{(\mathrm{N})} =: \mathbb{X}\beta^{(\mathrm{N})} \end{aligned}$$

When PH is the corresponding random variable to pH we have to perform a correction because pH is a logarithmized molar concentration. It makes sense to model these with the subsequent expected value.

$$\mathbb{E}\,\mathrm{PH} = \ln(X)\beta_k^{(\mathrm{pH})} + \beta_0^{(\mathrm{pH})} =: \mathbb{X}_{\ln}\beta^{(\mathrm{pH})}$$

In physics and chemistry it is a common and error-proven method to assume a normal distribution with a certain variance for measuring errors. So with the variances $(\sigma^2)^{(\mathrm{SOC})}, (\sigma^2)^{(\mathrm{N})}, (\sigma^2)^{(\mathrm{pH})} \in (0, \infty)$ our random variables become

$$\begin{aligned} P^{(\mathrm{SOC})} &\sim \mathcal{N}\left(\mathbb{X}\beta^{(\mathrm{SOC})}, (\sigma^2)^{(\mathrm{SOC})}\mathrm{I}_n\right) \\ P^{(\mathrm{N})} &\sim \mathcal{N}\left(\mathbb{X}\beta^{(\mathrm{N})}, (\sigma^2)^{(\mathrm{N})}\mathrm{I}_n\right) \\ \mathrm{PH} &\sim \mathcal{N}\left(\mathbb{X}_{\ln}\beta^{(\mathrm{pH})}, (\sigma^2)^{(\mathrm{pH})}\mathrm{I}_n\right) \end{aligned}$$

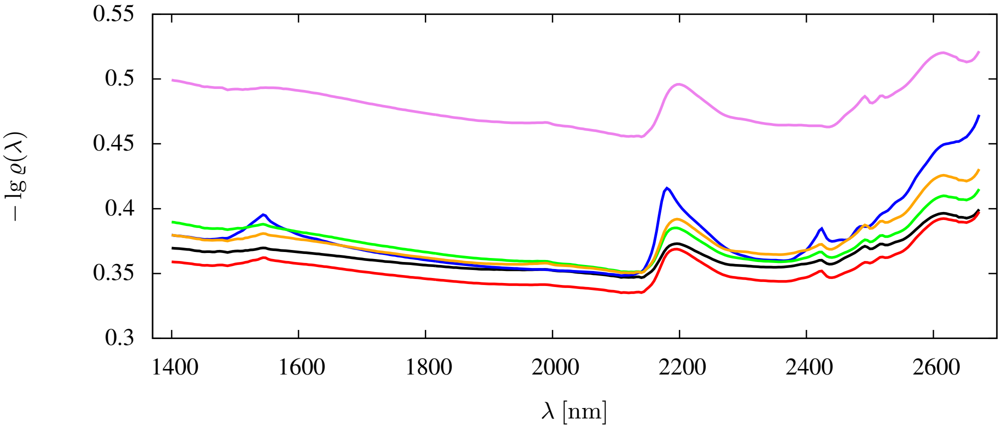

Figure 1: This figure shows near infrared soil spectra of six randomly chosen soil samples obtained from the used dataframe.

2.5 MLR

Multiple linear regression (MLR) or multivariate linear regression is a statistical method for estimating parameters that depend on linear independent variables. Let $\mathbb{X} \in \mathbb{R}^{n\times(k+1)}, n, k \in \mathbb{N}, k < n$ be the design matrix, $\sigma^2 \in (0,\infty)$ and Y be the vector of random variables with

$$Y \sim \mathcal{N}\left(\mathbb{X}\beta, \sigma^2 \mathrm{I}_n\right)$$

for a certain $\beta \in \mathbb{R}^{k+1}$ Then through the maximum-likelihood-method and a small correction we get two best unbiased estimators $\hat{\beta}, \hat{\sigma^2}$ for β and σ^2

$$\hat{\beta}(Y) = \left(\mathbb{X}^{\mathrm{T}}\mathbb{X}\right)^{-1}\mathbb{X}^{\mathrm{T}}Y$$
$$\hat{\sigma^2}(Y) = \frac{1}{n-(k+1)}\left\|Y - \mathbb{X}\hat{\beta}(Y)\right\|^2$$

Now let $y := (y_i) \in \mathbb{R}^n$ be a realization of Y. In this case we define

$$\hat{y} := \mathbb{X}\hat{\beta}(y) = \mathbb{X}\left(\mathbb{X}^{\mathrm{T}}\mathbb{X}\right)^{-1}\mathbb{X}^{\mathrm{T}}y$$
$$\hat{\sigma^2} := \hat{\sigma^2}(y)$$

For more information please refer to [Quelle:Skript,wikipedia].

2.6 Mallows' Cp

According to sections 2.4 and 2.5 at this time we are using $k+1 = 320$ predictors for our prediction model. Estimating Model Parameters with this large amount of predictors tends to overfit the measured data. So it would be sensible to choose a "good" subset of predictors

$$M \subset \{\lambda_i \mid i \in \mathbb{N}_0, i \leq k\}$$

where λ_0 stands for the defined offset. Now through M one can define a new design matrix $\mathbb{X}^{(M)}$. Applying MLR to this design matrix gives us new estimators

$$\hat{\beta}^{(M)}(Y) = \left(\mathbb{X}^{(M)^{\mathrm{T}}}\mathbb{X}^{(M)}\right)^{-1}\mathbb{X}^{(M)^{\mathrm{T}}}Y$$
$$\hat{\sigma^2}^{(M)}(Y) = \frac{1}{n-\#M}\left\|Y - \mathbb{X}^{(M)}\hat{\beta}^{(M)}(Y)\right\|^2$$

The term "good" refers to a criterion by which we can define the "goodness" of M. Here we will use Mallows' Cp.

$$\mathrm{Cp}^{(M)} := \frac{1}{\hat{\sigma^2}}\sum_{i=1}^{n}\left(y_i - \hat{y}_i^{(M)}\right)^2 - n + 2\#M$$

The minimization this value is equivalent to the minimization of the sum of predicted squared errors (SPSE).

2.7 Simulated Annealing

The set of predictors contains $k = 319$ free selectable elements (the constant shall remain). Therefore the power set $\mathcal{P}$, the set of possible subsets M, contains of $2^k = 2^{319}$ elements. If we want to find a subset M so that for all $N \in \mathcal{P}$ the inequation holds

$$\mathrm{Cp}^{(M)} \leq \mathrm{Cp}^{(N)}$$

we have to calculate $\mathrm{Cp}^{(N)}$ for every $N \in \mathcal{P}$. But this is a calculation beyond our current computing power.

Simulated annealing (SA) is a probabilistic technique for approximating the global optimum of a given function. Specifically, it is a metaheuristic to approximate global optimization in a large search space. It simulates the slow cooling of a thermodynamic system through random numbers. With this algorithm it is possible to find a "good" local minimum in a short time.

The algorithm works on an arbitrary set, in our case $\mathcal{P}$. Let $x_0 \in \mathcal{P}$ be the initial set of predictors, $T_0 \in (0, \infty)$ be the initial temperature of the system and $i_{\max} \in \mathbb{N}$ be the maximal number of time steps. Then the algorithm needs certain functions.

- $\mathrm{cost}\colon \mathcal{P} \to \mathbb{R}$
  Calculates the cost of a given predictor set.
- $\mathrm{temp}\colon \mathbb{R} \times \mathbb{N}^2 \to (0, \infty)$
  Calculates the temperature according to the given initial temperature and time steps. It is a monotonically decreasing function in the second parameter.
- $\mathrm{nbr}\colon \mathcal{P} \to \mathcal{P}$
  Generates a random neighbor of a given predictor set.
- $\mathrm{prob}\colon \mathbb{R}^2 \times (0, \infty) \to [0, 1]$
  Calculates the probability of changing the current set or state to the neighbor.
- $\mathrm{rnd}(0, 1)$
  Returns a random number in the interval $[0, 1]$.

The listing shows one variant of the pseudocode of the SA algorithm.

Listing: SA algorithm

```
$c_0 = \mathrm{cost}(x_0)$

**for** $(i = 1, i \leq i_{\max})$ {
    $T = \mathrm{temp}(T_0, i, i_{\max})$

    $x_1 = \mathrm{nbr}(x_0)$
    $c_1 = \mathrm{cost}(x_1)$

    **if** $(\mathrm{prob}(c_0, c_1, T) \geq \mathrm{rnd}(0, 1))$ {
        $x_0 = x_1$
        $c_0 = c_1$
    }
}
```

2.8 Model Validation

3 Model Selection

3.1 Choosing a Neighbor

As said in section 2.7 with simulated annealing we want to select a good model for the prediction. For this we have to define the functions and parameters of the algorithm. The most important one is the nbr-function whose purpose it is to choose a neighbor with high quality since the final solution will be determined by a sequence of neighbors. In the most cases it is best to select a neighbor not too far away from the given subset.

Our nbr-function generates a random natural number $r \in \{2, \ldots, k + 1\}$. This number represents the index of a measured wavelength. If this predictor is already in our current subset then we remove it. In the other case we append it to the subset. That way our function obviously picks a near neighbor. The pseudocode is shown in the following listing.

Listing: nbr-function

```
**function** $\mathrm{nbr}(M)${
    $r = \mathrm{rnd}\,\{2, \ldots, k + 1\}$

    **if** $(\lambda_r \in M)${
        $\tilde{M} = M \setminus \{\lambda_r\}$
    }**else**{
        $\tilde{M} = M \cup \{\lambda_r\}$
    }

    **return** $\tilde{M}$
}
```

All other functions were defined with a standard scheme, respectively. It is clear that we set

$$\mathrm{cost}(M) := \mathrm{Cp}^{(M)}$$

In most applications prob is defined to be an analogy with the transitions of a physical system.

$$\mathrm{prob}(c_0, c_1, T) := \exp\left(\frac{c_0 - c_1}{T}\right)$$

Details of temp are not really important as long as it monotonically decreases in the second parameter. So let $\alpha \in (0, 1)$.

$$\mathrm{temp}(T_0, i, i_{\max}) := T_0 \alpha^i$$

3.2 Calibration

3.3 Suitability

4 Simulation

5 Conclusion

A Prediction Parameters

1400	1424	1436	1464	1468	1516
826.036122604903	-1324.58742726353	-642.720350868789	-980.901456464112	1201.95187828214	1117.94108322358

B R Source Code

Listing: utils.r

```
# get matrix for calculating parameters
mlr.par.mat = function(design_mat){
    transp_design_mat <- t(design_mat)

    # return
    solve(transp_design_mat %*% design_mat) %*% transp_design_mat
}

# get hat-matrix of a given design-matrix
# design_mat must have full rank
mlr.hat.mat = function(design_mat){
    # return
    design_mat %*% mlr.par.mat(design_mat)
}

# multiple linear regression residual sum of squares (rss)
mlr.rss = function(obs, design_mat){
    hat_mat <- hat.mat(design_mat)
    res <- obs - (hat_mat %*% obs)

    # return
    as.numeric(t(res) %*% res)
}

# multiple linear regression variance estimator
mlr.var = function(obs, design_mat){
    # return
    (mlr.rss(obs, design_mat)) / (length(obs) - dim(design_mat)[2])
}

# mallows cp
# idx_vec describes given model
# inv_mlr_var is the inverse variance of the given full model
mallows.cp = function(obs, design_mat, idx_vec, inv_mlr_var){
    # construct design matrix of given model
    design_mat_mod <- design_mat[,idx_vec]

    # return (will be returned as matrix; do not know why)
    (mlr.rss(obs, design_mat_mod) * inv_mlr_var) + (2*length(idx_vec)) - length(obs)
}

# model selection: forward selection method
ms.fwd.sel = function(obs, design_mat, inv_mlr_var){
    full_idx_vec <- seq(1, dim(design_mat)[2])
    # first column will be used every time
    idx_vec <- 1
    cp <- mallows.cp(obs, design_mat, idx_vec, inv_mlr_var)

    repeat{
        # vector of selection
        sel_vec = setdiff(full_idx_vec, idx_vec)

        if (length(sel_vec) == 0){
            break
        }
        tmp_idx_vec_1 <- c(idx_vec, sel_vec[1])
        tmp_cp_1 <- mallows.cp(obs, design_mat, tmp_idx_vec_1, inv_mlr_var)

        for (i in 2:length(sel_vec)){
            tmp_idx_vec_2 <- c(idx_vec, sel_vec[i])
            tmp_cp_2 <- mallows.cp(obs, design_mat, tmp_idx_vec_2, inv_mlr_var)

            if (tmp_cp_2 <= tmp_cp_1){
                tmp_cp_1 <- tmp_cp_2
                tmp_idx_vec_1 <- tmp_idx_vec_2
            }
        }

        if (cp >= tmp_cp_1){
            cp <- tmp_cp_1
            idx_vec <- tmp_idx_vec_1
        }else{
            break
        }

        # debug information
        print(idx_vec)
        print(cp)
    }

    # return
    idx_vec
}

# model selection: simulated annealing: neighbour function
ms.sa.nbr = function(idx_vec, max_idx){
    # get random index (1 is not used)
    rand_idx <- sample(2:max_idx, size = 1)

    if (rand_idx %in% idx_vec){
        # delete rand_idx in idx_vec
        nbr_idx_vec <- idx_vec[idx_vec != rand_idx]
    }else{
        # add rand_idx to idx_vec
        nbr_idx_vec <- c(idx_vec, rand_idx)
    }

    # return
    nbr_idx_vec
}

# model selection: simulated annealing: probability function
# costs will be minimized
ms.sa.prob = function(old_cost, new_cost, temp){
    # return
    exp( (old_cost - new_cost) / temp )
}

# model selection: simulated annealing
ms.sa = function(obs, design_mat, inv_mlr_var, idx_vec = c(1), temp = 10, alpha = 0.999, it_
     max = 20000, it_exit = 1200){
    max_idx <- dim(design_mat)[2]
    old_cost <- mallows.cp(obs, design_mat, idx_vec, inv_mlr_var);
    it_same <- 0

    for (i in 1:it_max){
        nbr_idx_vec <- ms.sa.nbr(idx_vec, max_idx);
        new_cost <- mallows.cp(obs, design_mat, nbr_idx_vec, inv_mlr_var)

        if ( ms.sa.prob(old_cost, new_cost, temp) >= runif(1) ){
            idx_vec <- nbr_idx_vec
            old_cost <- new_cost
            it_same <- 0
        }else{
            it_same <- it_same + 1
            if (it_same >= it_exit){
                break
            }
        }

        temp <- alpha * temp

        # debug information
        print(idx_vec)
        print(old_cost)
        print(temp)
    }

    # return
    idx_vec
}
```

References