197

Towards a Soil Information System with quantified accuracy

A prototype for mapping continuous soil properties

D.J. Brus
R. Vasat
G.B.M. Heuvelink
M. Knotters
F. de Vries
D.J.J. Walvoort

werkdocumenten

WOt
Wettelijke Onderzoekstaken Natuur & Milieu

WAGENINGEN UR
For quality of life

Towards a Soil Information System with quantified accuracy

The 'Working Documents' series presents interim results of research commissioned by the Statutory Research Tasks Unit for Nature & the Environment (WOT Natuur & Milieu) from various external agencies. The series is intended as an internal channel of communication and is not being distributed outside the WOT Unit. The content of this document is mainly intended as a reference for other researchers engaged in projects commissioned by the Unit. As soon as final research results become available, these are published through other channels. The present series includes documents reporting research findings as well as documents relating to research management issues.

This document was produced in accordance with the Quality Manual of the Statutory Research Tasks Unit for Nature & the Environment (WOT Natuur & Milieu) and has been accepted by Frank Veeneklaas programme coordinator of WOT Natuur & Milieu.

WOt Working Document **197** presents the findings of a research project commissioned by the Netherlands Environmental Assessment Agency (PBL) and funded by the Dutch Ministry of Agriculture, Nature and Food Quality (LNV). This document contributes to the body of knowledge which will be incorporated in more policy-oriented publications such as the Nature Balance and Environmental Balance reports, and Thematic Assessments.

Towards a Soil Information System with quantified accuracy

A prototype for mapping continuous soil properties

D.J. Brus

R. Vasat

G.B.M. Heuvelink

M. Knotters

F. de Vries

D.J.J. Walvoort

Werkdocument 197

Wettelijke Onderzoekstaken Natuur & Milieu

Wageningen, juni 2010

ABSTRACT

Brus, D.J., R. Vašát, G.B.M. Heuvelink, M. Knotters, F. de Vries, D.J.J. Walvoort, 2009. *Towards a Soil Information System with quantified accuracy. A prototype for mapping continuous soil properties.* Wageningen, Statutory Research Tasks Unit for Nature and the Environment, WOt-werkdocument 197. 160 p, 31 fig.; 5 tab.; 17 ref.

This report describes the potential and functionality of software for spatial analysis, prediction and stochastic simulation of continuous soil properties using data from the Dutch Soil Information System (BIS). A geostatistical framework and R codes were developed. The geostatistical model of a soil property has a deterministic component representing the mean value within a soil category, and a stochastic component of standardized residuals. The standardized residuals are interpolated or simulated based on the simple kriging system. The software was tested in four case studies: exchangeable soil pH, clay content, organic matter content and Mean Spring Water table depth (MSW). It is concluded that the geostatistical framework and R codes developed in this study enable to predict values of continuous soil properties spatially, and to quantify the inaccuracy of these predictions. The inaccuracy of a spatial prediction at a certain location is quantified by the kriging variance, which can be interpreted as an indication of the uncertainty about the true value.

Key words: R code, Accuracy, Uncertainty, Kriging, Simulation, Geostatistics, Digital Soil Mapping, Soil Map

P.O. Box 47, 6700 AA Wageningen, Netherlands
Phone: (0317) 48 07 00; Fax: (0317) 41 90 00; e-mail: info.alterra@wur.nl

The Working Documents series is published by the Statutory Research Tasks Unit for Nature & the Environment (WOT Natuur & Milieu), part of Wageningen UR. This document is available from the secretary's office, and can be downloaded from www.wotnatuurenmilieu.wur.nl.

Statutory Research Tasks Unit for Nature & the Environment, P.O. Box 47, NL-6700 AA Wageningen, The Netherlands
Phone: +31 317 48 54 71; Fax: +31 317 41 90 00; e-mail: info.wnm@wur.nl;
Internet: www.wotnatuurenmilieu.wur.nl

Contents

Preface

In the years to come we are challenged to update and upgrade the Soil Information System of the Netherlands. Part of this challenge is to quantify uncertainty about soil properties in an intelligible way, ready to be applied in the many decision processes in which information from the Soil Information System is used. This report describes the second step 'Towards a Soil Information System with quantified accuracy'. The first step has been described in WOt report 58, 'Towards a Soil Information System with quantified accuracy. Three approaches for stochastic simulation of soil maps' (Brus and Heuvelink, 2007).

This research is part of the strategic research program "Sustainable spatial development of ecosystems, landscapes, seas and regions" and is funded by the Dutch Ministry of Agriculture, Nature Conservation and Food Quality (LNV). The Netherlands Environmental Assessment Agency (PBL) is one of the main stakeholders of this program. Soil information with quantified accuracy is important in environmental modeling projects for PBL. We would like to thank Harm Houweling and Frank Veeneklaas of WOt for their unremitting support of research on quantified accuracy.

Wageningen, December 2009

Dick Brus, Radim Vašát, Gerard Heuvelink, Martin Knotters, Folkert de Vries, Dennis Walvoort

Summary

Information on soil properties, provided by the Dutch soil information system (BIS), is based on predictions, estimates and more or less accurate observations. In many decision processes it is important to consider statistical information on uncertainty about soil properties. Information on uncertainty should be provided in an intelligible way, ready to be applied in tools on statistical decision theory. This report describes the potential and functionality of software for spatial analysis, prediction and stochastic simulation using data from the BIS. The current study focuses on the development of R codes for kriging methods.

A geostatistical framework was developed with the following six stages: i) importing data from the soil information system, ii) data preprocessing, iii) exploratory data analysis, iv) building models of spatial variation, v) geostatistical (co)prediction and (co)simulation, and vi) exporting resulting maps. Because it might be expected that the mean and variance of many soil properties depend on the soil type and vary across soil categories, the soil map was used as an auxiliary data source. The geostatistical model of a soil property has a deterministic component representing the mean value within a soil category, and a stochastic component of standardized residuals. The standardized residuals are interpolated or simulated based on the simple kriging system.

The framework was tested in four case studies. In the first case study the spatial distribution of exchangeable soil pH in the soil layer from 0 to 25 cm was analyzed. Weighted averages for this layer were calculated from all contributing soil horizons. Prior to the weighted averaging the activity of hydrogen ions was taken, and back-transformed to the pH scale afterward. Spatial predictions were based on 1621 point observations. The mean and variance of soil pH vary with the so called PAWN soil types (21 classes). This dependence was incorporated in the geostatistical model. Simple kriging of residuals was applied to predict soil pH spatially. Besides a map with predicted values a map with standard deviations of the predictions was constructed. This map reflects the uncertainty about the true spatial distribution of soil pH. Information on uncertainty was also provided by maps representing the 95% lower limit and 95% upper limit of the predictions, and by maps of simulated possible realities.

In the second case study a map of clay content of the layer from 0 to 50 cm was made for the Netherlands, as well as detailed maps of high resolution for two subareas in the western and eastern part of the country. High resolution should not be confused with high accuracy, however.

The third case study concerns the spatial distribution of organic matter content in three different soil layers: 0-25 cm (195,547 observations), 25-50 cm (116,509 ob-

servations) and 50-100 cm (47,547 observations). The data of the three layers were analyzed simultaneously. Cross-correlations in organic matter content between layers were used in spatial prediction and simulation for the Netherlands and for a subarea in the eastern part of the country. Because cross-correlations were weak, predictions for less intensively observed layers could not benefit from observations in more intensively observed layers.

In the fourth case study a characteristic of the temporal fluctuation of water table depth, the mean spring water table depth (MSW), was interpolated spatially and simulated. Prior to spatial interpolation a logarithmical transformation was applied to the data to obey the normality assumption. The soil type map of the Netherlands was used as auxiliary information in the interpolation. Simple kriging of residuals was applied to predict MSW spatially. Conditional sequential Gaussian simulation based on simple kriging was performed to simulate possible realities of the spatial distribution of MSW.

It is concluded that the geostatistical framework and R codes developed in this study enable to predict values of continuous soil properties spatially, and to quantify the inaccuracy of these predictions. The inaccuracy of a spatial prediction at a certain location is quantified by the kriging variance, which can be interpreted as an indication of the uncertainty about the true value. Maps of kriging variances can be very useful in optimizing observation networks. It should be noted that the kriging variance reflects the inaccuracy given a *model* of spatial structure. Because uncertainty about the true spatial structure is not accounted for, the kriging variance is an approximation of the uncertainty about the true values of soil properties at unvisited locations. A valuable next step would be to improve quantified uncertainty by using results of validation.

Chapter 1

Introduction

1.1 Background

Information on soil properties, provided by the Dutch soil information system (BIS), is used in decision processes in fields such as agriculture, forestry, environmental protection, rural planning, and nature conservation. Since the information in BIS is based on predictions, estimates and more or less accurate observations, we are uncertain about the true soil properties. In many decision processes it is important to consider statistical information on this uncertainty (based on Morgan and Henrion (1990)):

- when the attitude of people toward risk is important, for example in case of a clear risk aversion, or when decisions are taken under the precautionary principle, the role of which is increasingly important in international and national policy (United Nations, 1992; Commission of the European Communities, 2000; Minister van Volkshuisvesting, Ruimtelijke Ordening en Milieubeheer, 2001);
- when inaccurate information from different sources must be combined. Quantified uncertainty determines the weighting;
- when a decision has to be taken about whether to reduce uncertainty by acquiring additional information, e.g., data worth analysis (Freeze et al., 1992; Back, 2007);
- when the loss function in the analysis depends on the inaccurate quantity and decision variable in a way that including uncertainty in the analysis gives different results than fixing all continuous uncertain quantities at some central, nominal value.

To these four situations described by Morgan and Henrion (1990) we add the following cases in which it is important to consider statistical information on uncertainty:

- controversial decision making processes in which quantitative information on uncertainty can prevent involved parties for over- or under-emphasizing uncertainties;

- in testing against legal standards;
- in assessing the significance of effects of measures.

1.2 Problem definition

Although statistical information on uncertainty about soil properties is indispensable or at least very useful in many decision processes, this information is not provided by the BIS yet. Information on uncertainty about soil properties should be provided in an intelligible way, ready to be applied in tools on statistical decision theory. Software which provides quantitative information on uncertainty about soil properties need to be developed.

1.3 Aim

This report aims to describe the potential and functionality of software, that was developed to perform spatial analysis, prediction and stochastic simulation using data from the Dutch soil information system (BIS) database. Brus and Heuvelink (2007) describe three methods for mapping soil properties that account for uncertainty: kriging methods, Bayesian Maximum Entropy and Markov Random Fields. The current study focuses on the development of R codes for kriging methods. Two different groups of users are assumed to be working with these codes. While for the R beginners there are basic settings before each step, which allows them to control the main parameters entering the analysis process, advanced R users can modify any part of the codes at will.

1.4 Outline

Chapter 2 describes the framework for geostatistical simulation of soil properties: importing data from BIS, data preprocessing, exploratory data analysis, modelling spatial variation, geostatistical (co)prediction and (co)simulation, and exporting maps to BIS and other platforms. Chapter 3 presents four case studies. The first case study is a spatial single layer analysis of exchangeable soil pH. The second case study is a multiple layer analysis of clay content. The third case study is a multiple layer analysis of organic matter content. The fourth case study is on spatial simulation of 'mean spring water table depth' (MSW), which is a fluctuation characteristic of the water table depth. In the Netherlands these fluctuation characteristics are mapped concurrently with soil surveys, and stored in the BIS. The R codes are given in the Appendices. Chapter 4 gives some concluding remarks concerning the information on uncertainty about soil properties obtained by the software described in this report.

Chapter 2

Framework for spatial prediction and simulation of soil properties

2.1 Introduction

This chapter presents a geostatistical framework for spatial prediction and simulation of soil properties. This framework has the following six stages:

1. *Importing data from the soil information system.*
2. *Data preprocessing*: selecting data for the area and period of interest, defining layers of interest, calculating soil property values for the layers of interest.
3. *Exploratory data analysis*: exploring the distribution of soil properties by plotting the values at a map, by plotting histograms, normal Q-Q plots and boxplots. Removing outlying values.
4. *Building models of spatial variation*: variogram and cross-variogram modelling.
5. *Geostatistical (co)prediction and (co)simulation*: spatial prediction by local simple (co)kriging, generating possible realities of spatial soil property distributions by conditional sequential Gaussian (co)simulation.
6. *Exporting resulting maps.*

The next Sections give the details of these six stages.

2.2 Importing data from the soil information system

A soil information system (SIS) is a convenient way to store and organize soil data. The Dutch SIS, also known as BIS, contains soil maps at various spatial scales, thousands of soil profile descriptions, and many records on soil physical and chemical properties.

The information stored in the SIS is accessible via various interfaces (*e.g.*, a console, GIS). In this report, we use R (R Development Core Team, 2009) to communicate with the SIS. This is an obvious choice, since throughout this report all analyses will also be performed by R. Several R-add-on packages are available for database access. We selected the RJDBC-package (Urbanek, 2009) for its simplicity, efficacy, and adherence to DBI (R-SIG-DB, 2009).

The SIS contains thousands of tables. To keep the database transparent and easily accessible, three views have been defined for the deltaBIS project:

- V_PFB_DELTABIS, which contains profile descriptions from the national soil survey scale 1:50000 and other soil surveys (field estimates and laboratory analysis);
- V_LSK_DELTABIS, which contains profile descriptions of the LSK soil survey (Visschers et al., 2007);
- V_AUGERING_DELTABIS, which contains other profile descriptions (field estimates only, horizons have not been analyzed in the laboratory)

A view is a virtual table that is dynamically build by invoking a stored query. The query combines data residing in physical (real) tables in the database. Views greatly facilitate database access, in particular for non-regular SIS-users. The views given above contain all information necessary for spatial prediction and simulation of soil properties.

2.3 Data preprocessing

After the data are loaded into the R environment the data are filtered to obtain a data set covering the period and area of interest. Two types of filtering are distinguished for this purpose: time filtering and spatial extent filtering. Spatial extent data filtering can be based on spatial co-ordinates indicating the boundaries of a rectangular area, or on spatial polygons (ESRI shapefile polygons) delineating irregularly shaped areas.

After data filtering soil samples are excluded that were taken from more than one soil horizon (i.e., the soil sample length crosses the border between two soil horizons), because the measured soil property cannot be assigned to one particular soil horizon.

In the next step the soil property value is estimated for the soil horizon. For locations where only one soil sample was taken from the appropriate soil horizon, the estimation simply equals the value of the soil property observed at this soil sample. At locations where more than one sample was taken from one horizon the weighted average is taken as an estimate, using the lengths of the soil samples as weights.

After the soil properties are estimated for each soil horizon, the upper depths and lower depths of the soil layers for which the analysis should be done are defined. Next, the values of the soil properties in the soil horizons that contribute to the defined soil layers are extracted. The lengths with which soil horizons contribute to the defined soil layers are used as weights in calculating the average values of the soil property for the defined soil layers. The ratio by which each horizon contributes to

the estimate for a soil layer is calculated to check if the information at an individual sampling location is complete. If the sum of ratios for a certain soil layer at a certain location is less than 1, then information is missing, and the soil profile is excluded from analysis. The final step in data preparation is to compute the weighted average at locations where the information on a soil property for the defined soil layers is complete. Dependent on the individual soil property two different approaches can be applied in computing the weighted average values. In the case of soil properties like exchangeable soil pH the lengths by which soil horizons contribute to the defined soil layers simply represent the averaging weights. For soil properties like organic matter content and clay content the bulk density is employed as well in calculating weights, because this type of soil properties are not related to soil volume but soil mass. After multiplying the soil horizon volume with the bulk density the soil mass is obtained, which is used to compute the organic carbon mass (or clay mass) in the next step. The weighted average then equals the sum of organic matter (clay) mass divided by the sum of soil mass multiplied by 100.

2.4 Exploratory data analysis

A map of all observed values on a soil property provides a first insight into the spatial variation. Basic statistics like mean, variance, minimum, maximum and median can be computed to obtain insight into the distribution of the soil property. Histograms can be constructed to evaluate whether the data depart from the normal distribution or not. Alternatively, the normal Q-Q plot can indicate deviations from the normal distribution. These plots can also be used to evaluate the effect of data transformation. Furthermore, a set of box-plots describing how the soil property distribution varies with the soil type may be useful when taking a decision about the geostatistical model. Outlying values, e.g. observation errors, can be removed from the data set in this stage.

2.5 Building models of spatial variation

It might be expected that the mean and variance of many soil properties depend on the soil type and vary across soil categories. Therefore, the geostatistical model of spatial variation is build up such a way that this important feature can be taken into account. A soil map, for example the PAWN soil type map (Wösten et al. (1988), see Figure 2.1) can be used as an auxiliary data source, but any other map of different soil categories can be incorporated into the code on demand. The geostatistical model is then defined as follows:

$$Z(\mathbf{x}) = m_i + \sigma_i \cdot \varepsilon(\mathbf{x}) \,, \tag{2.1}$$

with $Z(\mathbf{x})$ the soil variable, m_i the mean value of $Z(\mathbf{x})$ in the ith soil category, σ_i the standard deviation $Z(\mathbf{x})$ in the ith soil category, and $\varepsilon(\mathbf{x})$ is a stochastic residual with zero mean and unit variance.

The spatial structure of the stochastic residuals is described by experimental variograms (and cross-variograms in the case of co-kriging). For an explanation of variograms we refer to textbooks on geostatistics like Isaaks and Srivastava (1989)

and Goovaerts (1997). Next appropriate variogram models are fitted to the experimental variograms. Often a nested variogram model combining two spherical curves is most appropriate.

2.6 Geostatistical (co)prediction and (co)simulation

Values of soil properties at unvisited locations can be predicted by the geostatistical interpolation technique *kriging*. For a description of kriging algorithms we refer to textbooks like Isaaks and Srivastava (1989) and Goovaerts (1997). Besides an interpolated value, kriging provides a kriging variance, reflecting the accuracy of the interpolated value. Since the mean of stochastic residuals equals zero, *simple kriging* can be applied to predict the residual spatial distribution over the entire area. The prediction process can be speeded up by applying the local kriging approach, in which observations in a restricted neighborhood are used for interpolation.

Interpolation by kriging results in best linear unbiased predictions (BLUP) for unvisited locations, and kriging variances reflecting the accuracy of these predictions. Besides this information, a large number of possible realities of spatial distributions of a soil property might be useful, in particular as an input in uncertainty analysis. These realities can be generated by Conditional sequential Gaussian simulation (Goovaerts, 1997).

In multiple-layer spatial analysis local *simple co-kriging* (Goovaerts, 1997) is applied. Also, the multivariate simulation is based on local simple co-kriging.

2.7 Exporting resulting maps

Spatial interpolation and prediction results in maps of the soil properties of interest. By using the capabilities of the R-system, these maps can be exported to various data formats. For example,

- graphical formats like png, jpeg, and pdf. These formats can be used to include maps in reports or presentations;
- GIS-layers. The resulting maps can be exported as layers for geographical information systems like ArcView, ArcGIS, SAGA for further processing;
- ASCII files, for further processing (*e.g.*, as inputs to sophisticated process models);
- as database tables, including as part of the SIS itself.

In future releases it is intended to implement the possibility to store only the 'recipe' to regenerate each map. This greatly reduces storage demands en guarantees reproducibility.

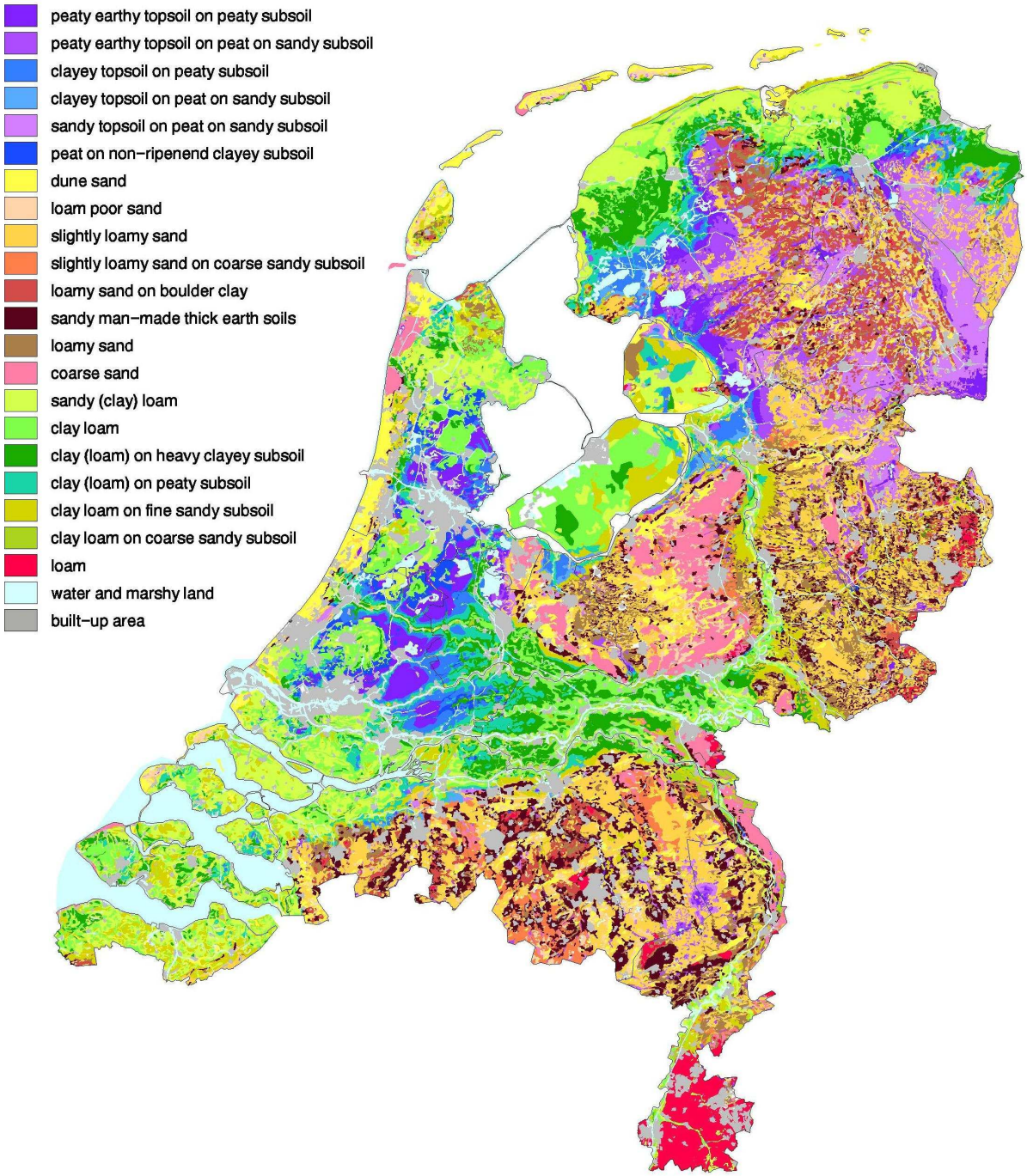


Figure 2.1: PAWN soil type map.

Chapter 3

Case studies

3.1 Exchangeable soil pH - single layer spatial analysis

For the R code used in this case study we refer to Appendix A.

This case study presents an example of exchangeable soil pH data analysis, spatial prediction and simulation, for the soil layer from 0 to 25 cm.

The soil pH values used in this example were prepared following the process described in Section 2.3, and represent weighted average values from all soil horizons which contribute to the required soil layer. The activity of hydrogen ions (H^+) is taken prior to the weighted averaging, and back-transformed to the pH scale afterward. Finally 1621 point observations which fulfil all required conditions are used in the predictions.

The exploratory data analysis shows a very large range in collected data, with a minimum of 2.26 (extremely acid soils) and a maximum of 8.12 (moderately alkaline soils). Some other descriptive statistics of the exchangeable soil pH data are given in Table 3.1.

The bimodal form of the histogram in Figure 3.1 indicates that the area consists of two parts where the soil pH takes significantly different values. The map with plotted observations indicates that very high values occur in the coastal regions and in the very south of the Netherlands, while the central parts and the east have clearly more acid soils (Figure 3.2). The high alkalinity along the North Sea coast can be explained by the marine sediments rich of calcareous materials which are widely spread in these areas.

Figure 3.3 gives a set of box-plots, to illustrate the relationship between soil pH and PAWN soil types. The mean and the variance apparently vary with the PAWN soil types (21 classes). This dependence is incorporated in the geostatistical model, see Eq. (2.1). The geostatistical model was applied to compute standardized pH residuals. Figure 3.4 shows the histogram of standardized residuals, which approximates the normal distribution.

The spatial auto-correlation of standardized residuals was described with a nested variogram model, which combines two spherical models (Figure 3.5). Apparently,

the residuals show a strong spatial dependence up to 5 km approximately, and a weak spatial dependence from 5 to 100 km.

Since the model mean and variance are assumed to depend on the PAWN soil types, a simple kriging approach may be used to predict the spatial distribution of soil pH standardized residuals for the Netherlands (prediction grid 500 x 500 m). The map of exchangeable soil pH prediction for the Netherlands is given in Figure 3.6.

Figure 3.6 (right) shows the standard deviation map. Figure 3.7 shows two maps of the soil pH spatial distribution representing the 95% lower limit and 95% upper limit of the predictions.

As described in Section 2.6 possible realities of standardized residuals for the Netherlands can be simulated by conditional sequential Gaussian simulation based on simple kriging. Figure 3.8 shows four possible realizations of the spatial distribution. The number of observations that are used for each particular prediction or simulation is limited to be 30, and the maximum distance between the observation point and the prediction point or simulation point is set as 20 km.

To evaluate the prediction process the predictions are compared with the original data. Figure 3.9 shows the histograms of the original data (left) and the predictions (right). Note that the bimodal character of the distribution of soil pH data (1.621 observations) persists even more pronounced in the predicted values (122.282 values). This is possibly caused by different ratios of original and predicted location numbers within PAWN soil categories. The basic statistics in Table 3.1 indicate a relatively short range of predictions. This reflects the smoothing effect of kriging (Isaaks and Srivastava, 1989).

The histograms of the simulated values indicate that the bimodal character of the distribution completely disappeared (Figure 3.10). The mean and the median of the observed and simulated values (Table 3.1) show a good match. However, the variance of simulated values is larger than the variance of the the observed values, and the minimum and maximum values indicate that the simulated values have a larger range of variation than the observed values. Note that the minimum and maximum values of the simulations are not realistic from a physical point of view.

Table 3.1: Basic statistics for the observed and predicted pH_KCl values, and three simulations.

Statistics	pH_KCl				
	observations	predictions	simulation 1	simulation 2	simulation 3
Min	2.26	3.35	-0.57	-0.25	-1.00
Max	8.12	7.97	12.19	11.78	11.97
Median	5.35	5.04	5.37	5.33	5.38
Mean	5.56	5.47	5.48	5.44	5.50
Variance	2.08	1.42	2.26	2.29	2.32
Standard deviation	1.44	1.44	1.50	1.51	1.52

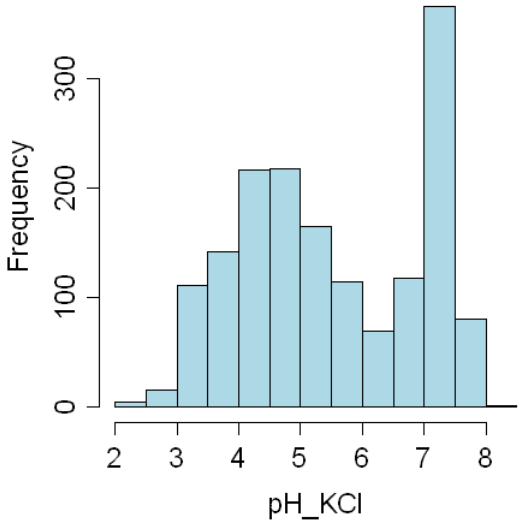


Figure 3.1: Histogram of exchangeable soil pH for the soil depth 0 - 25 cm.

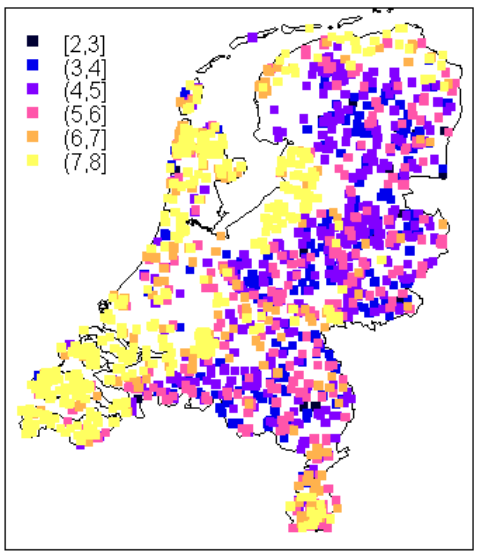


Figure 3.2: The Netherlands with plotted observations on exchangeable soil pH for the soil depth 0 - 25 cm.

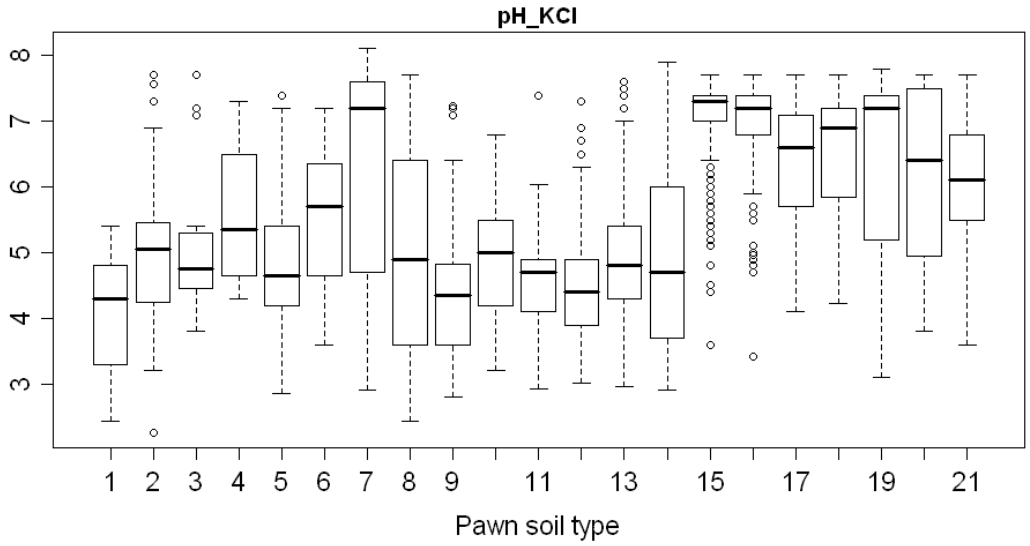


Figure 3.3: Box-plots of soil pH for each of the 21 PAWN soil categories.

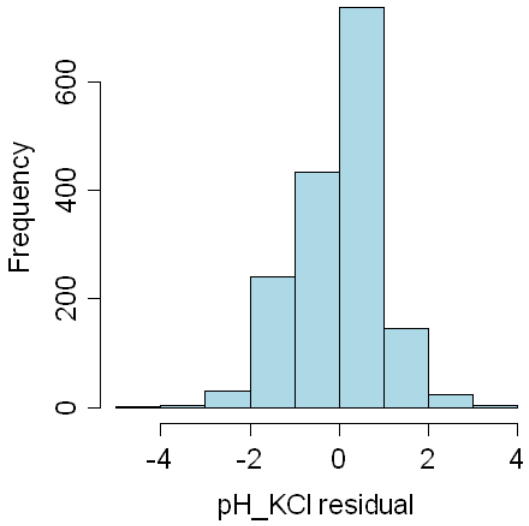
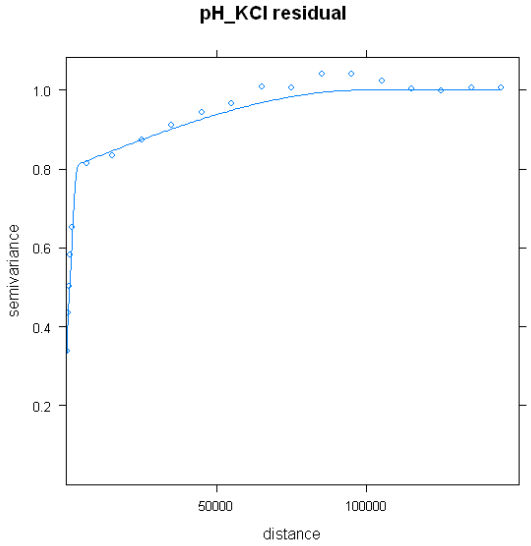


Figure 3.4: Histogram of soil pH standardized residuals.



Nugget	=	0.30
Partial sill	=	0.70
Range	=	100000
Model	=	double spherical

Figure 3.5: Standardized residual variogram (double spherical) model for soil pH data.

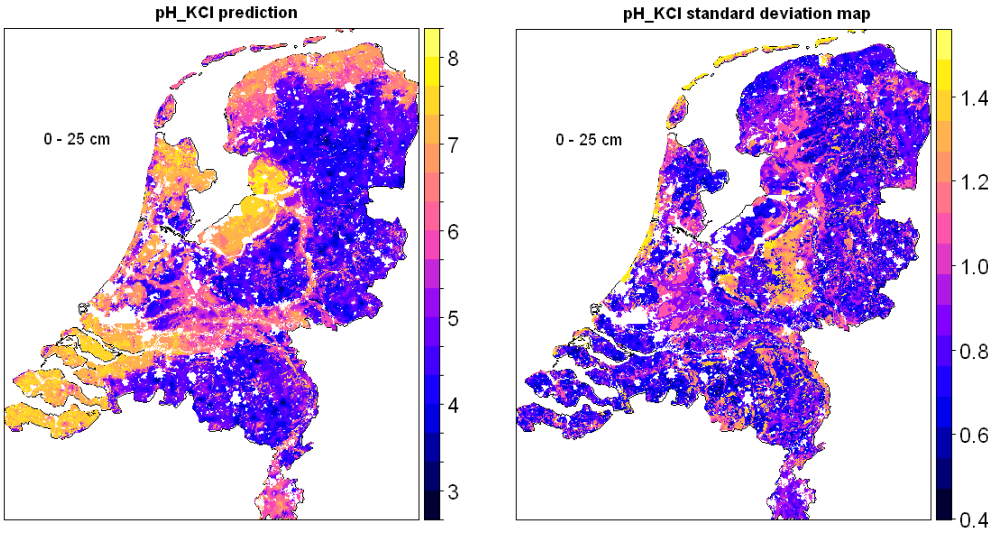


Figure 3.6: Kriging prediction map of exchangeable soil pH (left) and standard deviation map (right).

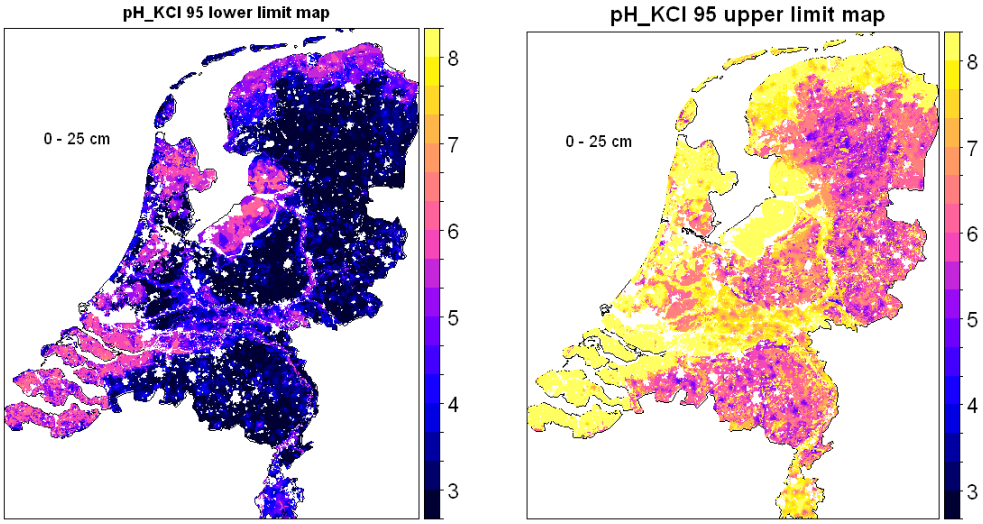


Figure 3.7: Soil pH spatial distribution 95% lower limit map (left) and 95% upper limit map (right).

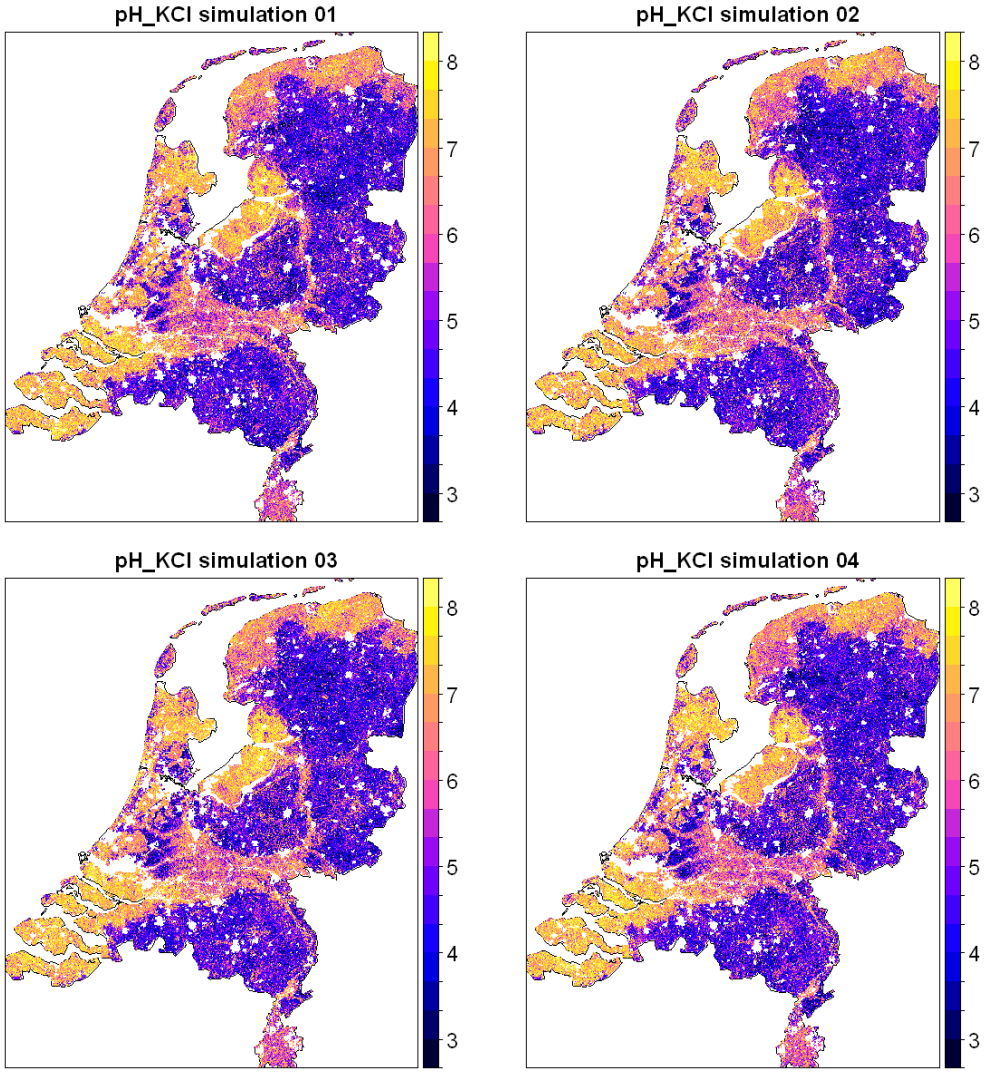


Figure 3.8: Four possible realities of soil pH spatial distribution generated with sequential Gaussian simulation.

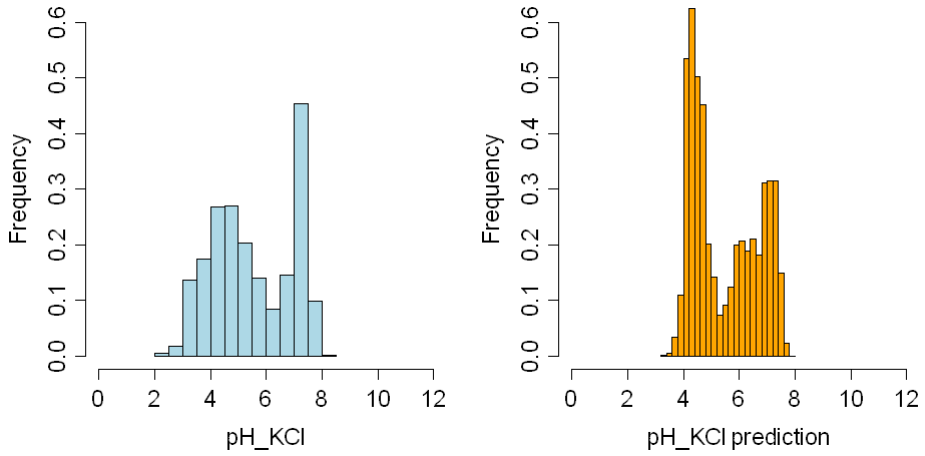


Figure 3.9: Comparison of observed and predicted soil pH values.

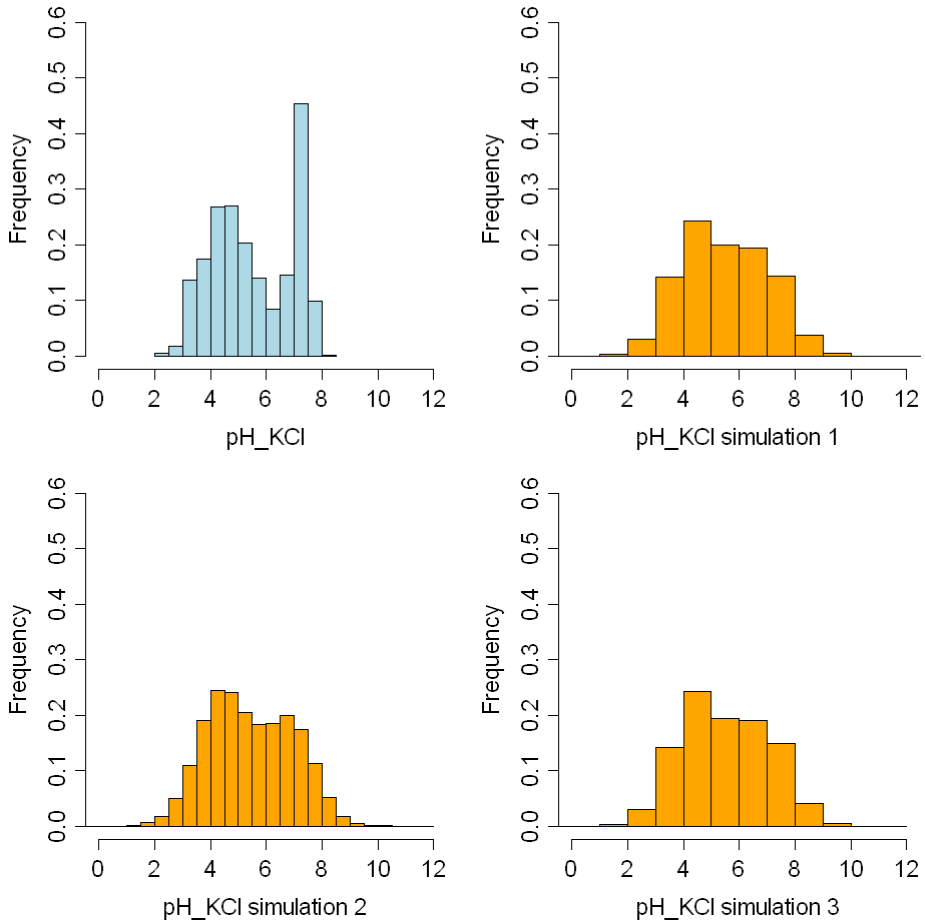


Figure 3.10: Comparison of observed soil pH values and three simulated maps of pH_KCl

3.2 Clay content - zooming facility

Appendices B and C give the R codes used in this case study.

This case study illustrates the spatial extent filter described in Section 2.3. In many studies the area of interest is not the entire area of the Netherlands, but a particular part. This can be a rectangular shape defined with X and Y coordinates, or an irregular shape, polygon or group of polygons, representing a particular soil unit, a land use category, a province, etc. Zooming into these smaller areas is an option of the current R code. This zooming facility is illustrated with an example on clay content.

Shape and resolution of the prediction or simulation map are determined by the extent and cell size. Obviously, the more we zoom the smaller the cell size should be set to get satisfactory results. High resolution should not be confused with high accuracy, however.

Figure 3.11 shows a map of predicted clay content of the layer from 0 to 50 cm for the Netherlands. The next two examples show some of the zooming possibilities. Figure 3.12 (left) shows predicted clay contents within peat soils in the western part of the Netherlands. Figure 3.12 (right) shows a rectangular area delimitated with X and Y coordinates which focuses on the neighborhood of two cities in the eastern part of the country (Arnhem and Nijmegen). A relatively high clay content for the soil layer 0 to 25 cm is found in the former backswamps between the rivers Rijn and Waal. North of the river Rijn, at the ice-pushed ridges of the Veluwe area with Pleistocene sandy deposits, the clay content is small.

As compared to the prediction map for the Netherlands the zooming examples show evidently more detailed information, which may be useful in particular studies. The interpolation grid size is 100 m in both cases, but can be set smaller or larger.

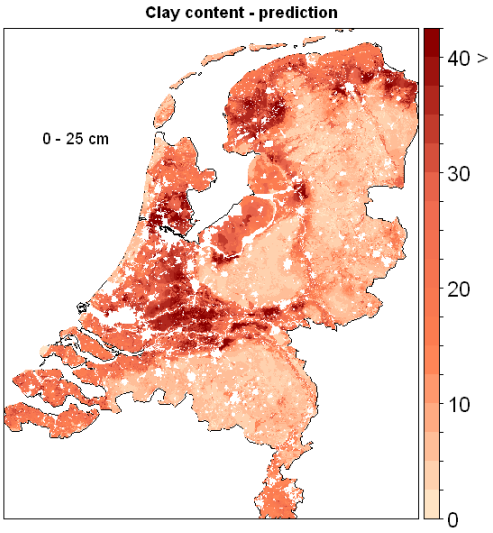


Figure 3.11: Clay content kriging prediction map for The Netherlands.

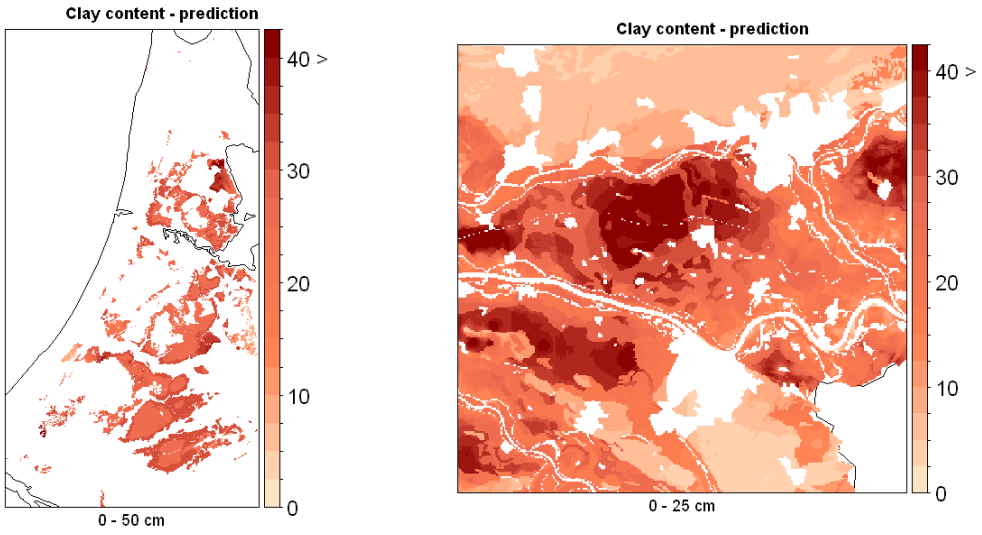


Figure 3.12: Clay content kriging prediction maps zoomed to the peat soils in the west (left) and to the rectangular area in the middle-east (right).

3.3 Organic matter content - multi-layer spatial analysis

Appendices D and E give the R codes used in this case study.

In several cases it might be necessary to perform the spatial analysis not only for one, but for multiple soil layers at the same time. The R codes presented in appendices D and E enable to perform a multi-layer spatial analysis. Without any additional modification of the code three different soil layers can be simultaneously analyzed. Furthermore, the multivariate approach may considerably improve the precision of final prediction and simulation maps for the individual soil layers.

Three different soil layers (0-25, 25-50 and 50-100 cm) are defined for the multivariate analysis of organic matter content. There are 195,547 points for the first layer, 116,509 for the second and only 47,446 points are available for the third layer. After weighted averaging for all layers is done as described in Section 2.3, single variograms and cross-variograms can be modelled for the residuals, see Eq. (2.1). Next these models are used in co-prediction and co-simulation. A function for automatic fitting of a linear model of coregionalization (LMC) is provided in the Gstat package (Pebesma, 2004), but also an alternative option to fit LMC manually is possible when desired.

Figure 3.13 shows the variogram for organic matter content in the layer from 0 to 25 cm. Figure 3.14 shows experimental variograms and cross-variograms for the three soil layers, which are automatically fitted with LMC. While the nugget and sill values may differ from case to case, the range and variogram model (exponential) are always equal. From the large nugget-to-sill ratios we can conclude that the autocorrelations and cross-correlations of residual organic matter content over the entire area are weak. It should be noted that in a multivariate approach the predictions can only benefit from *strong* relationships in residual organic matter content between layers.

Simple kriging is applied to predict the spatial distribution of organic matter content for each layer separately, and simple cokriging is applied to predict the spatial distribution of organic matter content for all three layers simultaneously. Figure 3.15 shows the prediction map constructed with simple kriging, and a map of the kriging standard deviation. Figure 3.16 shows the prediction map constructed with simple cokriging. In both local simple kriging and local simple cokriging the number of observations used to predict or to simulate is limited to 30, and the maximum distance between the observation point and the prediction point or simulation point is 20 km.

Possible realities of organic matter content are generated for layers separately by conditional sequential Gaussian simulation. Realities are simultaneously generated by conditional sequential Gaussian co-simulation. Figure 3.17 shows maps of simulated realities. The differences in results of simulation and co-simulation are not very pronounced, which is explained from the weak cross-correlations (Figure 3.14).

Figure 3.18 shows simulations and co-simulations for the eastern part of the country (Arnhem, Nijmegen, and neighborhood). Also for this area the differences in simulations and co-simulations are small, resulting from the weak cross-correlations.

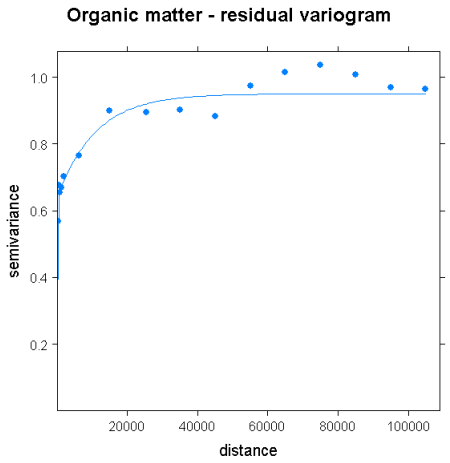


Figure 3.13: Variogram for organic matter content in the layer from 0 to 25 cm.

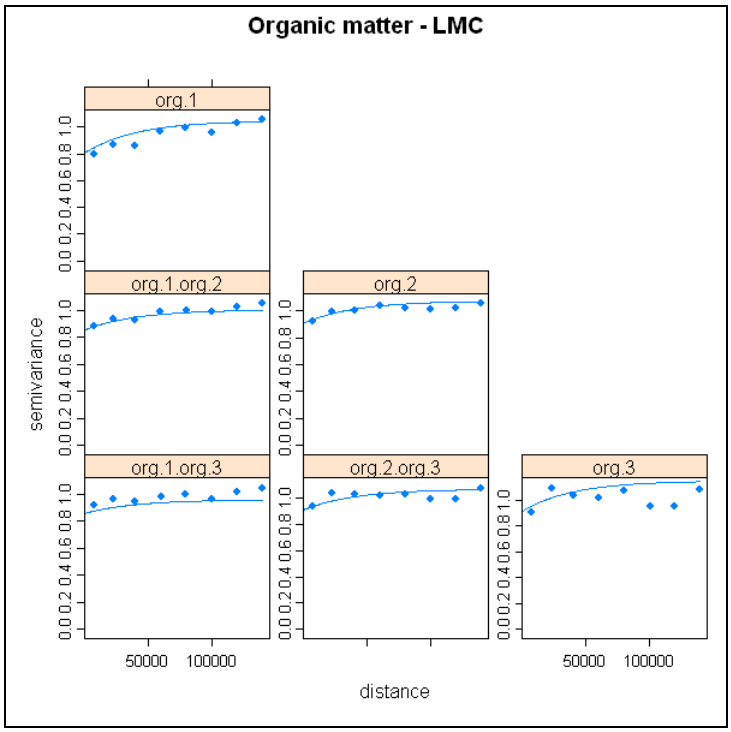


Figure 3.14: Organic matter single and cross-variograms for three different soil layers, fitted with LMC.

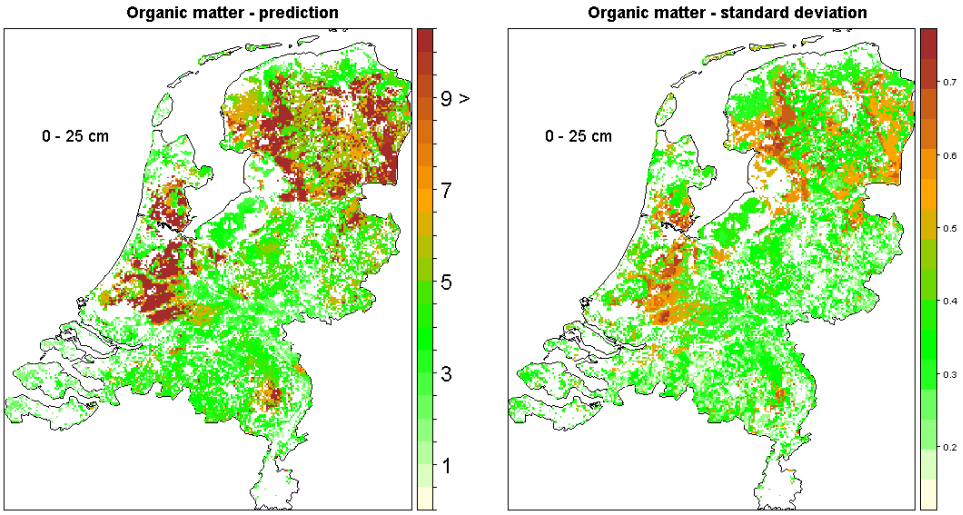


Figure 3.15: Prediction map of organic matter content, created with the univariate geostatistical approach (left) and standard deviations of spatial predictions (right).

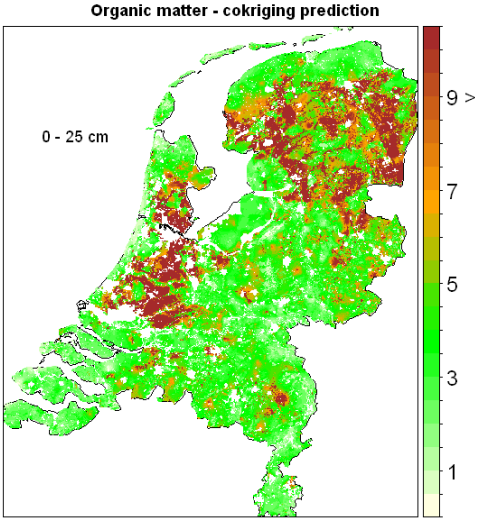


Figure 3.16: Prediction maps of organic matter content, created with the multivariate geostatistical approach.

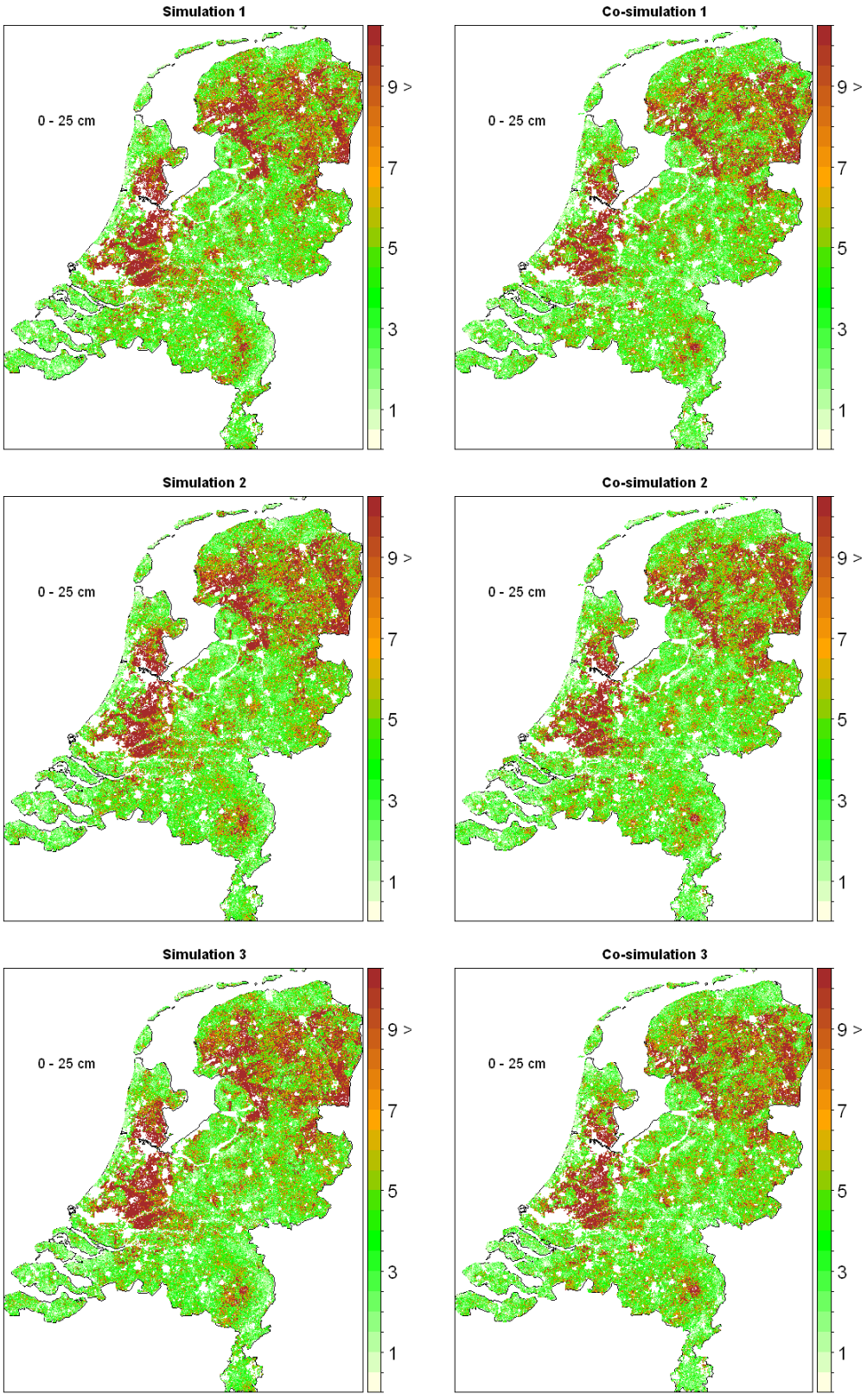


Figure 3.17: Three possible realities of organic matter content spatial distribution as a result of sequential Gaussian simulation for single layer only (left), and another three created with co-simulation (right).

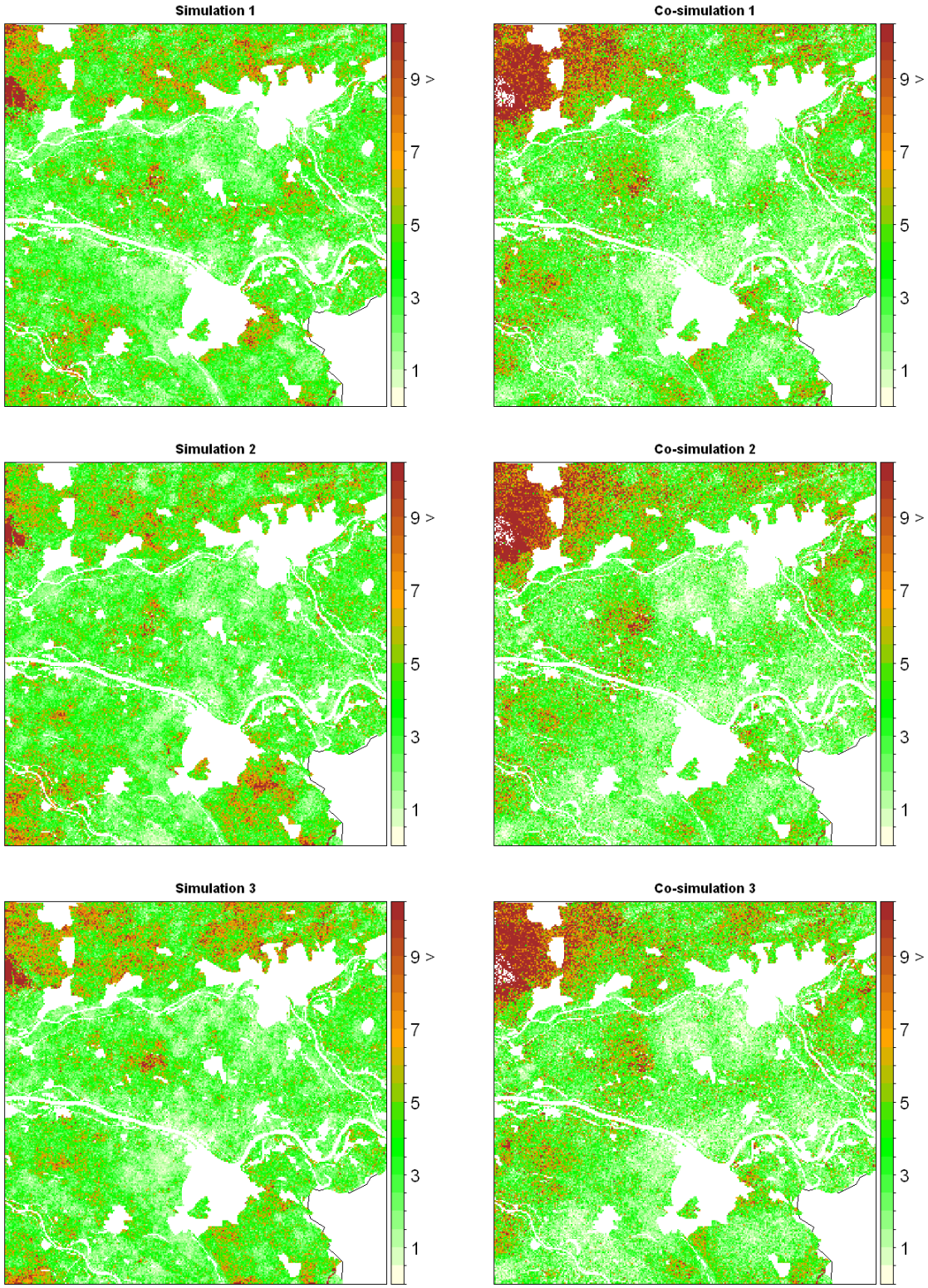


Figure 3.18: Three possible realities of organic matter content spatial distribution as a result of sequential Gaussian simulation for single layer only (left), and another three created with co-simulation (right). Zoomed into the eastern part of the country.

3.4 Mean spring water table depth (MSW): spatial prediction and stochastic simulation

Appendix F gives the R codes used in this case study.

3.4.1 Data preparation

Data for the spatial analysis of mean water table depths were extracted from the Dutch soil information system (BIS) establishing the connection between BIS and R environment using RODBC R-package, see Section 2.2. Two tables containing data on the mean highest water table (MHW) and mean lowest water table (MLW) were downloaded:
PFB_ALG table - 7749 records
LSK_ALG table - 3391 records
These data are not only field measurements, but also estimates at locations where the water table depth was not found within the augering depth or the depth of the observation well. These so called censored observations, ending with 1 (e.g. 121, 251), can be treated in several ways. Information would get lost when these censored observations were simply removed, especially for areas with high elevations and deep water tables. Since we do not want to loose any possible data we decided to keep the censored values within the entire dataset, while the influence of these data is largely reduced by logarithmic transformation. Besides this, ecological models for groundwater-dependent systems are not sensitive to variations of the water table at large depths.

During the data exploration, locations were found for which both tables give different MHW and MLW values. 133 of these duplicated locations were found in PFB_ALG and 138 in LSK_ALG. Records with duplicated locations were identified and extracted to examine which records are most likely and which should be erased. After merging the two tables a subset was made, based on the "bispnt.prn" data file containing information about soil type. Next, the mean spring water table depth (MSW) was computed using the following formula of Van der Sluijs (1990):

$$MSW = 5.4 + 0.38 \times MHW + 0.19 \times MLW\,. \tag{3.1}$$

By definition, the MSW value should always be inside the interval of MHW and MLW:

$$MSW \in \langle MHW, MLW \rangle\,. \tag{3.2}$$

However, this condition is not guaranteed by Eq. (3.1), in particular when the MHW and MLW values are very close to each other or equal. To obey the definition of MSW in these situations, the MSW value was forced to be equal to the nearest border of the interval:

$$\begin{aligned} \text{if } MSW > MLW \quad &\Rightarrow \quad MSW = MLW\,, \\ \text{if } MSW < MHW \quad &\Rightarrow \quad MSW = MHW\,. \end{aligned} \tag{3.3}$$

3.4.2 Exploratory data analysis

Figure 3.19 shows that the histogram of the mean spring water table depth data is skewed to the right. This indicates that the MSW data are far from the normal

distribution and hence some kind of transformation of the original data is needed prior to geostatistical modelling, prediction and simulation.

To get more insight into the distribution of the MSW, histograms which show the proportional representation and frequency distribution for each soil category were created (Figure 3.20). Notice that more than half of the overall MSW records belongs to two soil categories (Table 3.2).

Because the histogram of original MSW data is very skewed we decided to perform logarithmic transformation (natural logarithm) on the data. Since the natural logarithm does not exist for negative values, which occur in the MSW data (situations where the water level is above the earth surface most of the time during the year), all values were moved to set the minimum (-6.55) to be 1 before the transformation:

$$t\_MSW = \log\{MSW - \min(MSW) + 1\}\,. \tag{3.4}$$

After transformation the MSW data seem to be close enough to the normal distribution (Figure 3.21), to proceed with the geostatistical analysis. Note that the log-transformation causes that only positive interpolations will be produced and hence the minimum of interpolated data after back-transformation can never be smaller than the minimum of the original data. Figure 3.22 shows histograms of transformed MSW data for each soil type separately. The Normal Q-Q plot also indicates that the transformed MSW data are close to the normal distribution (Figure 3.23).

Table 3.3 gives some descriptive statistics of the MSW and transformed MSW data. Note that the MSW values are spread quite widely around the mean. The highest values occur mostly in the south and central-east parts of The Netherlands (i.e. Zuid-Limburg and the Veluwe), where the elevation is relatively high.

To examine the relationship between mean water table depth and soil type a set of box-plots was created (Figure 3.24). This set indicates only small differences between soil types 1 to 3, but quite large differences between the first three soil types and the others.

As there are clear differences between soil type means (Table 3.4), we assumed that the MSW mean is non-constant and varies with soil category. The variances also depend on the soil type. The varying mean and variance between soil types were incorporated into the geostatistical model given in Eq. (2.1).

3.4.3 Variography

The variogram of transformed MSW data indicates a clear, but weak, spatial dependence over the entire area of interest. A spherical model was fitted to the experimental variogram (Figure 3.25). The variogram of transformed MSW residuals (Figure 3.26) shows even weaker spatial dependence. However, when zoomed in to shorter distances a strong spatial dependence, which reaches a distance up to 5000 meters, was found (Figure 3.27).

3.4.4 Spatial prediction

To predict the MSW for the Netherlands, the soil type map was used as an auxiliary data source. Simple kriging of residuals was performed first, and next the predictions

were computed following the geostatistical model in Eq. (2.1). The number of nearest points included into the kriging prediction was set at 100. The variogram model of residuals as fitted in Figure 3.27 was used for the predictions. The prediction of transformed MSW data was next computed by multiplying the interpolated residual by the standard deviation and adding the mean. The MSW map was created by back-transformation of the transformed MSW predictions. As a consequence, the expected value of the prediction after back transformation is not the mean, but the median (Pebesma and De Kwaadsteniet, 1997). Figure 3.28 shows a map of predicted MSW for the Netherlands.

3.4.5 Simulation

Conditional sequential Gaussian simulation based on simple kriging was performed to simulate three possible realities of MSW residuals. Next transformed MSW values were computed using Eq. (2.1). After back-transformation the three possible realities of MSW were plotted (Figure 3.29). Note that these possible realities of the MSW do not suffer from the mean-median difficulties observed before in the kriging predictions.

3.4.6 Evaluation

The original and simulated MSW data of 500 locations were compared. Figure 3.30 gives the histograms of the original and simulated, transformed data. Table 3.5 summarizes some basic statistics for the original and simulated, untransformed data. Table 3.5 indicates large differences of basic statistics between original and simulated data. These large differences can be explained from the different proportions of individual soil types in the original data set and the 500 locations used in the evaluation.

Table 3.2: Count of records for each soil category.

Soil type	S1(SP)	S2 (SR)	S3(SC)	S4(CN)	S5(CC)	S6(LN)	S7(PN)
Count	2507	2189	163	1161	827	130	685

SP = Sand Poor; SR = Sand Rich; SC = Sand Calcareous; CN = Clay Normal (Non-calcareous); CC = Clay Calcareous; LN = Loess Normal; PN = Peat Normal

Table 3.3: Descriptive statistics of MSW and transformed MSW data.

Statistics	MSW (cm)	t_MSW
Min	-6.55	0
Max	2001.00	7.65
Median	67.69	4.32
Mean	87.86	4.36
Variance	8237.90	0.34
Standard deviation	90.76	0.58

Table 3.4: Selective statistics for each of the seven soil types.

Soil type	S1(SP)	S2 (SR)	S3(SC)	S4(CN)	S5(CC)	S6(LN)	S7(PN)
Mean	4.52	4.42	4.53	4.14	4.43	5.32	3.66
Variance	0.29	0.30	0.55	0.18	0.11	0.75	0.18
St. dev.	0.54	0.55	0.74	0.42	0.33	0.87	0.43

SP = Sand Poor; SR = Sand Rich; SC = Sand Calcareous; CN = Clay Normal (Non-calcareous); CC = Clay Calcareous; LN = Loess Normal; PN = Peat Normal

Table 3.5: Basic statistics for original and simulated MSW data (500 locations)

Statistic	original data	simulation 1	simulation 2	simulation 3
Min	-6.55	1.13	6.39	3.38
Max	2001.00	563.44	840.32	1185.04
Median	67.69	57.00	52.21	52.75
Mean	87.86	74.13	70.13	68.91
Variance	8237.90	4321.98	5081.00	5205.32
Standard deviation	90.76	65.74	71.28	72.15

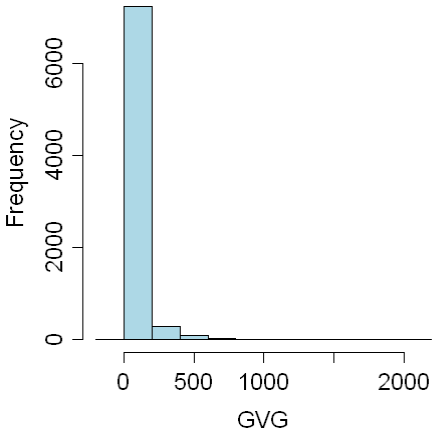


Figure 3.19: Histogram of original MSW data.

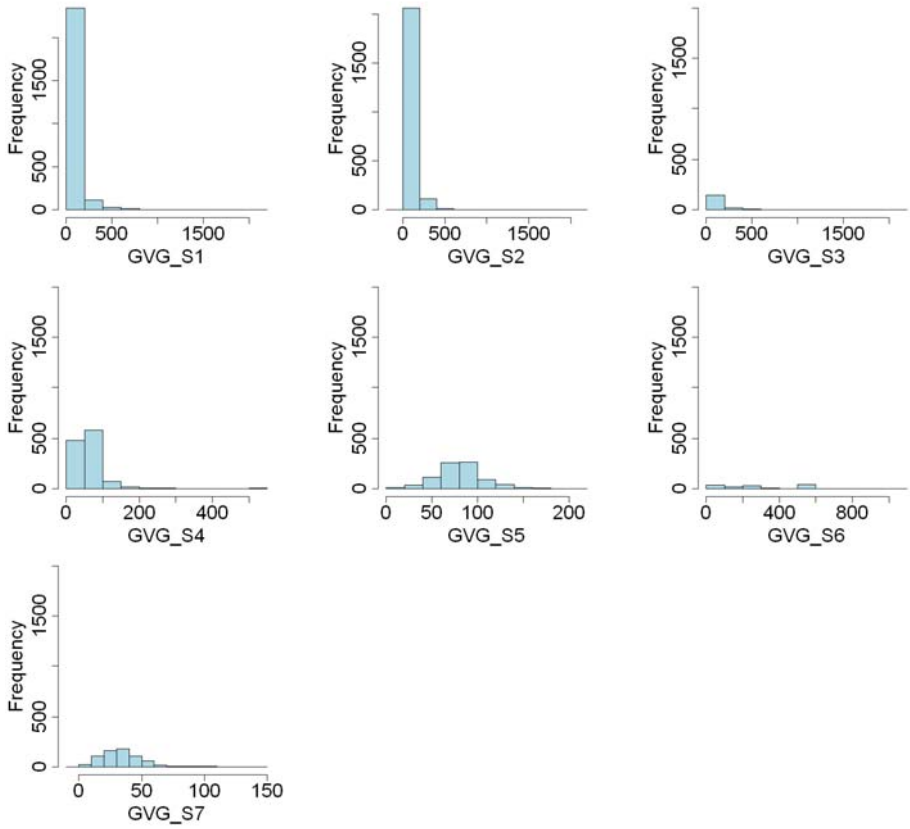


Figure 3.20: MSW histograms for each of the seven soil categories.

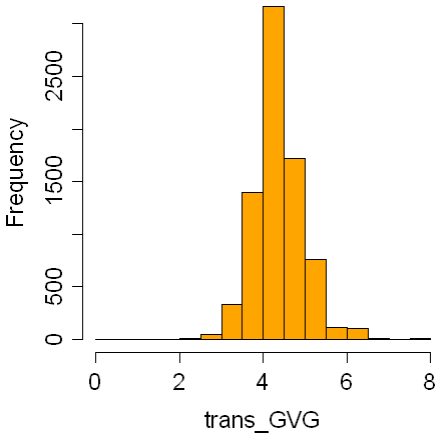


Figure 3.21: Histogram of log-transformed MSW data.

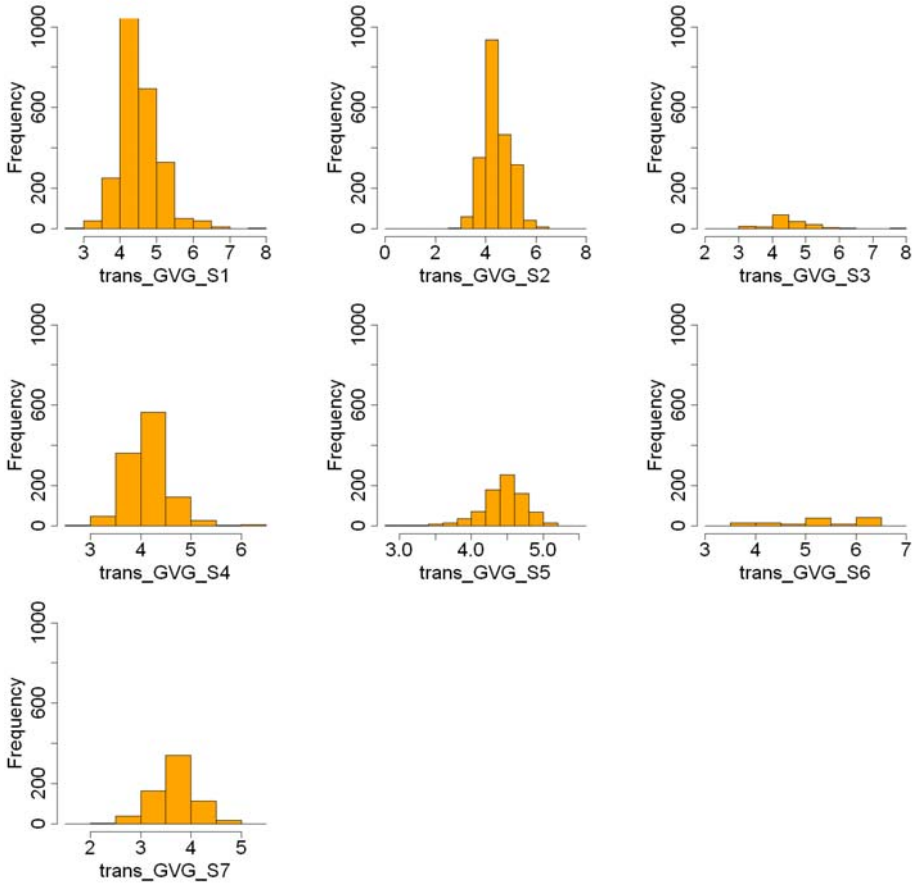


Figure 3.22: Histograms of log-transformed MSW data for each soil type.

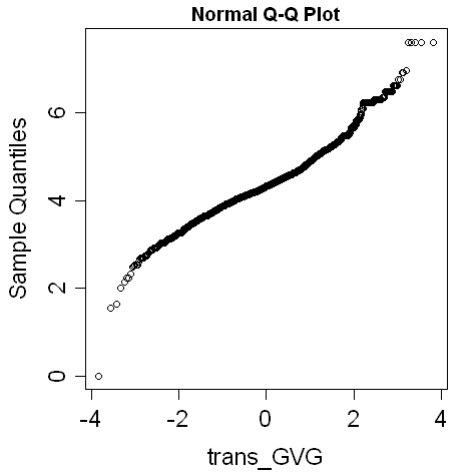


Figure 3.23: Normal Q-Q plot for transformed MSW data.

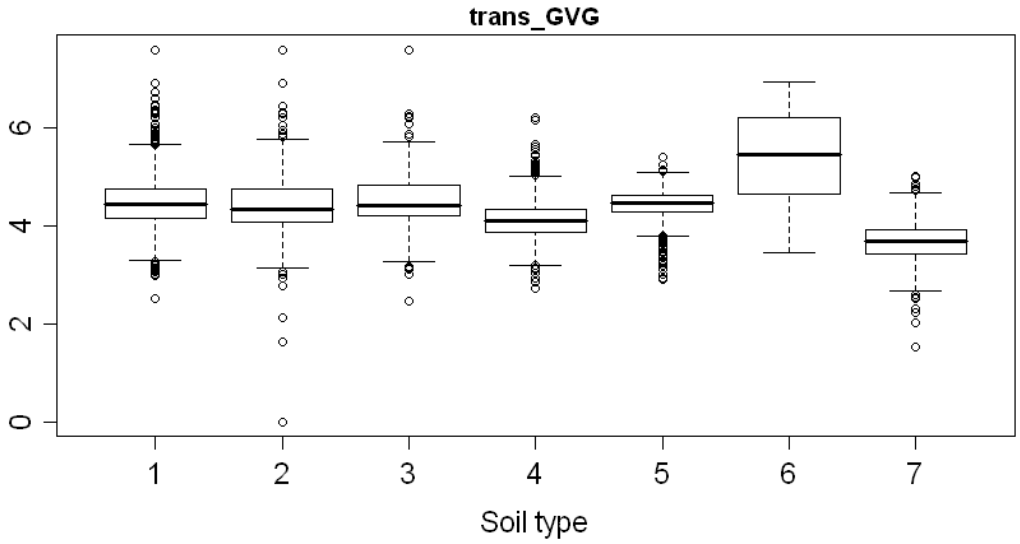
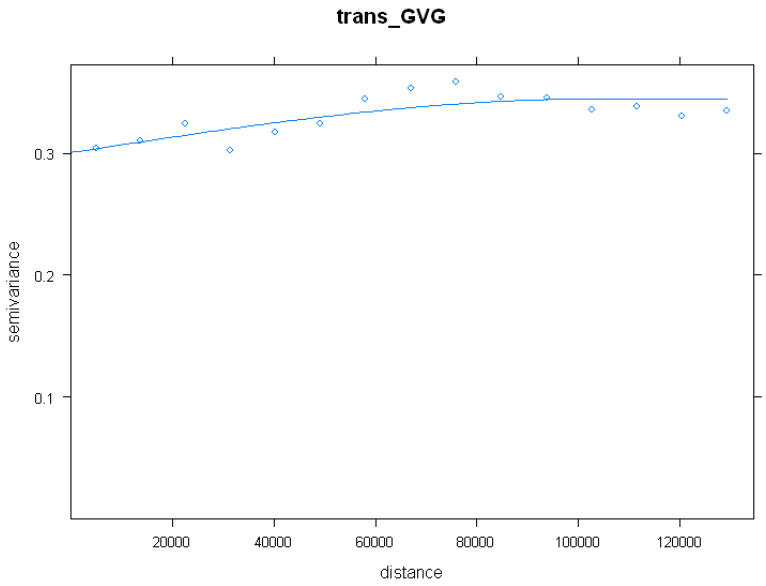
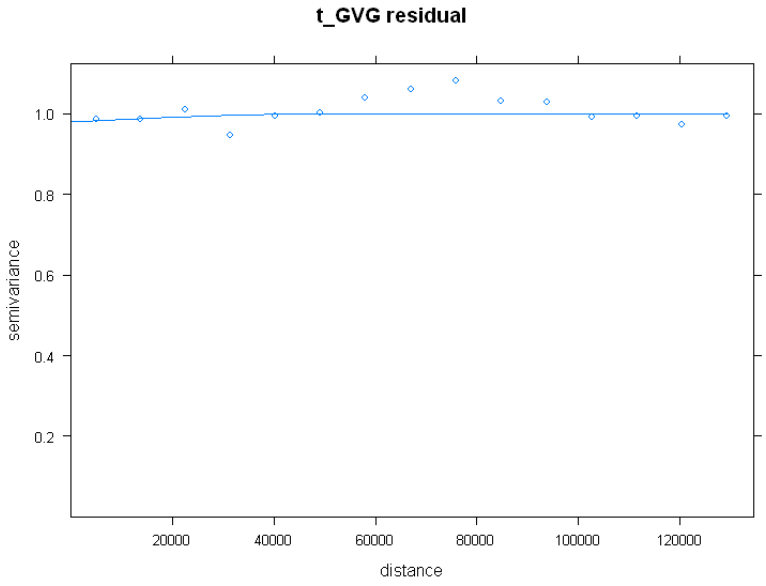


Figure 3.24: Box-plots of log-transformed MSW data for each of the seven soil types.



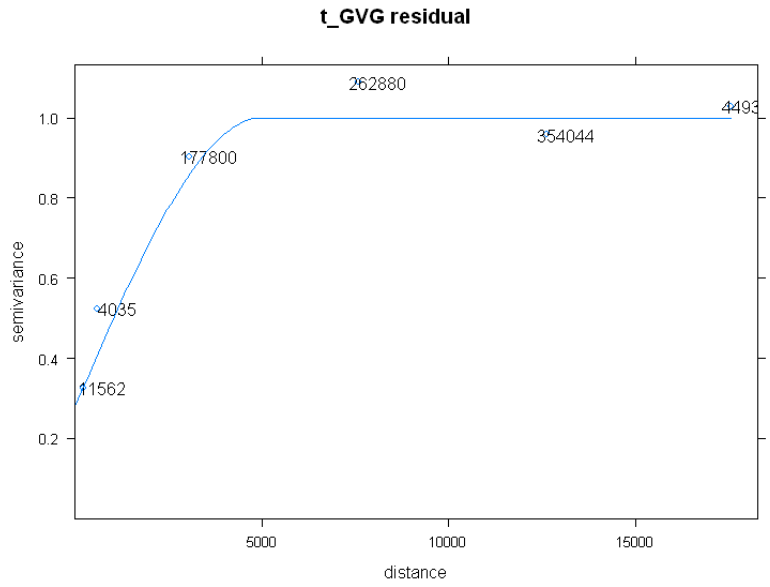
Nugget = 0.30
Psill = 0.044
Range = 102919
Model = Spherical

Figure 3.25: Variogram of transformed MSW data.



Nugget = 0.98
Psill = 0.02
Range = 50000
Model = Spherical

Figure 3.26: Variogram of residuals.



Nugget = 0.28
Psill = 0.72
Range = 5000
Model = Spherical

Figure 3.27: Variogram of residuals zoomed in to the shorter distances.

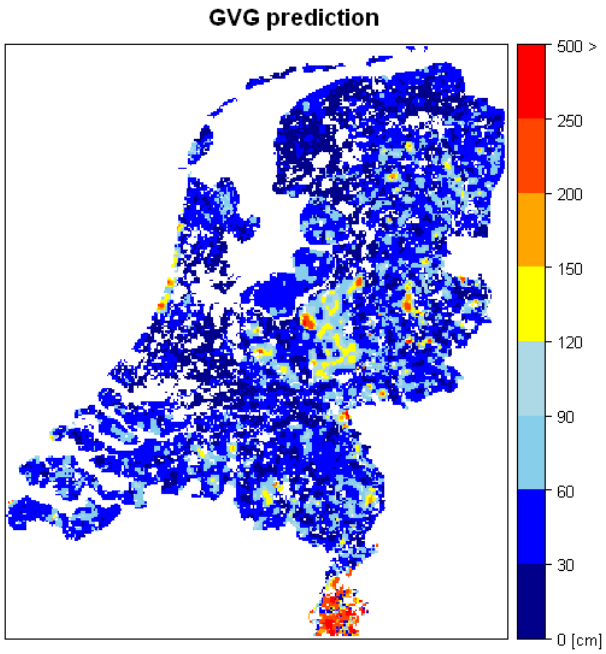


Figure 3.28: Prediction map of MSW data for the Netherlands.

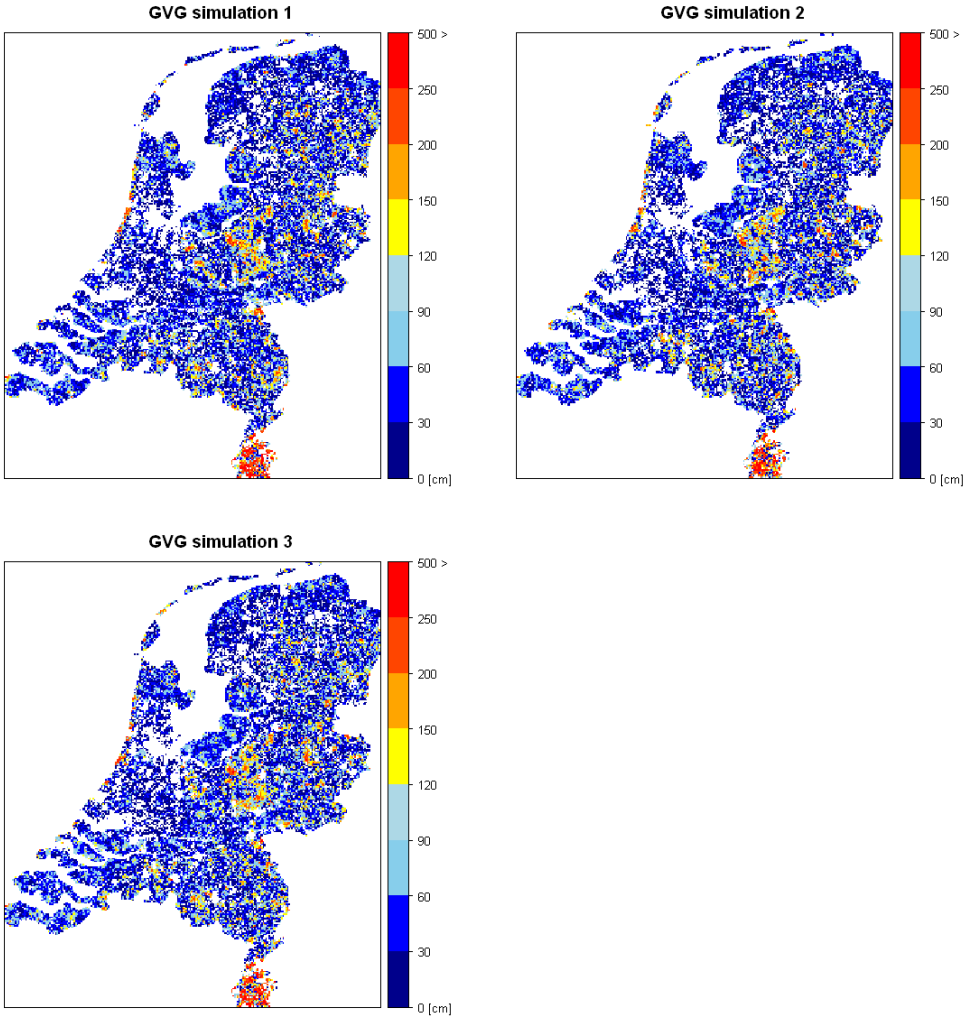


Figure 3.29: Three possible realities of MSW spatial distribution simulated over the Netherlands.

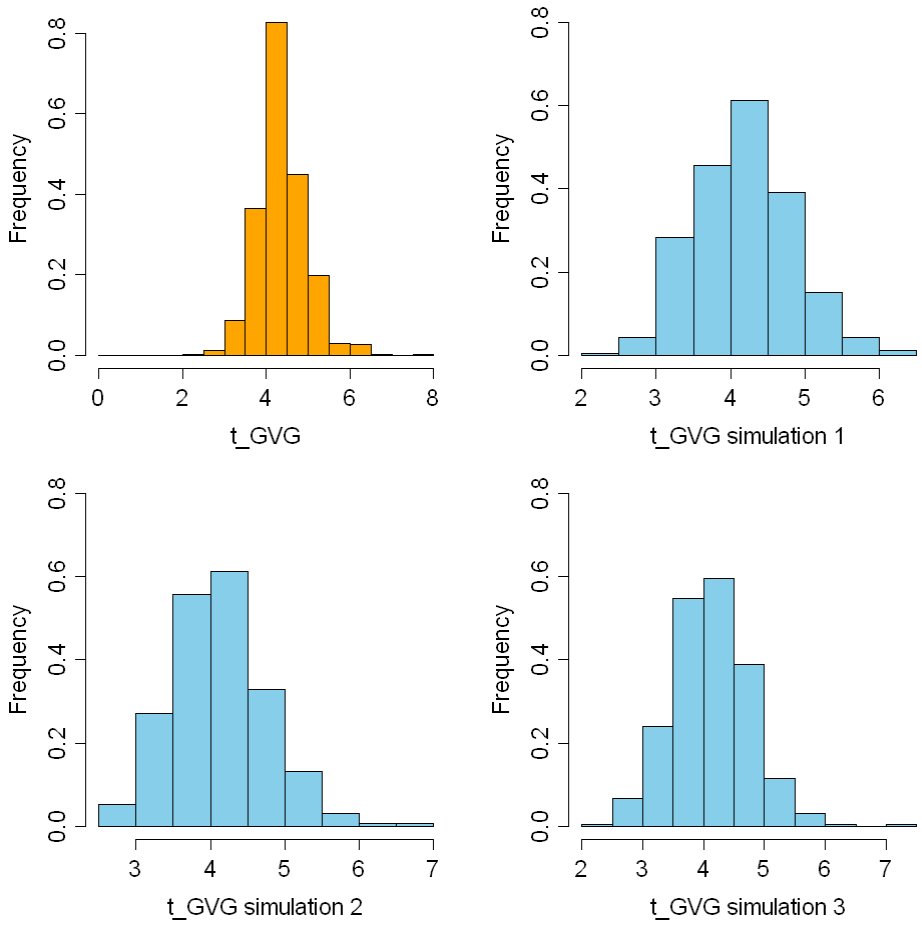


Figure 3.30: Comparison of transformed original and simulated MSW data at 500 target locations. The y axis represents frequency of probability distribution.

Chapter 4

Conclusions

The geostatistical framework and R codes presented in this report enable to predict values of continuous soil properties spatially, and to quantify the inaccuracy of these predictions. The inaccuracy of a spatial prediction at a certain location is quantified by the kriging variance, which can be interpreted as an indication of the uncertainty about the true value. A map of kriging variances is very useful in optimizing observation networks, because it indicates areas where predictions are relatively inaccurate. It should be noted, however, that the kriging variance reflects the inaccuracy given a *model* of spatial structure. Uncertainty about the true spatial structure is not accounted for. Therefore, the kriging variance is an approximation of the uncertainty about the true values of soil properties at unvisited locations. This makes the kriging variance less valuable in the decision processes mentioned in Chapter 1. Results of additional validation studies can be used to adjust the kriging variances to a level that approximates the uncertainty about the true values of soil properties more closely. This would be a valuable next step in the development of a δ-BIS that provides information to be applied in statistical decision making.

Besides predictions the presented software enable to simulate large numbers possible realities of spatial distributions of continuous soil properties. These simulations are very useful to visualize uncertainty, and can be used as input in sensitivity and uncertainty analyses. As with the kriging variance, model uncertainty of spatial structure is not accounted for in the simulations. Also the simulations could reflect the uncertainty more closely if results of an additional validation were used in the simulation procedure. Nevertheless, the current software tool is an important first step in informing decision makers about uncertainty.

Bibliography

Back, P.-E. (2007). A model for estimating the value of sampling programs and the optimal number of samples for contaminated soil. *Environmental Geology*, 52:573–585.

Brus, D. J. and Heuvelink, G. B. M. (2007). Towards a soil information system with quantified accuracy : three approaches for stochastic simulation of soil maps. Technical report, Wettelijke Onderzoekstaken Natuur en Milieu.

Commission of the European Communities (2000). Communication from the Commission on the precautionary principle; COM(2000) 1 final.

R Development Core Team (2009). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Freeze, R. A., James, B., Massmann, J., Sperling, T., and Smith, L. (1992). Hydrogeological decision analysis: 4. The concept of data worth and its use in the development of site investigation strategies. *Ground Water*, 30:574–588.

Goovaerts, P. (1997). *Geostatistics for natural reources evaluation*. Geostatistics for natural resources evaluation. Oxford University Press; Applied Geostatistics Series.

Isaaks, E. H. and Srivastava, R. H. (1989). *Applied geostatistics*. Oxford University Press, New York [etc.].

Minister van Volkshuisvesting, Ruimtelijke Ordening en Milieubeheer (2001). *Strategienota omgaan met stoffen. Nationaal en internationaal stoffenbeleid.* SDU-Uitgevers, Den Haag.

Morgan, M. G. and Henrion, M. (1990). *Uncertainty. A guide to dealing with uncertainty in quantitative risk and policy analysis*. Cambridge University Press, Cambridge.

Pebesma, E. J. (2004). Multivariable geostatistics in S: the gstat package. *Computers and Geosciences*, 30(7):683–691.

Pebesma, E. J. and De Kwaadsteniet, J. W. (1997). Mapping groundwater quality in the Netherlands. *Journal of Hydrology*, 200:364–386.

R-SIG-DB (2009). *DBI: R Database Interface*. R package version 0.2-4.

United Nations (1992). Rio declaration on environment and development of United Nations Conference on Environment and Development.

Urbanek, S. (2009). *RJDBC: Provides access to databases through the JDBC interface*. R package version 0.1-5.

Van der Sluijs, P. (1990). Grondwatertrappen. In Locher, W. P. and De Bakker, H., editors, *Bodemkunde van Nederland. Algemene bodemkunde.*, volume 1, pages 167–180. Malmberg, Den Bosch.

Visschers, R., Finke, P. A., and Gruijter, J. J. d. (2007). A soil sampling program for the Netherlands. *Geoderma*, 139:60–72.

Wösten, J. H. M., De Vries, F., and Denneboom, J. (1988). Generalisatie en bodemfysische vertaling van de bodemkaart van Nederland, 1 : 250.000, ten behoeve van de PAWN-studie. Technical Report 2055, Stichting voor Bodemkartering.

Appendix A

R code for pH_KCl in the soil layer 0 - 25 cm

Note:
Before the running the R code, the connection between your local machine and Oracle database has to be established and the following packages have to be installed in R environment: RODBC, sp, gstat, maptools, foreign, lattice. Furthermore, the PAWN soil types map (pawn-map08.shp) and the Netherlands boundary (NL.shp) have to be present in the working directory. The R code will yield several figures which will be stored in the actual working directory, these are as following: variogram model, prediction map, 95 % upper limit map, 95 % lower limit map, simulation maps, standard deviation of the prediction map, box-plots).

```
# The code starts here

# 00 Set parametres

d1 <- 0         # upper depth
d2 <- 25        # lower depth
cs <- 500       # interpolation grid cell size
ns <- 10        # number of simulations

# 01 Libraries

library(RODBC)
library(foreign)
library(sp)
library(maptools)
library(gstat)
library(lattice)

# 02 Data extraction - PFB


# Connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# Fetch all records from V_pfb_deltabis view
pfb <- sqlFetch(channel = channel, sqtable = "V_PFB_DELTABIS")

# Disconnect from database
odbcClose(channel = channel)
rm(channel)

# 03 Exclude samples collected from more than 1 soil horizon - PFB

isCrossing <- (!is.na(pfb$MON_VNR)  &
```

```
    ((pfb$MON_DIEPB < pfb$HOR_DIEPB) & (pfb$MON_DIEPO > pfb$HOR_DIEPB)) |
    ((pfb$MON_DIEPB < pfb$HOR_DIEPO) & (pfb$MON_DIEPO > pfb$HOR_DIEPO))
)

pfb <- pfb[!isCrossing, ]
rm(isCrossing)

# 04 H activity - PFB

pfb$H <- 10^-pfb$PH_KCL

# 05 Soil Layers weighted averaging of H+ activity - PFB

# Subset removing NA records in PH_KCL column
pfb <- subset(x = pfb, subset = !is.na(pfb$H))

# Weighted averiging of H activity
pfb$s_thick <- pfb$MON_DIEPO - pfb$MON_DIEPB
pfb$s_thick.H <- pfb$s_thick * pfb$H

pfb$HOR_ID <- as.character(pfb$LAAG_ID)

sum1 <- tapply(X=pfb$s_thick.H,INDEX=pfb$HOR_ID,FUN=sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X=pfb$s_thick,INDEX=pfb$HOR_ID,FUN=sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_H <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
pfb <- subset(pfb,duplicated(pfb$LAAG_ID)==FALSE)

# 06 Horizons selection - PFB

pfb <- subset(pfb,
    (HOR_DIEPB> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB< d1 & HOR_DIEPO> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB> d1 & HOR_DIEPB< d2 & HOR_DIEPO> d2) |
    (HOR_DIEPB<=d1 & HOR_DIEPO>=d2) )

# Computing the thickness of a horizon part which contribute to the required
# soil layer
pfb$h_thick <- 0

s1 <- ((pfb$HOR_DIEPB > d1) & (pfb$HOR_DIEPO< d2))
pfb[s1,]$h_thick <- pfb[s1,]$HOR_DIEPO - pfb[s1,]$HOR_DIEPB

s2 <- ((pfb$HOR_DIEPB < d1) & (pfb$HOR_DIEPO > d1) & (pfb$HOR_DIEPO < d2))
pfb[s2,]$h_thick <- pfb[s2,]$HOR_DIEPO - d1

s3 <- ((pfb$HOR_DIEPB > d1) & (pfb$HOR_DIEPB < d2) & (pfb$HOR_DIEPO > d2))
pfb[s3,]$h_thick <- d2 - pfb[s3,]$HOR_DIEPB

s4 <- ((pfb$HOR_DIEPB <= d1) & (pfb$HOR_DIEPO >= d2))
pfb[s4,]$h_thick <- d2 - d1

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
pfb$ratio <- (pfb$h_thick/(d2-d1))

# The sum of ratios
pfb$PROF_ID <- as.character(pfb$PROFIEL_ID)
sum_ratio <- tapply(X=pfb$ratio,INDEX=pfb$PROF_ID,FUN=sum)
pfb$sum_ratio <- sum_ratio[pfb$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
pfb <- subset(pfb,sum_ratio > 0.999)

# 07 Weighted averiging for the defined depth - PFB
```

```
pfb$h_thick.H <- pfb$H*pfb$h_thick

sum1 <- tapply(X=pfb$h_thick.H,INDEX=pfb$PROF_ID,FUN=sum)
pfb$sum1 <- sum1[pfb$PROF_ID]

sum2 <- tapply(X=pfb$h_thick,INDEX=pfb$PROF_ID,FUN=sum)
pfb$sum2 <- sum2[pfb$PROF_ID]

pfb$wa_H <- pfb$sum1/pfb$sum2
rm(sum1,sum2)

# Transform to the log scale
pfb$wa_pH <- -log10(pfb$wa_H)

# Subset with unique soil profiles
pfb <- subset(pfb,duplicated(pfb$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X=pfb$X,Y=pfb$Y)
dup2 <- duplicated(dup)

pfb$dupl_loc <- dup2
rm(dup,dup2)

pfb <- subset(pfb,dupl_loc==FALSE)

# Available PH_KCL data for the depth d1-d2
pfb <- data.frame(X=pfb$X,Y=pfb$Y,pH=pfb$wa_pH)


# 08 Adding pawn soil types - PFB

# Read the pawn_map08.shp file

map <- readShapePoly("pawn-map08.shp",IDvar="PAWN_ID")

# Overlay sampling locaations with pawn-map
coordinates(pfb)=~X+Y
o <- overlay(map,pfb)

pfb$pawn <- o$PAWN_CODE
rm(o)

# 09 Data extraction - LSK

# Connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# Fetch all records from V_lsk_deltabis view
lsk <- sqlFetch(channel = channel, sqtable = "V_LSK_DELTABIS")

# Disconnect from database
odbcClose(channel = channel)
rm(channel)

# 10 Exclude samples collected from more than 1 soil horizon - LSK

isCrossing <- (!is.na(lsk$MON_VNR)  &
    ((lsk$MON_DIEPB < lsk$HOR_DIEPB) & (lsk$MON_DIEPO > lsk$HOR_DIEPB)) |
    ((lsk$MON_DIEPB < lsk$HOR_DIEPO) & (lsk$MON_DIEPO > lsk$HOR_DIEPO))
)

lsk <- lsk[!isCrossing, ]
rm(isCrossing)

# 11 H activity - LSK

lsk$H <- 10^-lsk$PH_KCL

# 12 Soil Layers weighted averaging of H activity - LSK

# Subset removing NA records in PH_KCL column
lsk <- subset(x = lsk, subset = !is.na(lsk$H))
```

```
# Weighted averiging of H activity
lsk$s_thick <- lsk$MON_DIEPO - lsk$MON_DIEPB
lsk$s_thick.H <- lsk$s_thick * lsk$H

lsk$HOR_ID <- as.character(lsk$LAAG_ID)

sum1 <- tapply(X=lsk$s_thick.H,INDEX=lsk$HOR_ID,FUN=sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X=lsk$s_thick,INDEX=lsk$HOR_ID,FUN=sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_H <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
lsk <- subset(lsk,duplicated(lsk$LAAG_ID)==FALSE)

# 13 Horizons selection - LSK

lsk <- subset(lsk,
    (HOR_DIEPB> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB< d1 & HOR_DIEPO> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB> d1 & HOR_DIEPB< d2 & HOR_DIEPO> d2) |
    (HOR_DIEPB<=d1 & HOR_DIEPO>=d2) )

# Computing the thickness of a horizon part which contribute to the required
# soil layer
lsk$h_thick <- 0

s1 <- ((lsk$HOR_DIEPB > d1) & (lsk$HOR_DIEPO< d2))
lsk[s1,]$h_thick <- lsk[s1,]$HOR_DIEPO - lsk[s1,]$HOR_DIEPB

s2 <- ((lsk$HOR_DIEPB < d1) & (lsk$HOR_DIEPO > d1) & (lsk$HOR_DIEPO < d2))
lsk[s2,]$h_thick <- lsk[s2,]$HOR_DIEPO - d1

s3 <- ((lsk$HOR_DIEPB > d1) & (lsk$HOR_DIEPB < d2) & (lsk$HOR_DIEPO > d2))
lsk[s3,]$h_thick <- d2 - lsk[s3,]$HOR_DIEPB

s4 <- ((lsk$HOR_DIEPB <= d1) & (lsk$HOR_DIEPO >= d2))
lsk[s4,]$h_thick <- d2 - d1

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
lsk$ratio <- (lsk$h_thick/(d2-d1))

# The sum of ratios
lsk$PROF_ID <- as.character(lsk$PROFIEL_ID)
sum_ratio <- tapply(X=lsk$ratio,INDEX=lsk$PROF_ID,FUN=sum)
lsk$sum_ratio <- sum_ratio[lsk$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
lsk <- subset(lsk,sum_ratio > 0.999)


# 14 Weighted averiging for the defined depth - LSK

lsk$h_thick.H <- lsk$H*lsk$h_thick

sum1 <- tapply(X=lsk$h_thick.H,INDEX=lsk$PROF_ID,FUN=sum)
lsk$sum1 <- sum1[lsk$PROF_ID]

sum2 <- tapply(X=lsk$h_thick,INDEX=lsk$PROF_ID,FUN=sum)
lsk$sum2 <- sum2[lsk$PROF_ID]

lsk$wa_H <- lsk$sum1/lsk$sum2
rm(sum1,sum2)

# Transform to the log scale
lsk$wa_pH <- -log10(lsk$wa_H)

# Subset with unique soil profiles
```

```
lsk <- subset(lsk,duplicated(lsk$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X=lsk$X,Y=lsk$Y)
dup2 <- duplicated(dup)

lsk$dupl_loc <- dup2
rm(dup,dup2)

lsk <- subset(lsk,dupl_loc==FALSE)

# Available PH_KCL data for the depth d1-d2
lsk <- data.frame(X=lsk$X,Y=lsk$Y,pH=lsk$wa_pH)


# 15 Adding pawn soil types - LSK

# Overlay sampling locations with pawn-map
coordinates(lsk)=~X+Y
o <- overlay(map,lsk)

lsk$pawn <- o$PAWN_CODE
rm(o)

# 16 Merging two tables

ph <- merge(pfb,lsk,all=TRUE)

# Exlude locations which fall into the water,build up aread, and NA
ph <- subset(ph,pawn<22)
rm(pfb,lsk)

# 17 Variography

# Residuals
mean <- tapply(ph$pH,ph$pawn,mean)
ph$mean <- mean[ph$pawn]

sd <- tapply(ph$pH,ph$pawn,sd)
ph$sd <- sd[ph$pawn]

ph$residua <- (ph$pH-ph$mean)/ph$sd


# Variogram
coordinates(ph)=~X+Y
gPH <- gstat(id=c("pH_residua"),formula=residua~1,data=ph)
gPH.vg <- variogram(gPH,boundaries=c(500,700,1000,1500,2500,10000,20000,30000,40000,
    50000,60000,70000,80000,90000,100000,110000,120000,130000,140000,150000))

# Fit nested variogram model
vgm <- fit.variogram(gPH.vg, vgm(psill= 1,model="Sph",range=100000,
add.to=vgm(psill= 1,model="Sph",range=4000,nugget=1)),fit.sill=TRUE,fit.ranges=TRUE)

# Save the residual variogram figure
png("variogram.png", width=600, height=600, units="px", pointsize=12, bg="white",
     res=NA)
fontsize <- trellis.par.get("fomtsize")
fontsize$text <- 20
fontsize$points <- 20
trellis.par.set("fontsize",fontsize)
plot(gPH.vg,vgm,main="pH_KCl - residual variogram",pch=20)
dev.off()

# 18 Define interpolation grid

nl_poly <- readShapePoly("NL.shp")

grid <- spsample(nl_poly,cellsize=cs,type="regular")
o <- overlay(map,grid)
pawn <- data.frame(pawn=o$PAWN_CODE)
grid <- SpatialPointsDataFrame(grid,pawn)
grid <- subset(grid,pawn < 22)
rm(o)
```

```
# 19 Simulations

# Define gstat object to predict residuals

gstatPH <- gstat(id=c("pH"),formula=residua~1,data=ph,model=vgm,beta=0,nmax=100)

# Ready to predict pH_KCl residuals using Gaussian simulation based on simple
# kriging
blup <- predict.gstat(gstatPH, newdata = grid, BLUE = FALSE,nsim=ns)

# Compute simulations
sim <- data.frame(blup,pawn=grid$pawn)

sim$mean <- mean[sim$pawn]

sim$sd <- sd[sim$pawn]

sim_ns <- data.frame(X=sim$x,Y=sim$y)

sim_ns[,3:(ns+2)] <- sim[,3:(ns+2)] * sim$sd + sim$mean

# Save figures
coordinates(sim_ns) <- ~X+Y
gridded(sim_ns) <- TRUE

text <- list("sp.text",c(60000,550000),paste(d1,"-",d2,"cm"),cex=1.5)
nl <- list("sp.polygons", nl_poly)

for (i in 1:ns) {
png(filename = paste("sim",i,".png",sep=""), width=600, height=600, units="px",
     pointsize=12, bg="white", res=NA)
print(spplot(sim_ns[,i], main = list(label=paste("pH_KCl simulation",i), cex=1.5),
col.regions=bpy.colors(),
at=c(0,seq(3,8,1/3),14),
colorkey=list(space="right",width=2,at=1:18,
labels=list(cex=2,at=1:18,labels=c("","3","","","4","","","5","","","6","","",
     "7","","","8",""))),
sp.layout=list(nl,text)))
dev.off()
}

# 20 Prediction

# Ready to make pH residual prediction using simple kriging
blup2 <- predict.gstat(gstatPH, newdata = grid, BLUE = FALSE)

# Compute prediction
pred <- data.frame(blup2,pawn=grid$pawn)

pred$mean <- mean[pred$pawn]

pred$sd <- sd[pred$pawn]
rm(mean,sd)

pred$pred <- pred$pH.pred * pred$sd + pred$mean

coordinates(pred) <- ~x+y
gridded(pred) <- TRUE

# Standard deviation map
pred$pred_sd <- sqrt(pred$pH.var)
pred$map_sd <- pred$sd * pred$pred_sd

# 95 lower limit map
pred$lower95 <- pred$pred - 1.96 * pred$map_sd

# 95 upper limit map
pred$upper95 <- pred$pred + 1.96 * pred$map_sd

# Save figures

png(filename = "pred.png", width=600, height=600, units="px", pointsize=12,
     bg="white", res=NA)
spplot(pred["pred"],main=list(label="pH_KCl prediction",cex=1.5),
```

```
col.regions=bpy.colors(),
at=c(0,seq(3,8,1/3),14),
colorkey=list(space="right",width=2,at=1:18,labels=list(cex=2,at=1:18,
labels=c("","3","","","4","","","5","","","6","","","7","","","8","","")),
sp.layout=list(nl,text))
dev.off()

# Save standard deviation map
png(filename="stdv_map.png", width=600, height=600, units="px", pointsize=12,
     bg="white", res=NA)
spplot(pred["map_sd"],main=list(label="pH_KCl standard deviation map",cex=1.5),
col.regions=bpy.colors(),
colorkey=list(space="right",width=2,labels=list(cex=2)),
sp.layout=list(nl,text))
dev.off()

# Save 95 lower limit map
png(filename="95lower_map.png", width=600, height=600, units="px", pointsize=12,
     bg="white", res=NA)
spplot(pred["lower95"],main=list(label="pH_KCl 95 lower limit map",cex=1.5),
col.regions=bpy.colors(),
at=c(0,seq(3,8,1/3),14),
colorkey=list(space="right",width=2,at=1:18,labels=list(cex=2,at=1:18,
labels=c("","3","","","4","","","5","","","6","","","7","","","8","","")),
sp.layout=list(nl,text))
dev.off()


# Save 95 upper limit map
png("95upper_map.png",width=600,height=600,units="px",pointsize=12,bg="white",
     res=NA)
spplot(pred["upper95"],main=list(label="pH_KCl 95 upper limit map",cex=1.5),
col.regions=bpy.colors(),
at=c(0,seq(3,8,1/3),14),
colorkey=list(space="right",width=2,at=1:18,labels=list(cex=2,at=1:18,
labels=c("","3","","","4","","","5","","","6","","","7","","","8","","")),
sp.layout=list(nl,text))
dev.off()

# 21 Box-plots
png(filename = "box_plot.png", width=800, height=600, units="px", pointsize=12,
     bg="white", res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
boxplot(split(ph$pH,ph$pawn),xlab="Pawn soil type",main="pH_KCl",cex.main=2)
dev.off()

# 22 Histograms

# histogram of original dataset
windows(width = 5,height = 5)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(ph$pH,main="",xlab="pH_KCl",col="lightblue")

# histogram of residuals
windows(width = 5,height = 5)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(ph$residua,main="",xlab="pH_KCl residuals",col="lightblue")

# histogram of original dataset with freq=FALSE
windows(width = 5,height = 5)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(ph$pH,main="",xlab="pH_KCl",col="lightblue",freq=FALSE,
     ylab="Frequency",ylim=c(0,0.6),xlim=c(0,12))

# histogram of predicted data
windows(width = 5,height = 5)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(pred$pred,main="",xlab="pH_KCl prediction",col="orange",freq=FALSE,
     ylab="Frequency",ylim=c(0,0.6),xlim=c(0,12))

# histogram of simulated pH_KCl data, simulation 1
windows(width = 5,height = 5)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(sim10$sim1,main="",xlab="pH_KCl simulation 1",col="orange",freq=FALSE,
     ylab="Frequency",ylim=c(0,0.6),xlim=c(0,12))
```

```
# The end of the code
```

Appendix B

R code Clay Content, single layer analysis

B.1 Data extraction and preparation

```
# Data extraction and preparation

###### Set parameters #################
# Load libraries

library(RODBC)
library(foreign)
library(sp)
library(maptools)
library(gstat)
library(lattice)

# The thickness of soil layer (cm)
d1 <- 10
d2 <- 15

# Data filtering (spatial extent of input data)
X_MIN <- 0
X_MAX <- 300000
Y_MIN <- 300000
Y_MAX <- 620000

# Time filtering: yyyy-mm-dd

from <- as.POSIXct("1000-01-01")
to   <- as.POSIXct("3000-01-01")


#######################################

# View extraction - V_AUGERING_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_augering_deltabis view
augering <- sqlFetch(channel = channel, sqtable = "V_AUGERING_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - Augering
augering$MON_DATUM <- paste(augering$MON_DATUM,"-01-01",sep="")
```

```
augering$MON_DATUM <- as.POSIXct(augering$MON_DATUM)

augering <- subset(augering,(!is.na(augering$MON_DATUM) & augering$MON_DATUM >
     from & augering$MON_DATUM < to))

# Spatial extent filtering - Augering
augering <- subset(augering,(!is.na(augering$X) & augering$X > X_MIN & augering$X < X_MAX))
augering <- subset(augering,(!is.na(augering$Y) & augering$Y > Y_MIN & augering$Y < Y_MAX))

# Horizons selection - Augering

# Exclude records with missing clay content values
augering <- subset(x = augering, subset = !is.na(LUTUM))

augering <- subset(x = augering,subset =
    (HOR_DIEPB > d1 & HOR_DIEPO < d2) |
    (HOR_DIEPB < d1 & HOR_DIEPO > d1 & HOR_DIEPO < d2) |
    (HOR_DIEPB > d1 & HOR_DIEPB < d2 & HOR_DIEPO > d2) |
    (HOR_DIEPB <= d1 & HOR_DIEPO >= d2) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
augering$h_thick <- 0

s1 <- ((augering$HOR_DIEPB > d1) & (augering$HOR_DIEPO < d2))
augering[s1,]$h_thick <- augering[s1,]$HOR_DIEPO - augering[s1,]$HOR_DIEPB

s2 <- ((augering$HOR_DIEPB < d1) & (augering$HOR_DIEPO > d1) & (augering$HOR_DIEPO < d2))
augering[s2,]$h_thick <- augering[s2,]$HOR_DIEPO - d1

s3 <- ((augering$HOR_DIEPB > d1) & (augering$HOR_DIEPB < d2) & (augering$HOR_DIEPO > d2))
augering[s3,]$h_thick <- d2 - augering[s3,]$HOR_DIEPB

s4 <- ((augering$HOR_DIEPB <= d1) & (augering$HOR_DIEPO >= d2))
augering[s4,]$h_thick <- d2 - d1

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
augering$ratio <- (augering$h_thick/(d2-d1))

# The sum of ratios
augering$PROF_ID <- as.character(augering$PROFIEL_ID)
sum_ratio <- tapply(X = augering$ratio, INDEX = augering$PROF_ID, FUN = sum)
augering$sum_ratio <- sum_ratio[augering$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
augering <- subset(x = augering,subset = sum_ratio > 0.999)

# Bulk density - Augering

# Split the table into two dataset based on geocode value
# to apply two different aproaches for bulk density computation

augering1 <- subset(x = augering, subset = ((GEO_FOR_C > 0) & (augering$GEO_FOR_C <= 690)))

augering2 <- subset(x = augering, subset = ((GEO_FOR_C > 690) |
     (augering$GEO_FOR_C == 0) | (is.na(augering$GEO_FOR_C))))
rm(augering)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")
top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
augering1$bulk <- NA
augering1$GEO_FOR_C <- as.character(augering1$GEO_FOR_C)
t <- (!is.na(augering1$A_HORIZON))
augering1[t,]$bulk <- top[augering1[t,]$GEO_FOR_C]
augering1[!t,]$bulk <- sub[augering1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)
```

```
# The bulk density for the second subset
augering2$bulk <- NA
augering2 <- subset(augering2,!is.na(augering2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.4

s2 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.8

s4 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.65

s5 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 50) )
augering2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.5

s2 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.65

s4 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.6

s5 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 50))
augering2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
augering <- merge(augering1,augering2,all=TRUE)
rm(augering1,augering2)

# Profile weighted averaging - Augering

augering$SoilMass <- augering$h_thick * augering$bulk
augering$SoilMass.LUTUM <- augering$SoilMass * augering$LUTUM

sum1 <- tapply(X = augering$SoilMass.LUTUM, INDEX = augering$PROF_ID, FUN = sum)
augering$sum1 <- sum1[augering$PROF_ID]

sum2 <- tapply(X = augering$SoilMass, INDEX = augering$PROF_ID, FUN = sum)
augering$sum2 <- sum2[augering$PROF_ID]

augering$wa_clay <- augering$sum1/ augering$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
augering <- subset(x = augering, subset = duplicated(augering$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = augering$X, Y = augering$Y)
dup2 <- duplicated(dup)

augering$dupl_loc <- dup2
rm(dup,dup2)

augering <- subset(x = augering, subset = (dupl_loc==FALSE))
```

```
# All available clay content data for the depth d1 - d2
augering <- data.frame(X = augering$X, Y = augering$Y, clay = augering$wa_clay)

# Add pawn soil types - Augering

# read the pawn_map08.shp file
map <- readShapePoly("pawn-map08.shp",IDvar="PAWN_ID")

# overlay sampling locaations with pawn-map
coordinates(augering) <- ~X+Y
o <- overlay(map,augering)

augering$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_PFB_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_pfb_deltabis view
pfb <- sqlFetch(channel = channel, sqtable = "V_PFB_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$MON_DATUM) & pfb$MON_DATUM > from &
     pfb$MON_DATUM < to))

# Spatial extent filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$X) & pfb$X > X_MIN & pfb$X < X_MAX))
pfb <- subset(pfb,(!is.na(pfb$Y) & pfb$Y > Y_MIN & pfb$Y < Y_MAX))

# Bulk density - PFB

# Split the table into two dataset based on geocode value
# to apply two different aproaches for bulk density computation

pfb1 <- subset(pfb,((GEO_FOR_C > 0) & (pfb$GEO_FOR_C <= 690)))

pfb2 <- subset(pfb,((GEO_FOR_C > 690) | (pfb$GEO_FOR_C == 0) |
     (is.na(pfb$GEO_FOR_C))))
rm(pfb)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")
top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
pfb1$bulk <- NA
pfb1$GEO_FOR_C <- as.character(pfb1$GEO_FOR_C)
t <- (!is.na(pfb1$A_HORIZON))
pfb1[t,]$bulk <- top[pfb1[t,]$GEO_FOR_C]
pfb1[!t,]$bulk <- sub[pfb1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
pfb2$bulk <- NA
pfb2 <- subset(pfb2,!is.na(pfb2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.4

s2 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.8
```

```
s4 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.65

s5 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50) )
pfb2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.5

s2 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.65

s4 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.6

s5 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50))
pfb2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
pfb <- merge(pfb1,pfb2,all=TRUE)
rm(pfb1,pfb2)

# Exclude samples collected from more than 1 soil horizon - PFB

isCrossing <- (!is.na(pfb$MON_VNR)  &
    ((pfb$MON_DIEPB < pfb$HOR_DIEPB) & (pfb$MON_DIEPO > pfb$HOR_DIEPB)) |
    ((pfb$MON_DIEPB < pfb$HOR_DIEPO) & (pfb$MON_DIEPO > pfb$HOR_DIEPO))
)

pfb <- pfb[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - PFB

# Compute weighted average of organic matter and bulk density for each soil horizon
# (important where there are more samples which belong to the same horizon)

# Subset removing NA records in LUTUM column
pfb <- subset(x = pfb,subset = !is.na(pfb$LUTUM))

# Weighted averiging of organic carbon content

pfb$s_thick <- pfb$MON_DIEPO - pfb$MON_DIEPB
pfb$s_thick.clay <- pfb$s_thick * pfb$LUTUM

pfb$HOR_ID <- as.character(pfb$LAAG_ID)

sum1 <- tapply(X = pfb$s_thick.clay, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_clay <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Weighted averaging of bulk density

pfb$s_thick.bulk <- pfb$s_thick * pfb$bulk

sum1 <- tapply(X = pfb$s_thick.bulk, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_bulk <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)
```

```
# Remove duplicated layers - no more needed
pfb <- subset(x = pfb,subset = duplicated(pfb$LAAG_ID)==FALSE)

# Horizons selection - PFB

pfb <- subset(x = pfb,subset =
    (HOR_DIEPB> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB< d1 & HOR_DIEPO> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB> d1 & HOR_DIEPB< d2 & HOR_DIEPO> d2) |
    (HOR_DIEPB<=d1 & HOR_DIEPO>=d2) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
pfb$h_thick <- 0

s1 <- ((pfb$HOR_DIEPB > d1) & (pfb$HOR_DIEPO < d2))
pfb[s1,]$h_thick <- pfb[s1,]$HOR_DIEPO - pfb[s1,]$HOR_DIEPB

s2 <- ((pfb$HOR_DIEPB < d1) & (pfb$HOR_DIEPO > d1) & (pfb$HOR_DIEPO < d2))
pfb[s2,]$h_thick <- pfb[s2,]$HOR_DIEPO - d1

s3 <- ((pfb$HOR_DIEPB > d1) & (pfb$HOR_DIEPB < d2) & (pfb$HOR_DIEPO > d2))
pfb[s3,]$h_thick <- d2 - pfb[s3,]$HOR_DIEPB

s4 <- ((pfb$HOR_DIEPB <= d1) & (pfb$HOR_DIEPO >= d2))
pfb[s4,]$h_thick <- d2 - d1

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
pfb$ratio <- (pfb$h_thick/(d2-d1))

# The sum of ratios
pfb$PROF_ID <- as.character(pfb$PROFIEL_ID)
sum_ratio <- tapply(X = pfb$ratio, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum_ratio <- sum_ratio[pfb$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
pfb <- subset(x = pfb,subset = sum_ratio > 0.999)

# Profile weighted averaging - PFB

pfb$SoilMass <- pfb$h_thick * pfb$l_bulk
pfb$SoilMass.l_clay <- pfb$SoilMass * pfb$l_clay

sum1 <- tapply(X = pfb$SoilMass.l_clay, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$PROF_ID]

sum2 <- tapply(X = pfb$SoilMass, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$PROF_ID]

pfb$wa_clay <- pfb$sum1/ pfb$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
pfb <- subset(x = pfb, subset = duplicated(pfb$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = pfb$X, Y = pfb$Y)
dup2 <- duplicated(dup)

pfb$dupl_loc <- dup2
rm(dup,dup2)

pfb <- subset(x = pfb, subset = (dupl_loc==FALSE))

# All available clay content data for the depth d1 - d2
pfb <- data.frame(X = pfb$X, Y = pfb$Y, clay = pfb$wa_clay)

# Pawn soil types - PFB

# overlay sampling locaations with pawn-map
```

```
coordinates(pfb) <- ~ X+Y
o <- overlay(map,pfb)

pfb$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_LSK_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_lsk_deltabis view
lsk <- sqlFetch(channel = channel, sqtable = "V_LSK_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$MON_DATUM) & lsk$MON_DATUM > from &
     lsk$MON_DATUM < to))

# Spatial extent filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$X) & lsk$X > X_MIN & lsk$X < X_MAX))
lsk <- subset(lsk,(!is.na(lsk$Y) & lsk$Y > Y_MIN & lsk$Y < Y_MAX))

# Bulk density - LSK

# Split the table into two dataset based on geocode value
# to apply two different aproaches for bulk density computation

lsk1 <- subset(lsk,((GEO_FOR_C > 0) & (lsk$GEO_FOR_C <= 690)))

lsk2 <- subset(lsk,((GEO_FOR_C > 690) | (lsk$GEO_FOR_C == 0) |
     (is.na(lsk$GEO_FOR_C))))
rm(lsk)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")
top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
lsk1$bulk <- NA
lsk1$GEO_FOR_C <- as.character(lsk1$GEO_FOR_C)
t <- (!is.na(lsk1$A_HORIZON))
lsk1[t,]$bulk <- top[lsk1[t,]$GEO_FOR_C]
lsk1[!t,]$bulk <- sub[lsk1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
lsk2$bulk <- NA
lsk2 <- subset(lsk2,!is.na(lsk2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.4

s2 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.8

s4 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.65

s5 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50) )
lsk2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.5
```

```
s2 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.65

s4 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.6

s5 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50))
lsk2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
lsk <- merge(lsk1,lsk2,all=TRUE)
rm(lsk1,lsk2)

# Exclude samples collected from more than 1 soil horizon - LSK

isCrossing <- (!is.na(lsk$MON_VNR)  &
    ((lsk$MON_DIEPB < lsk$HOR_DIEPB) & (lsk$MON_DIEPO > lsk$HOR_DIEPB)) |
    ((lsk$MON_DIEPB < lsk$HOR_DIEPO) & (lsk$MON_DIEPO > lsk$HOR_DIEPO))
)

lsk <- lsk[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - LSK

# Subset removing NA records in LUTUM column
lsk <- subset(x = lsk,subset = !is.na(lsk$LUTUM))

lsk$s_thick <- lsk$MON_DIEPO - lsk$MON_DIEPB
lsk$s_thick.clay <- lsk$s_thick * lsk$LUTUM

lsk$HOR_ID <- as.character(lsk$LAAG_ID)

sum1 <- tapply(X = lsk$s_thick.clay, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_clay <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Weighted averaging of bulk density

lsk$s_thick.bulk <- lsk$s_thick * lsk$bulk

sum1 <- tapply(X = lsk$s_thick.bulk, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_bulk <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
lsk <- subset(x = lsk,subset = duplicated(lsk$LAAG_ID)==FALSE)

# Horizons selection - LSK

lsk <- subset(x = lsk,subset =
    (HOR_DIEPB> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB< d1 & HOR_DIEPO> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB> d1 & HOR_DIEPB< d2 & HOR_DIEPO> d2) |
    (HOR_DIEPB<=d1 & HOR_DIEPO>=d2) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
lsk$h_thick <- 0
```

```
s1 <- ((lsk$HOR_DIEPB > d1) & (lsk$HOR_DIEPO < d2))
lsk[s1,]$h_thick <- lsk[s1,]$HOR_DIEPO - lsk[s1,]$HOR_DIEPB

s2 <- ((lsk$HOR_DIEPB < d1) & (lsk$HOR_DIEPO > d1) & (lsk$HOR_DIEPO < d2))
lsk[s2,]$h_thick <- lsk[s2,]$HOR_DIEPO - d1

s3 <- ((lsk$HOR_DIEPB > d1) & (lsk$HOR_DIEPB < d2) & (lsk$HOR_DIEPO > d2))
lsk[s3,]$h_thick <- d2 - lsk[s3,]$HOR_DIEPB

s4 <- ((lsk$HOR_DIEPB <= d1) & (lsk$HOR_DIEPO >= d2))
lsk[s4,]$h_thick <- d2 - d1

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
lsk$ratio <- (lsk$h_thick/(d2-d1))

# The sum of ratios
lsk$PROF_ID <- as.character(lsk$PROFIEL_ID)
sum_ratio <- tapply(X = lsk$ratio, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum_ratio <- sum_ratio[lsk$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
lsk <- subset(x = lsk,subset = sum_ratio > 0.999)

# Profile weighted averaging - LSK

lsk$SoilMass <- lsk$h_thick * lsk$l_bulk
lsk$SoilMass.l_clay <- lsk$SoilMass * lsk$l_clay

sum1 <- tapply(X = lsk$SoilMass.l_clay, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$PROF_ID]

sum2 <- tapply(X = lsk$SoilMass, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$PROF_ID]

lsk$wa_clay <- lsk$sum1/ lsk$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
lsk <- subset(x = lsk, subset = duplicated(lsk$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = lsk$X, Y = lsk$Y)
dup2 <- duplicated(dup)

lsk$dupl_loc <- dup2
rm(dup,dup2)

lsk <- subset(x = lsk, subset = (dupl_loc==FALSE))

# Available clay content data for the depth d1 - d2
lsk <- data.frame(X = lsk$X, Y = lsk$Y, clay = lsk$wa_clay)

# Pawn soil types - LSK

# overlay sampling locaations with pawn-map
coordinates(lsk) <- ~ X+Y
o <- overlay(map,lsk)

lsk$pawn <- o$PAWN_CODE
rm(o)

# Merge tables

clay <- merge(pfb,lsk,all=TRUE)
clay <- merge(clay,augering,all=TRUE)
rm(pfb,lsk,augering)

# exlude locations which fall into the water,build up areas, and NA
clay <- subset(x = clay, subset = (pawn < 22) & (!is.na(pawn)))
```

```
# 26 Save Box-plots
png("box_plot.png",width=800,height=600,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
boxplot(split(clay$clay,clay$pawn),xlab="Pawn soil type",main="Clay content",
     cex.main=2)
dev.off()

# 27 Histograms

# histogram of original dataset
png("hist.png",width=400,height=400,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(clay$clay,main="",xlab="Clay content",col="lightblue")
dev.off()
```

B.2 Variography

```
# Variography

# Compute standardized residuals

mean <- tapply(clay$clay,clay$pawn,mean)
clay$mean <- mean[clay$pawn]

sd <- tapply(clay$clay,clay$pawn,sd)
clay$sd <- sd[clay$pawn]

clay$residua <- (clay$clay - clay$mean) / clay$sd

# Experimental variogram
coordinates(clay) <- ~X+Y

gCLAY <- gstat(id = c("CLAY"), formula = residua ~ 1, data = clay)
gCLAY.vg <- variogram(gCLAY,boundaries=c(500,700,1000,1500,2500,10000,20000,
     30000,40000,50000,60000,70000,80000,90000,100000,110000))


# Fit variogram model
vgm <- fit.variogram(gCLAY.vg, vgm(psill= 1,model="Exp",range=50000,
add.to=vgm(psill= 1,model="Sph",range=1000,nugget=1)),fit.sill=TRUE,fit.ranges=TRUE)

# Save residual variogram figure
png("variogram.png",width=600,height=600,units="px",pointsize=12,bg="white",res=NA)
fontsize <- trellis.par.get("fontsize")
fontsize$text <- 20
fontsize$points <- 20
trellis.par.set("fontsize",fontsize)
plot(gCLAY.vg,vgm,main="Clay content - residual variogram",pch=20)
dev.off()
```

B.3 Define interpolation grid

```
# Grid
# Define interpolation grid - chose one option only

# 1) rectangular grid

######### Set parameters ##########

# Output map extent
x.min <- 10000
x.max <- 280000
y.min <- 300000
y.max <- 620000

# Prediction grid cell size (output map resolution)
```

```
cs <- 1000

###################################

x <- seq(x.min,x.max,cs)
y <- seq(y.min,y.max,cs)
grid <- expand.grid(x1=x,x2=y)
coordinates(grid) <- ~x1+x2
o <- overlay(map,grid)
grid <- data.frame(grid,pawn=o$PAWN_CODE)
grid <- subset(grid,(pawn < 22 & !is.na(pawn)))
coordinates(grid) <- ~x1+x2
rm(o)


# 2) sampling within polygon(s)

######### Set parameters ##########

# Prediction grid cell size (output map resolution)
cs <- 1000

###################################

# Read ESRI shapefile

# the whole NL boundary (en example)
nl_poly <- readShapePoly("NL.shp")

grid <- spsample(nl_poly,cellsize=cs,type="regular")
o <- overlay(map,grid)
grid <- data.frame(grid,pawn=o$PAWN_CODE)
grid <- subset(grid,(pawn < 22 & !is.na(pawn)))
coordinates(grid)<- ~x1+x2
rm(o)
```

B.4 Stochastic simulation

```
# Stochastic simulation

######## Set parameters ################

# Number of simulations
ns <- 1

# Point or block kriging
# if "block <- 0" ... point kriging
# if "block <- 1" ... block kriging
block <- 0

# Block size (for block kriging only)
bs <- c(10,10)

########################################

# Define gstat object to simulate residuals

gstatCLAY <- gstat(id=c("CLAY"),formula=residua~1,data=clay,model=vgm,beta=0,
     nmax=30,maxdist=20000)

# Ready to predict clay content residuals using Gaussian simulation based on
# simple kriging
if(block==0)    blup <- predict.gstat(gstatCLAY, newdata = grid, BLUE = FALSE,
                nsim = ns) else
                blup <- predict.gstat(gstatCLAY, newdata = grid, BLUE = FALSE,
                nsim = ns,block=bs)

# Simulations computation
sim <- data.frame(blup, pawn = grid$pawn)

sim$mean <- mean[sim$pawn]
```

```
sim$sd <- sd[sim$pawn]

sim_ns <- data.frame(x1 = sim$x1, x2 = sim$x2)

sim_ns[,3:(ns+2)] <- sim[,3:(ns+2)] * sim$sd + sim$mean

# Save figures
coordinates(sim_ns) <- ~x1+x2
gridded(sim_ns) <- TRUE

nl_poly <- readShapePoly("NL.shp")
text <- list("sp.text",c(60000,550000),paste(d1,"-",d2,"cm"),cex=1.5)
nl <- list("sp.polygons", nl_poly)
rgb.palette <- colorRampPalette(c("bisque","coral","coral2","darkred"),
     space = "rgb")

for (i in 1:ns){
png(filename = paste("sim",i,".png",sep=""), width=600, height=600, units ="px",
     pointsize=12, bg="white", res=NA)
print(spplot(sim_ns[,i], main = list(label=paste("Simulation",i),cex=1.5),
col.regions=rgb.palette(17),
xlab=list(paste(d1,"-",d2,"cm"),cex=1.5),
at=c(seq(0,40,2.5),70),
colorkey=list(space="right",width=2,at=1:18,labels=list(cex=2,at=1:18,
labels=c("0","","","","10","","","","20","","","","30","","","","40 >",""))),
sp.layout=list(nl,text)))
dev.off()
}
```

B.5 Prediction

```
# Prediction

############ Set parameters #############

# Point or block kriging
# "block <- 0" ... point kriging
# "block <- 1" ... block kriging
block <- 0

# Block size (for block kriging only)
bs <- c(10,10)

########################################

# Define the gstat object to predict residuals

gstatCLAY <- gstat(id=c("CLAY"),formula=residua~1,data=clay,model=vgm,beta=0,
     nmax=30,maxdist=20000)

# Ready to compute clay content residual prediction using simple kriging
if(block==0)    blup2 <- predict.gstat(gstatCLAY, newdata = grid, BLUE = FALSE) else
            blup2 <- predict.gstat(gstatCLAY, newdata = grid, BLUE = FALSE,block=bs)

# Compute prediction
pred <- data.frame(blup2, pawn = grid$pawn)

pred$mean <- mean[pred$pawn]

pred$sd <- sd[pred$pawn]


pred$pred <- pred$CLAY.pred * pred$sd + pred$mean

coordinates(pred) <- ~x1+x2
gridded(pred) <- TRUE

# Standard deviation map
pred$pred_sd <- sqrt(pred$CLAY.var)
pred$map_sd <- pred$sd * pred$pred_sd
```

```
# 95 lower limit map
pred$lower95 <- pred$pred - 1.96 * pred$map_sd

# 95 upper limit map
pred$upper95 <- pred$pred + 1.96 * pred$map_sd

# Save figures
nl_poly <- readShapePoly("NL.shp")
text <- list("sp.text",c(60000,550000),paste(d1,"-",d2,"cm"),cex=1.5)
nl <- list("sp.polygons", nl_poly)
rgb.palette <- colorRampPalette(c("bisque","coral","coral2","darkred"),
     space = "rgb")

# Prediction map
png("pred.png",width=600,height=600,units="px",pointsize=12,bg="white",res=NA)
spplot(pred["pred"],main=list(label="Clay content - prediction",cex=1.5),
xlab=list(paste(d1,"-",d2,"cm"),cex=1.5),
col.regions=rgb.palette(17),
at=c(seq(0,40,2.5),70),
colorkey=list(space="right",width=2,at=1:18,labels=list(cex=2,at=1:18,
labels=c("0","","","","10","","","","20","","","","30","","","","40 >",""))),
sp.layout=list(nl,text))
dev.off()

# Standard deviation map
png("stdev.png",width=600,height=600,units="px",pointsize=12,bg="white",res=NA)
spplot(pred["map_sd"],main=list(label="Clay content - standard deviation",cex=1.5),
xlab=list(paste(d1,"-",d2,"cm"),cex=1.5),
col.regions=rgb.palette(17),
colorkey=list(space="right",width=2),
sp.layout=list(nl,text))
dev.off()

# 95% lower limit map
png("lower95.png",width=600,height=600,units="px",pointsize=12,bg="white",res=NA)
spplot(pred["lower95"],main=list(label="Clay content - 95% lower limit",cex=1.5),
xlab=list(paste(d1,"-",d2,"cm"),cex=1.5),
col.regions=rgb.palette(17),
at=c(seq(0,40,2.5),70),
colorkey=list(space="right",width=2,at=1:18,labels=list(cex=2,at=1:18,
labels=c("0","","","","10","","","","20","","","","30","","","","40 >",""))),
sp.layout=list(nl,text))
dev.off()

# 95% upper limit map
png("upper95.png",width=600,height=600,units="px",pointsize=12,bg="white",res=NA)
spplot(pred["upper95"],main=list(label="Clay content - 95% upper limit",cex=1.5),
xlab=list(paste(d1,"-",d2,"cm"),cex=1.5),
col.regions=rgb.palette(17),
at=c(seq(0,40,2.5),70),
colorkey=list(space="right",width=2,at=1:18,labels=list(cex=2,at=1:18,
labels=c("0","","","","10","","","","20","","","","30","","","","40 >",""))),
sp.layout=list(nl,text))
dev.off()
```

Appendix C

R code Clay Content, multiple layer analysis

C.1 First layer data extraction and preparation

```
# First layer data extraction and preparation

############ Set parameters ###############

# Load libraries

library(RODBC)
library(foreign)
library(sp)
library(maptools)
library(gstat)
library(lattice)

# The thickness of the first soil layer (cm)
d1 <- 10
d2 <- 15

# Data filtering (spatial extent of input data)
X_MIN <- 0
X_MAX <- 300000
Y_MIN <- 300000
Y_MAX <- 620000

# Time filtering: yyyy-mm-dd

from <- as.POSIXct("1000-01-01")
to   <- as.POSIXct("3000-01-01")


##########################################

# View extraction - V_AUGERING_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_augering_deltabis view
augering <- sqlFetch(channel = channel, sqtable = "V_AUGERING_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - Augering
```

```
augering$MON_DATUM <- paste(augering$MON_DATUM,"-01-01",sep="")
augering$MON_DATUM <- as.POSIXct(augering$MON_DATUM)

augering <- subset(augering,(!is.na(augering$MON_DATUM) & augering$MON_DATUM >
     from & augering$MON_DATUM < to))

# Spatial extent filtering - Augering
augering <- subset(augering,(!is.na(augering$X) & augering$X > X_MIN & augering$X < X_MAX))
augering <- subset(augering,(!is.na(augering$Y) & augering$Y > Y_MIN & augering$Y < Y_MAX))

# Horizons selection - Augering

# Exclude records with missing clay content values
augering <- subset(x = augering, subset = !is.na(LUTUM))

augering <- subset(x = augering,subset =
    (HOR_DIEPB > d1 & HOR_DIEPO < d2) |
    (HOR_DIEPB < d1 & HOR_DIEPO > d1 & HOR_DIEPO < d2) |
    (HOR_DIEPB > d1 & HOR_DIEPB < d2 & HOR_DIEPO > d2) |
    (HOR_DIEPB <= d1 & HOR_DIEPO >= d2) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
augering$h_thick <- 0

s1 <- ((augering$HOR_DIEPB > d1) & (augering$HOR_DIEPO < d2))
augering[s1,]$h_thick <- augering[s1,]$HOR_DIEPO - augering[s1,]$HOR_DIEPB

s2 <- ((augering$HOR_DIEPB < d1) & (augering$HOR_DIEPO > d1) & (augering$HOR_DIEPO < d2))
augering[s2,]$h_thick <- augering[s2,]$HOR_DIEPO - d1

s3 <- ((augering$HOR_DIEPB > d1) & (augering$HOR_DIEPB < d2) & (augering$HOR_DIEPO > d2))
augering[s3,]$h_thick <- d2 - augering[s3,]$HOR_DIEPB

s4 <- ((augering$HOR_DIEPB <= d1) & (augering$HOR_DIEPO >= d2))
augering[s4,]$h_thick <- d2 - d1

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
augering$ratio <- (augering$h_thick/(d2-d1))

# The sum of ratios
augering$PROF_ID <- as.character(augering$PROFIEL_ID)
sum_ratio <- tapply(X = augering$ratio, INDEX = augering$PROF_ID, FUN = sum)
augering$sum_ratio <- sum_ratio[augering$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
augering <- subset(x = augering,subset = sum_ratio > 0.999)

# Bulk density - Augering

# Split the table into two dataset based on geocode value
# to apply two different aproaches for bulk density computation

augering1 <- subset(x = augering, subset = ((GEO_FOR_C > 0) & (augering$GEO_FOR_C <= 690)))

augering2 <- subset(x = augering, subset = ((GEO_FOR_C > 690) |
     (augering$GEO_FOR_C == 0) | (is.na(augering$GEO_FOR_C))))
rm(augering)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")
top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
augering1$bulk <- NA
augering1$GEO_FOR_C <- as.character(augering1$GEO_FOR_C)
t <- (!is.na(augering1$A_HORIZON))
augering1[t,]$bulk <- top[augering1[t,]$GEO_FOR_C]
augering1[!t,]$bulk <- sub[augering1[!t,]$GEO_FOR_C]
```

```
rm(top,sub,t,BD)

# The bulk density for the second subset
augering2$bulk <- NA
augering2 <- subset(augering2,!is.na(augering2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.4

s2 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.8

s4 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.65

s5 <- ((!is.na(augering2$A_HORIZON)) &
     (augering2$ORGSTOF >= 50) )
augering2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.5

s2 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.65

s4 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.6

s5 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 50))
augering2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
augering <- merge(augering1,augering2,all=TRUE)
rm(augering1,augering2)

# Profile weighted averaging - Augering

augering$SoilMass <- augering$h_thick * augering$bulk
augering$SoilMass.LUTUM <- augering$SoilMass * augering$LUTUM

sum1 <- tapply(X = augering$SoilMass.LUTUM, INDEX = augering$PROF_ID, FUN = sum)
augering$sum1 <- sum1[augering$PROF_ID]

sum2 <- tapply(X = augering$SoilMass, INDEX = augering$PROF_ID, FUN = sum)
augering$sum2 <- sum2[augering$PROF_ID]

augering$wa_clay <- augering$sum1/ augering$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
augering <- subset(x = augering, subset = duplicated(augering$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = augering$X, Y = augering$Y)
dup2 <- duplicated(dup)

augering$dupl_loc <- dup2
rm(dup,dup2)
```

```
augering <- subset(x = augering, subset = (dupl_loc==FALSE))

# All available clay content data for the depth d1 - d2
augering <- data.frame(X = augering$X, Y = augering$Y, clay = augering$wa_clay)

# Add pawn soil types - Augering

# read the pawn_map08.shp file
map <- readShapePoly("pawn-map08.shp",IDvar="PAWN_ID")

# overlay sampling locaations with pawn-map
coordinates(augering) <- ~X+Y
o <- overlay(map,augering)

augering$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_PFB_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_pfb_deltabis view
pfb <- sqlFetch(channel = channel, sqtable = "V_PFB_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$MON_DATUM) & pfb$MON_DATUM > from &
     pfb$MON_DATUM < to))

# Spatial extent filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$X) & pfb$X > X_MIN & pfb$X < X_MAX))
pfb <- subset(pfb,(!is.na(pfb$Y) & pfb$Y > Y_MIN & pfb$Y < Y_MAX))

# Bulk density - PFB

# Split the table into two dataset based on geocode value
# to apply two different aproaches for bulk density computation

pfb1 <- subset(pfb,((GEO_FOR_C > 0) & (pfb$GEO_FOR_C <= 690)))

pfb2 <- subset(pfb,((GEO_FOR_C > 690) | (pfb$GEO_FOR_C == 0) |
     (is.na(pfb$GEO_FOR_C))))
rm(pfb)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")
top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
pfb1$bulk <- NA
pfb1$GEO_FOR_C <- as.character(pfb1$GEO_FOR_C)
t <- (!is.na(pfb1$A_HORIZON))
pfb1[t,]$bulk <- top[pfb1[t,]$GEO_FOR_C]
pfb1[!t,]$bulk <- sub[pfb1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
pfb2$bulk <- NA
pfb2 <- subset(pfb2,!is.na(pfb2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.4

s2 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
```

```
pfb2[s3,]$bulk <- 0.8

s4 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.65

s5 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50) )
pfb2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.5

s2 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.65

s4 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.6

s5 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50))
pfb2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
pfb <- merge(pfb1,pfb2,all=TRUE)
rm(pfb1,pfb2)

# Exclude samples collected from more than 1 soil horizon - PFB

isCrossing <- (!is.na(pfb$MON_VNR)  &
    ((pfb$MON_DIEPB < pfb$HOR_DIEPB) & (pfb$MON_DIEPO > pfb$HOR_DIEPB)) |
    ((pfb$MON_DIEPB < pfb$HOR_DIEPO) & (pfb$MON_DIEPO > pfb$HOR_DIEPO))
)

pfb <- pfb[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - PFB

# Compute weighted average of organic matter and bulk density for each soil horizon
# (important where there are more samples which belong to the same horizon)

# Subset removing NA records in LUTUM column
pfb <- subset(x = pfb,subset = !is.na(pfb$LUTUM))

# Weighted averiging of organic carbon content

pfb$s_thick <- pfb$MON_DIEPO - pfb$MON_DIEPB
pfb$s_thick.clay <- pfb$s_thick * pfb$LUTUM

pfb$HOR_ID <- as.character(pfb$LAAG_ID)

sum1 <- tapply(X = pfb$s_thick.clay, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_clay <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Weighted averaging of bulk density

pfb$s_thick.bulk <- pfb$s_thick * pfb$bulk

sum1 <- tapply(X = pfb$s_thick.bulk, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_bulk <- (pfb$sum1/pfb$sum2)
```

```
rm(sum1,sum2)

# Remove duplicated layers - no more needed
pfb <- subset(x = pfb,subset = duplicated(pfb$LAAG_ID)==FALSE)

# Horizons selection - PFB

pfb <- subset(x = pfb,subset =
    (HOR_DIEPB> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB< d1 & HOR_DIEPO> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB> d1 & HOR_DIEPB< d2 & HOR_DIEPO> d2) |
    (HOR_DIEPB<=d1 & HOR_DIEPO>=d2) )

# Computing the thickness of a part of horizon which contribute to the
# required soil layer
pfb$h_thick <- 0

s1 <- ((pfb$HOR_DIEPB > d1) & (pfb$HOR_DIEPO < d2))
pfb[s1,]$h_thick <- pfb[s1,]$HOR_DIEPO - pfb[s1,]$HOR_DIEPB

s2 <- ((pfb$HOR_DIEPB < d1) & (pfb$HOR_DIEPO > d1) & (pfb$HOR_DIEPO < d2))
pfb[s2,]$h_thick <- pfb[s2,]$HOR_DIEPO - d1

s3 <- ((pfb$HOR_DIEPB > d1) & (pfb$HOR_DIEPB < d2) & (pfb$HOR_DIEPO > d2))
pfb[s3,]$h_thick <- d2 - pfb[s3,]$HOR_DIEPB

s4 <- ((pfb$HOR_DIEPB <= d1) & (pfb$HOR_DIEPO >= d2))
pfb[s4,]$h_thick <- d2 - d1

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
pfb$ratio <- (pfb$h_thick/(d2-d1))

# The sum of ratios
pfb$PROF_ID <- as.character(pfb$PROFIEL_ID)
sum_ratio <- tapply(X = pfb$ratio, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum_ratio <- sum_ratio[pfb$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
pfb <- subset(x = pfb,subset = sum_ratio > 0.999)

# Profile weighted averaging - PFB

pfb$SoilMass <- pfb$h_thick * pfb$l_bulk
pfb$SoilMass.l_clay <- pfb$SoilMass * pfb$l_clay

sum1 <- tapply(X = pfb$SoilMass.l_clay, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$PROF_ID]

sum2 <- tapply(X = pfb$SoilMass, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$PROF_ID]

pfb$wa_clay <- pfb$sum1/ pfb$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
pfb <- subset(x = pfb, subset = duplicated(pfb$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = pfb$X, Y = pfb$Y)
dup2 <- duplicated(dup)

pfb$dupl_loc <- dup2
rm(dup,dup2)

pfb <- subset(x = pfb, subset = (dupl_loc==FALSE))

# All available clay content data for the depth d1 - d2
pfb <- data.frame(X = pfb$X, Y = pfb$Y, clay = pfb$wa_clay)

# Pawn soil types - PFB
```

```
# overlay sampling locaations with pawn-map
coordinates(pfb) <- ~ X+Y
o <- overlay(map,pfb)

pfb$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_LSK_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_lsk_deltabis view
lsk <- sqlFetch(channel = channel, sqtable = "V_LSK_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$MON_DATUM) & lsk$MON_DATUM > from &
     lsk$MON_DATUM < to))

# Spatial extent filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$X) & lsk$X > X_MIN & lsk$X < X_MAX))
lsk <- subset(lsk,(!is.na(lsk$Y) & lsk$Y > Y_MIN & lsk$Y < Y_MAX))

# Bulk density - LSK

# Split the table into two dataset based on geocode value
# to apply two different aproaches for bulk density computation

lsk1 <- subset(lsk,((GEO_FOR_C > 0) & (lsk$GEO_FOR_C <= 690)))

lsk2 <- subset(lsk,((GEO_FOR_C > 690) | (lsk$GEO_FOR_C == 0) |
     (is.na(lsk$GEO_FOR_C))))
rm(lsk)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")
top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
lsk1$bulk <- NA
lsk1$GEO_FOR_C <- as.character(lsk1$GEO_FOR_C)
t <- (!is.na(lsk1$A_HORIZON))
lsk1[t,]$bulk <- top[lsk1[t,]$GEO_FOR_C]
lsk1[!t,]$bulk <- sub[lsk1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
lsk2$bulk <- NA
lsk2 <- subset(lsk2,!is.na(lsk2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.4

s2 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.8

s4 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.65

s5 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50) )
lsk2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
```

```
lsk2[s1,]$bulk <- 1.5

s2 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.65

s4 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.6

s5 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50))
lsk2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
lsk <- merge(lsk1,lsk2,all=TRUE)
rm(lsk1,lsk2)

# Exclude samples collected from more than 1 soil horizon - LSK

isCrossing <- (!is.na(lsk$MON_VNR)  &
    ((lsk$MON_DIEPB < lsk$HOR_DIEPB) & (lsk$MON_DIEPO > lsk$HOR_DIEPB)) |
    ((lsk$MON_DIEPB < lsk$HOR_DIEPO) & (lsk$MON_DIEPO > lsk$HOR_DIEPO))
)

lsk <- lsk[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - LSK

# Subset removing NA records in LUTUM column
lsk <- subset(x = lsk,subset = !is.na(lsk$LUTUM))

lsk$s_thick <- lsk$MON_DIEPO - lsk$MON_DIEPB
lsk$s_thick.clay <- lsk$s_thick * lsk$LUTUM

lsk$HOR_ID <- as.character(lsk$LAAG_ID)

sum1 <- tapply(X = lsk$s_thick.clay, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_clay <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Weighted averaging of bulk density

lsk$s_thick.bulk <- lsk$s_thick * lsk$bulk

sum1 <- tapply(X = lsk$s_thick.bulk, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_bulk <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
lsk <- subset(x = lsk,subset = duplicated(lsk$LAAG_ID)==FALSE)

# Horizons selection - LSK

lsk <- subset(x = lsk,subset =
    (HOR_DIEPB> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB< d1 & HOR_DIEPO> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB> d1 & HOR_DIEPB< d2 & HOR_DIEPO> d2) |
    (HOR_DIEPB<=d1 & HOR_DIEPO>=d2) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
```

```
lsk$h_thick <- 0

s1 <- ((lsk$HOR_DIEPB > d1) & (lsk$HOR_DIEPO < d2))
lsk[s1,]$h_thick <- lsk[s1,]$HOR_DIEPO - lsk[s1,]$HOR_DIEPB

s2 <- ((lsk$HOR_DIEPB < d1) & (lsk$HOR_DIEPO > d1) & (lsk$HOR_DIEPO < d2))
lsk[s2,]$h_thick <- lsk[s2,]$HOR_DIEPO - d1

s3 <- ((lsk$HOR_DIEPB > d1) & (lsk$HOR_DIEPB < d2) & (lsk$HOR_DIEPO > d2))
lsk[s3,]$h_thick <- d2 - lsk[s3,]$HOR_DIEPB

s4 <- ((lsk$HOR_DIEPB <= d1) & (lsk$HOR_DIEPO >= d2))
lsk[s4,]$h_thick <- d2 - d1

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
lsk$ratio <- (lsk$h_thick/(d2-d1))

# The sum of ratios
lsk$PROF_ID <- as.character(lsk$PROFIEL_ID)
sum_ratio <- tapply(X = lsk$ratio, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum_ratio <- sum_ratio[lsk$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
lsk <- subset(x = lsk,subset = sum_ratio > 0.999)

# Profile weighted averaging - LSK

lsk$SoilMass <- lsk$h_thick * lsk$l_bulk
lsk$SoilMass.l_clay <- lsk$SoilMass * lsk$l_clay

sum1 <- tapply(X = lsk$SoilMass.l_clay, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$PROF_ID]

sum2 <- tapply(X = lsk$SoilMass, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$PROF_ID]

lsk$wa_clay <- lsk$sum1/ lsk$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
lsk <- subset(x = lsk, subset = duplicated(lsk$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = lsk$X, Y = lsk$Y)
dup2 <- duplicated(dup)

lsk$dupl_loc <- dup2
rm(dup,dup2)

lsk <- subset(x = lsk, subset = (dupl_loc==FALSE))

# Available clay content data for the depth d1 - d2
lsk <- data.frame(X = lsk$X, Y = lsk$Y, clay = lsk$wa_clay)

# Pawn soil types - LSK

# overlay sampling locaations with pawn-map
coordinates(lsk) <- ~ X+Y
o <- overlay(map,lsk)

lsk$pawn <- o$PAWN_CODE
rm(o)

# Merge tables

clay <- merge(pfb,lsk,all=TRUE)
clay <- merge(clay,augering,all=TRUE)
rm(pfb,lsk,augering)

# exlude locations which fall into the water,build up areas, and NA
```

```
clay <- subset(x = clay, subset = (pawn < 22) & (!is.na(pawn)))

# Rename table
clay1 <- clay
rm(clay)

# Write tableto the text file
write.table(clay1,"clay1.txt",sep="\t",row.names=FALSE)

# 26 Save Box-plots
png("box_plot.png",width=800,height=600,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
boxplot(split(clay1$clay,clay1$pawn),xlab="Pawn soil type",main="Clay content",
     cex.main=2)
dev.off()

# 27 Histograms

# histogram of original dataset
png("hist.png",width=400,height=400,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(clay1$clay,main="",xlab="Clay content",col="lightblue")
dev.off()
```

C.2 Second layer data extraction and preparation

```
# Second layer data extraction and preparation

############ Set parameters ##############

# The thickness of the second soil layer (cm)
d3 <- 30
d4 <- 45

##########################################

# View extraction - V_AUGERING_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_augering_deltabis view
augering <- sqlFetch(channel = channel, sqtable = "V_AUGERING_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - Augering
augering$MON_DATUM <- paste(augering$MON_DATUM,"-01-01",sep="")
augering$MON_DATUM <- as.POSIXct(augering$MON_DATUM)

augering <- subset(augering,(!is.na(augering$MON_DATUM) & augering$MON_DATUM > from &
     augering$MON_DATUM < to))

# Spatial extent filtering - Augering
augering <- subset(augering,(!is.na(augering$X) & augering$X > X_MIN & augering$X < X_MAX))
augering <- subset(augering,(!is.na(augering$Y) & augering$Y > Y_MIN & augering$Y < Y_MAX))

# Horizons selection - Augering

# Exclude records with missing clay content values
augering <- subset(x = augering, subset = !is.na(LUTUM))

augering <- subset(x = augering,subset =
    (HOR_DIEPB > d3 & HOR_DIEPO < d4) |
    (HOR_DIEPB < d3 & HOR_DIEPO > d3 & HOR_DIEPO < d4) |
    (HOR_DIEPB > d3 & HOR_DIEPB < d4 & HOR_DIEPO > d4) |
    (HOR_DIEPB <= d3 & HOR_DIEPO >= d4) )

# Computing the thickness of a part of horizon which contribute to the required
```

```
# soil layer
augering$h_thick <- 0

s1 <- ((augering$HOR_DIEPB > d3) & (augering$HOR_DIEPO < d4))
augering[s1,]$h_thick <- augering[s1,]$HOR_DIEPO - augering[s1,]$HOR_DIEPB

s2 <- ((augering$HOR_DIEPB < d3) & (augering$HOR_DIEPO > d3) & (augering$HOR_DIEPO < d4))
augering[s2,]$h_thick <- augering[s2,]$HOR_DIEPO - d3

s3 <- ((augering$HOR_DIEPB > d3) & (augering$HOR_DIEPB < d4) & (augering$HOR_DIEPO > d4))
augering[s3,]$h_thick <- d4 - augering[s3,]$HOR_DIEPB

s4 <- ((augering$HOR_DIEPB <= d3) & (augering$HOR_DIEPO >= d4))
augering[s4,]$h_thick <- d4 - d3

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
augering$ratio <- (augering$h_thick/(d4-d3))

# The sum of ratios
augering$PROF_ID <- as.character(augering$PROFIEL_ID)
sum_ratio <- tapply(X = augering$ratio, INDEX = augering$PROF_ID, FUN = sum)
augering$sum_ratio <- sum_ratio[augering$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
augering <- subset(x = augering,subset = sum_ratio > 0.999)

# Bulk density - Augering

# Split the table into two dataset based on geocode value
# to apply two different aproaches for bulk density computation

augering1 <- subset(x = augering, subset = ((GEO_FOR_C > 0) & (augering$GEO_FOR_C <= 690)))

augering2 <- subset(x = augering, subset = ((GEO_FOR_C > 690) | (augering$GEO_FOR_C == 0) |
     (is.na(augering$GEO_FOR_C))))
rm(augering)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")
top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
augering1$bulk <- NA
augering1$GEO_FOR_C <- as.character(augering1$GEO_FOR_C)
t <- (!is.na(augering1$A_HORIZON))
augering1[t,]$bulk <- top[augering1[t,]$GEO_FOR_C]
augering1[!t,]$bulk <- sub[augering1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
augering2$bulk <- NA
augering2 <- subset(augering2,!is.na(augering2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.4

s2 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.8

s4 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.65
```

```
s5 <- ((!is.na(augering2$A_HORIZON)) &
     (augering2$ORGSTOF >= 50) )
augering2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.5

s2 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.65

s4 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.6

s5 <- ((is.na(augering2$A_HORIZON)) &
     (augering2$ORGSTOF >= 50))
augering2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
augering <- merge(augering1,augering2,all=TRUE)
rm(augering1,augering2)

# Profile weighted averaging - Augering

augering$SoilMass <- augering$h_thick * augering$bulk
augering$SoilMass.LUTUM <- augering$SoilMass * augering$LUTUM

sum1 <- tapply(X = augering$SoilMass.LUTUM, INDEX = augering$PROF_ID, FUN = sum)
augering$sum1 <- sum1[augering$PROF_ID]

sum2 <- tapply(X = augering$SoilMass, INDEX = augering$PROF_ID, FUN = sum)
augering$sum2 <- sum2[augering$PROF_ID]

augering$wa_clay <- augering$sum1/ augering$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
augering <- subset(x = augering, subset = duplicated(augering$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = augering$X, Y = augering$Y)
dup2 <- duplicated(dup)

augering$dupl_loc <- dup2
rm(dup,dup2)

augering <- subset(x = augering, subset = (dupl_loc==FALSE))

# All available clay content data for the depth d3 - d4
augering <- data.frame(X = augering$X, Y = augering$Y, clay = augering$wa_clay)

# Add pawn soil types - Augering

# read the pawn_map08.shp file
map <- readShapePoly("pawn-map08.shp",IDvar="PAWN_ID")

# overlay sampling locaations with pawn-map
coordinates(augering) <- ~X+Y
o <- overlay(map,augering)

augering$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_PFB_DELTABIS

# connect to database
```

```
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_pfb_deltabis view
pfb <- sqlFetch(channel = channel, sqtable = "V_PFB_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$MON_DATUM) & pfb$MON_DATUM > from &
     pfb$MON_DATUM < to))

# Spatial extent filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$X) & pfb$X > X_MIN & pfb$X < X_MAX))
pfb <- subset(pfb,(!is.na(pfb$Y) & pfb$Y > Y_MIN & pfb$Y < Y_MAX))

# Bulk density - PFB

# Split the table into two dataset based on geocode value
# to apply two different aproaches for bulk density computation

pfb1 <- subset(pfb,((GEO_FOR_C > 0) & (pfb$GEO_FOR_C <= 690)))

pfb2 <- subset(pfb,((GEO_FOR_C > 690) | (pfb$GEO_FOR_C == 0) |
     (is.na(pfb$GEO_FOR_C))))
rm(pfb)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")
top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
pfb1$bulk <- NA
pfb1$GEO_FOR_C <- as.character(pfb1$GEO_FOR_C)
t <- (!is.na(pfb1$A_HORIZON))
pfb1[t,]$bulk <- top[pfb1[t,]$GEO_FOR_C]
pfb1[!t,]$bulk <- sub[pfb1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
pfb2$bulk <- NA
pfb2 <- subset(pfb2,!is.na(pfb2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.4

s2 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.8

s4 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.65

s5 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50) )
pfb2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.5

s2 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.65

s4 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.6
```

```
s5 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50))
pfb2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
pfb <- merge(pfb1,pfb2,all=TRUE)
rm(pfb1,pfb2)

# Exclude samples collected from more than 1 soil horizon - PFB

isCrossing <- (!is.na(pfb$MON_VNR)  &
    ((pfb$MON_DIEPB < pfb$HOR_DIEPB) & (pfb$MON_DIEPO > pfb$HOR_DIEPB)) |
    ((pfb$MON_DIEPB < pfb$HOR_DIEPO) & (pfb$MON_DIEPO > pfb$HOR_DIEPO))
)

pfb <- pfb[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - PFB

# Compute weighted average of organic matter and bulk density for each soil horizon
# (important where there are more samples which belong to the same horizon)

# Subset removing NA records in LUTUM column
pfb <- subset(x = pfb,subset = !is.na(pfb$LUTUM))

# Weighted averiging of organic carbon content

pfb$s_thick <- pfb$MON_DIEPO - pfb$MON_DIEPB
pfb$s_thick.clay <- pfb$s_thick * pfb$LUTUM

pfb$HOR_ID <- as.character(pfb$LAAG_ID)

sum1 <- tapply(X = pfb$s_thick.clay, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_clay <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Weighted averaging of bulk density

pfb$s_thick.bulk <- pfb$s_thick * pfb$bulk

sum1 <- tapply(X = pfb$s_thick.bulk, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_bulk <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
pfb <- subset(x = pfb,subset = duplicated(pfb$LAAG_ID)==FALSE)

# Horizons selection - PFB

pfb <- subset(x = pfb,subset =
    (HOR_DIEPB> d3 & HOR_DIEPO< d4) |
    (HOR_DIEPB< d3 & HOR_DIEPO> d3 & HOR_DIEPO< d4) |
    (HOR_DIEPB> d3 & HOR_DIEPB< d4 & HOR_DIEPO> d4) |
    (HOR_DIEPB<=d3 & HOR_DIEPO>=d4) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
pfb$h_thick <- 0

s1 <- ((pfb$HOR_DIEPB > d3) & (pfb$HOR_DIEPO < d4))
pfb[s1,]$h_thick <- pfb[s1,]$HOR_DIEPO - pfb[s1,]$HOR_DIEPB

s2 <- ((pfb$HOR_DIEPB < d3) & (pfb$HOR_DIEPO > d3) & (pfb$HOR_DIEPO < d4))
```

```
pfb[s2,]$h_thick <- pfb[s2,]$HOR_DIEPO - d3

s3 <- ((pfb$HOR_DIEPB > d3) & (pfb$HOR_DIEPB < d4) & (pfb$HOR_DIEPO > d4))
pfb[s3,]$h_thick <- d4 - pfb[s3,]$HOR_DIEPB

s4 <- ((pfb$HOR_DIEPB <= d3) & (pfb$HOR_DIEPO >= d4))
pfb[s4,]$h_thick <- d4 - d3

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
pfb$ratio <- (pfb$h_thick/(d4-d3))

# The sum of ratios
pfb$PROF_ID <- as.character(pfb$PROFIEL_ID)
sum_ratio <- tapply(X = pfb$ratio, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum_ratio <- sum_ratio[pfb$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
pfb <- subset(x = pfb,subset = sum_ratio > 0.999)

# Profile weighted averaging - PFB

pfb$SoilMass <- pfb$h_thick * pfb$l_bulk
pfb$SoilMass.l_clay <- pfb$SoilMass * pfb$l_clay

sum1 <- tapply(X = pfb$SoilMass.l_clay, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$PROF_ID]

sum2 <- tapply(X = pfb$SoilMass, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$PROF_ID]

pfb$wa_clay <- pfb$sum1/ pfb$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
pfb <- subset(x = pfb, subset = duplicated(pfb$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = pfb$X, Y = pfb$Y)
dup2 <- duplicated(dup)

pfb$dupl_loc <- dup2
rm(dup,dup2)

pfb <- subset(x = pfb, subset = (dupl_loc==FALSE))

# All available clay content data for the depth d3 - d4
pfb <- data.frame(X = pfb$X, Y = pfb$Y, clay = pfb$wa_clay)

# Pawn soil types - PFB

# overlay sampling locaations with pawn-map
coordinates(pfb) <- ~ X+Y
o <- overlay(map,pfb)

pfb$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_LSK_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_lsk_deltabis view
lsk <- sqlFetch(channel = channel, sqtable = "V_LSK_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - LSK
```

```
lsk <- subset(lsk,(!is.na(lsk$MON_DATUM) & lsk$MON_DATUM > from &
     lsk$MON_DATUM < to))

# Spatial extent filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$X) & lsk$X > X_MIN & lsk$X < X_MAX))
lsk <- subset(lsk,(!is.na(lsk$Y) & lsk$Y > Y_MIN & lsk$Y < Y_MAX))

# Bulk density - LSK

# Split the table into two dataset based on geocode value
# to apply two different aproaches for bulk density computation

lsk1 <- subset(lsk,((GEO_FOR_C > 0) & (lsk$GEO_FOR_C <= 690)))

lsk2 <- subset(lsk,((GEO_FOR_C > 690) | (lsk$GEO_FOR_C == 0) |
     (is.na(lsk$GEO_FOR_C))))
rm(lsk)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")
top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
lsk1$bulk <- NA
lsk1$GEO_FOR_C <- as.character(lsk1$GEO_FOR_C)
t <- (!is.na(lsk1$A_HORIZON))
lsk1[t,]$bulk <- top[lsk1[t,]$GEO_FOR_C]
lsk1[!t,]$bulk <- sub[lsk1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
lsk2$bulk <- NA
lsk2 <- subset(lsk2,!is.na(lsk2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.4

s2 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.8

s4 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.65

s5 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50) )
lsk2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.5

s2 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.65

s4 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.6

s5 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50))
lsk2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
lsk <- merge(lsk1,lsk2,all=TRUE)
rm(lsk1,lsk2)

# Exclude samples collected from more than 1 soil horizon - LSK
```

```
isCrossing <- (!is.na(lsk$MON_VNR)  &
    ((lsk$MON_DIEPB < lsk$HOR_DIEPB) & (lsk$MON_DIEPO > lsk$HOR_DIEPB)) |
    ((lsk$MON_DIEPB < lsk$HOR_DIEPO) & (lsk$MON_DIEPO > lsk$HOR_DIEPO))
)

lsk <- lsk[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - LSK

# Subset removing NA records in LUTUM column
lsk <- subset(x = lsk,subset = !is.na(lsk$LUTUM))

lsk$s_thick <- lsk$MON_DIEPO - lsk$MON_DIEPB
lsk$s_thick.clay <- lsk$s_thick * lsk$LUTUM

lsk$HOR_ID <- as.character(lsk$LAAG_ID)

sum1 <- tapply(X = lsk$s_thick.clay, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_clay <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Weighted averaging of bulk density

lsk$s_thick.bulk <- lsk$s_thick * lsk$bulk

sum1 <- tapply(X = lsk$s_thick.bulk, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_bulk <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
lsk <- subset(x = lsk,subset = duplicated(lsk$LAAG_ID)==FALSE)

# Horizons selection - LSK

lsk <- subset(x = lsk,subset =
    (HOR_DIEPB> d3 & HOR_DIEPO< d4) |
    (HOR_DIEPB< d3 & HOR_DIEPO> d3 & HOR_DIEPO< d4) |
    (HOR_DIEPB> d3 & HOR_DIEPB< d4 & HOR_DIEPO> d4) |
    (HOR_DIEPB<=d3 & HOR_DIEPO>=d4) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
lsk$h_thick <- 0

s1 <- ((lsk$HOR_DIEPB > d3) & (lsk$HOR_DIEPO < d4))
lsk[s1,]$h_thick <- lsk[s1,]$HOR_DIEPO - lsk[s1,]$HOR_DIEPB

s2 <- ((lsk$HOR_DIEPB < d3) & (lsk$HOR_DIEPO > d3) & (lsk$HOR_DIEPO < d4))
lsk[s2,]$h_thick <- lsk[s2,]$HOR_DIEPO - d3

s3 <- ((lsk$HOR_DIEPB > d3) & (lsk$HOR_DIEPB < d4) & (lsk$HOR_DIEPO > d4))
lsk[s3,]$h_thick <- d4 - lsk[s3,]$HOR_DIEPB

s4 <- ((lsk$HOR_DIEPB <= d3) & (lsk$HOR_DIEPO >= d4))
lsk[s4,]$h_thick <- d4 - d3

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
lsk$ratio <- (lsk$h_thick/(d4-d3))

# The sum of ratios
lsk$PROF_ID <- as.character(lsk$PROFIEL_ID)
```

```
sum_ratio <- tapply(X = lsk$ratio, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum_ratio <- sum_ratio[lsk$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
lsk <- subset(x = lsk,subset = sum_ratio > 0.999)

# Profile weighted averaging - LSK

lsk$SoilMass <- lsk$h_thick * lsk$l_bulk
lsk$SoilMass.l_clay <- lsk$SoilMass * lsk$l_clay

sum1 <- tapply(X = lsk$SoilMass.l_clay, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$PROF_ID]

sum2 <- tapply(X = lsk$SoilMass, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$PROF_ID]

lsk$wa_clay <- lsk$sum1/ lsk$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
lsk <- subset(x = lsk, subset = duplicated(lsk$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = lsk$X, Y = lsk$Y)
dup2 <- duplicated(dup)

lsk$dupl_loc <- dup2
rm(dup,dup2)

lsk <- subset(x = lsk, subset = (dupl_loc==FALSE))

# Available clay content data for the depth d3 - d4
lsk <- data.frame(X = lsk$X, Y = lsk$Y, clay = lsk$wa_clay)

# Pawn soil types - LSK

# overlay sampling locaations with pawn-map
coordinates(lsk) <- ~ X+Y
o <- overlay(map,lsk)

lsk$pawn <- o$PAWN_CODE
rm(o)

# Merge tables

clay <- merge(pfb,lsk,all=TRUE)
clay <- merge(clay,augering,all=TRUE)
rm(pfb,lsk,augering)

# exlude locations which fall into the water,build up areas, and NA
clay <- subset(x = clay, subset = (pawn < 22) & (!is.na(pawn)))

# Rename table
clay2 <- clay
rm(clay)

# Write tableto the text file
write.table(clay2,"clay2.txt",sep="\t",row.names=FALSE)

# 26 Save Box-plots
png("box_plot2.png",width=800,height=600,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
boxplot(split(clay2$clay,clay2$pawn),xlab="Pawn soil type",main="Clay content",
     cex.main=2)
dev.off()

# 27 Histograms

# histogram of original dataset
png("hist2.png",width=400,height=400,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
```

```
hist(clay2$clay,main="",xlab="Clay content",col="lightblue")
dev.off()
```

C.3 Third layer data extraction and preparation

```
# Third layer data extraction and preparation

############ Set parameters ##############

# The thickness of the second soil layer (cm)
d5 <- 50
d6 <- 60

##########################################

# View extraction - V_AUGERING_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_augering_deltabis view
augering <- sqlFetch(channel = channel, sqtable = "V_AUGERING_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - Augering
augering$MON_DATUM <- paste(augering$MON_DATUM,"-01-01",sep="")
augering$MON_DATUM <- as.POSIXct(augering$MON_DATUM)

augering <- subset(augering,(!is.na(augering$MON_DATUM) & augering$MON_DATUM >
     from & augering$MON_DATUM < to))

# Spatial extent filtering - Augering
augering <- subset(augering,(!is.na(augering$X) & augering$X > X_MIN & augering$X < X_MAX))
augering <- subset(augering,(!is.na(augering$Y) & augering$Y > Y_MIN & augering$Y < Y_MAX))

# Horizons selection - Augering

# Exclude records with missing clay content values
augering <- subset(x = augering, subset = !is.na(LUTUM))

augering <- subset(x = augering,subset =
    (HOR_DIEPB > d5 & HOR_DIEPO < d6) |
    (HOR_DIEPB < d5 & HOR_DIEPO > d5 & HOR_DIEPO < d6) |
    (HOR_DIEPB > d5 & HOR_DIEPB < d6 & HOR_DIEPO > d6) |
    (HOR_DIEPB <= d5 & HOR_DIEPO >= d6) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
augering$h_thick <- 0

s1 <- ((augering$HOR_DIEPB > d5) & (augering$HOR_DIEPO < d6))
augering[s1,]$h_thick <- augering[s1,]$HOR_DIEPO - augering[s1,]$HOR_DIEPB

s2 <- ((augering$HOR_DIEPB < d5) & (augering$HOR_DIEPO > d5) & (augering$HOR_DIEPO < d6))
augering[s2,]$h_thick <- augering[s2,]$HOR_DIEPO - d5

s3 <- ((augering$HOR_DIEPB > d5) & (augering$HOR_DIEPB < d6) & (augering$HOR_DIEPO > d6))
augering[s3,]$h_thick <- d6 - augering[s3,]$HOR_DIEPB

s4 <- ((augering$HOR_DIEPB <= d5) & (augering$HOR_DIEPO >= d6))
augering[s4,]$h_thick <- d6 - d5

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
augering$ratio <- (augering$h_thick/(d6-d5))

# The sum of ratios
```

```
augering$PROF_ID <- as.character(augering$PROFIEL_ID)
sum_ratio <- tapply(X = augering$ratio, INDEX = augering$PROF_ID, FUN = sum)
augering$sum_ratio <- sum_ratio[augering$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
augering <- subset(x = augering,subset = sum_ratio > 0.999)

# Bulk density - Augering

# Split the table into two dataset based on geocode value
# to apply two different aproaches for bulk density computation

augering1 <- subset(x = augering, subset = ((GEO_FOR_C > 0) & (augering$GEO_FOR_C <= 690)))

augering2 <- subset(x = augering, subset = ((GEO_FOR_C > 690) | (augering$GEO_FOR_C == 0) |
     (is.na(augering$GEO_FOR_C))))
rm(augering)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")
top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
augering1$bulk <- NA
augering1$GEO_FOR_C <- as.character(augering1$GEO_FOR_C)
t <- (!is.na(augering1$A_HORIZON))
augering1[t,]$bulk <- top[augering1[t,]$GEO_FOR_C]
augering1[!t,]$bulk <- sub[augering1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
augering2$bulk <- NA
augering2 <- subset(augering2,!is.na(augering2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.4

s2 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.8

s4 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.65

s5 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 50) )
augering2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.5

s2 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.65

s4 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.6

s5 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 50))
augering2[s5,]$bulk <- 0.35
```

```
rm(s1,s2,s3,s4,s5)

# Merge tables
augering <- merge(augering1,augering2,all=TRUE)
rm(augering1,augering2)

# Profile weighted averaging - Augering

augering$SoilMass <- augering$h_thick * augering$bulk
augering$SoilMass.LUTUM <- augering$SoilMass * augering$LUTUM

sum1 <- tapply(X = augering$SoilMass.LUTUM, INDEX = augering$PROF_ID, FUN = sum)
augering$sum1 <- sum1[augering$PROF_ID]

sum2 <- tapply(X = augering$SoilMass, INDEX = augering$PROF_ID, FUN = sum)
augering$sum2 <- sum2[augering$PROF_ID]

augering$wa_clay <- augering$sum1/ augering$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
augering <- subset(x = augering, subset = duplicated(augering$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = augering$X, Y = augering$Y)
dup2 <- duplicated(dup)

augering$dupl_loc <- dup2
rm(dup,dup2)

augering <- subset(x = augering, subset = (dupl_loc==FALSE))

# All available clay content data for the depth d5 - d6
augering <- data.frame(X = augering$X, Y = augering$Y, clay = augering$wa_clay)

# Add pawn soil types - Augering

# read the pawn_map08.shp file
map <- readShapePoly("pawn-map08.shp",IDvar="PAWN_ID")

# overlay sampling locaations with pawn-map
coordinates(augering) <- ~X+Y
o <- overlay(map,augering)

augering$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_PFB_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_pfb_deltabis view
pfb <- sqlFetch(channel = channel, sqtable = "V_PFB_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$MON_DATUM) & pfb$MON_DATUM > from &
     pfb$MON_DATUM < to))

# Spatial extent filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$X) & pfb$X > X_MIN & pfb$X < X_MAX))
pfb <- subset(pfb,(!is.na(pfb$Y) & pfb$Y > Y_MIN & pfb$Y < Y_MAX))

# Bulk density - PFB

# Split the table into two dataset based on geocode value
# to apply two different aproaches for bulk density computation

pfb1 <- subset(pfb,((GEO_FOR_C > 0) & (pfb$GEO_FOR_C <= 690)))
```

```
pfb2 <- subset(pfb,((GEO_FOR_C > 690) | (pfb$GEO_FOR_C == 0) |
     (is.na(pfb$GEO_FOR_C))))
rm(pfb)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")
top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
pfb1$bulk <- NA
pfb1$GEO_FOR_C <- as.character(pfb1$GEO_FOR_C)
t <- (!is.na(pfb1$A_HORIZON))
pfb1[t,]$bulk <- top[pfb1[t,]$GEO_FOR_C]
pfb1[!t,]$bulk <- sub[pfb1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
pfb2$bulk <- NA
pfb2 <- subset(pfb2,!is.na(pfb2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.4

s2 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.8

s4 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.65

s5 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50) )
pfb2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.5

s2 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.65

s4 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.6

s5 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50))
pfb2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
pfb <- merge(pfb1,pfb2,all=TRUE)
rm(pfb1,pfb2)

# Exclude samples collected from more than 1 soil horizon - PFB

isCrossing <- (!is.na(pfb$MON_VNR)  &
    ((pfb$MON_DIEPB < pfb$HOR_DIEPB) & (pfb$MON_DIEPO > pfb$HOR_DIEPB)) |
    ((pfb$MON_DIEPB < pfb$HOR_DIEPO) & (pfb$MON_DIEPO > pfb$HOR_DIEPO))
)

pfb <- pfb[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - PFB

# Compute weighted average of organic matter and bulk density for each soil horizon
# (important where there are more samples which belong to the same horizon)
```

```
# Subset removing NA records in LUTUM column
pfb <- subset(x = pfb,subset = !is.na(pfb$LUTUM))

# Weighted averiging of organic carbon content

pfb$s_thick <- pfb$MON_DIEPO - pfb$MON_DIEPB
pfb$s_thick.clay <- pfb$s_thick * pfb$LUTUM

pfb$HOR_ID <- as.character(pfb$LAAG_ID)

sum1 <- tapply(X = pfb$s_thick.clay, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_clay <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Weighted averaging of bulk density

pfb$s_thick.bulk <- pfb$s_thick * pfb$bulk

sum1 <- tapply(X = pfb$s_thick.bulk, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_bulk <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
pfb <- subset(x = pfb,subset = duplicated(pfb$LAAG_ID)==FALSE)

# Horizons selection - PFB

pfb <- subset(x = pfb,subset =
    (HOR_DIEPB> d5 & HOR_DIEPO< d6) |
    (HOR_DIEPB< d5 & HOR_DIEPO> d5 & HOR_DIEPO< d6) |
    (HOR_DIEPB> d5 & HOR_DIEPB< d6 & HOR_DIEPO> d6) |
    (HOR_DIEPB<=d5 & HOR_DIEPO>=d6) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
pfb$h_thick <- 0

s1 <- ((pfb$HOR_DIEPB > d5) & (pfb$HOR_DIEPO < d6))
pfb[s1,]$h_thick <- pfb[s1,]$HOR_DIEPO - pfb[s1,]$HOR_DIEPB

s2 <- ((pfb$HOR_DIEPB < d5) & (pfb$HOR_DIEPO > d5) & (pfb$HOR_DIEPO < d6))
pfb[s2,]$h_thick <- pfb[s2,]$HOR_DIEPO - d5

s3 <- ((pfb$HOR_DIEPB > d5) & (pfb$HOR_DIEPB < d6) & (pfb$HOR_DIEPO > d6))
pfb[s3,]$h_thick <- d6 - pfb[s3,]$HOR_DIEPB

s4 <- ((pfb$HOR_DIEPB <= d5) & (pfb$HOR_DIEPO >= d6))
pfb[s4,]$h_thick <- d6 - d5

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
pfb$ratio <- (pfb$h_thick/(d6-d5))

# The sum of ratios
pfb$PROF_ID <- as.character(pfb$PROFIEL_ID)
sum_ratio <- tapply(X = pfb$ratio, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum_ratio <- sum_ratio[pfb$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
pfb <- subset(x = pfb,subset = sum_ratio > 0.999)
```

```
# Profile weighted averaging - PFB

pfb$SoilMass <- pfb$h_thick * pfb$l_bulk
pfb$SoilMass.l_clay <- pfb$SoilMass * pfb$l_clay

sum1 <- tapply(X = pfb$SoilMass.l_clay, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$PROF_ID]

sum2 <- tapply(X = pfb$SoilMass, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$PROF_ID]

pfb$wa_clay <- pfb$sum1/ pfb$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
pfb <- subset(x = pfb, subset = duplicated(pfb$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = pfb$X, Y = pfb$Y)
dup2 <- duplicated(dup)

pfb$dupl_loc <- dup2
rm(dup,dup2)

pfb <- subset(x = pfb, subset = (dupl_loc==FALSE))

# All available clay content data for the depth d5 - d6
pfb <- data.frame(X = pfb$X, Y = pfb$Y, clay = pfb$wa_clay)

# Pawn soil types - PFB

# overlay sampling locaations with pawn-map
coordinates(pfb) <- ~ X+Y
o <- overlay(map,pfb)

pfb$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_LSK_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_lsk_deltabis view
lsk <- sqlFetch(channel = channel, sqtable = "V_LSK_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$MON_DATUM) & lsk$MON_DATUM > from &
     lsk$MON_DATUM < to))

# Spatial extent filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$X) & lsk$X > X_MIN & lsk$X < X_MAX))
lsk <- subset(lsk,(!is.na(lsk$Y) & lsk$Y > Y_MIN & lsk$Y < Y_MAX))

# Bulk density - LSK

# Split the table into two dataset based on geocode value
# to apply two different aproaches for bulk density computation

lsk1 <- subset(lsk,((GEO_FOR_C > 0) & (lsk$GEO_FOR_C <= 690)))

lsk2 <- subset(lsk,((GEO_FOR_C > 690) | (lsk$GEO_FOR_C == 0) |
     (is.na(lsk$GEO_FOR_C))))
rm(lsk)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")
top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))
```

```
# The bulk density for the first subset
lsk1$bulk <- NA
lsk1$GEO_FOR_C <- as.character(lsk1$GEO_FOR_C)
t <- (!is.na(lsk1$A_HORIZON))
lsk1[t,]$bulk <- top[lsk1[t,]$GEO_FOR_C]
lsk1[!t,]$bulk <- sub[lsk1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
lsk2$bulk <- NA
lsk2 <- subset(lsk2,!is.na(lsk2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.4

s2 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.8

s4 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.65

s5 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50) )
lsk2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.5

s2 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.65

s4 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.6

s5 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50))
lsk2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
lsk <- merge(lsk1,lsk2,all=TRUE)
rm(lsk1,lsk2)

# Exclude samples collected from more than 1 soil horizon - LSK

isCrossing <- (!is.na(lsk$MON_VNR)  &
    ((lsk$MON_DIEPB < lsk$HOR_DIEPB) & (lsk$MON_DIEPO > lsk$HOR_DIEPB)) |
    ((lsk$MON_DIEPB < lsk$HOR_DIEPO) & (lsk$MON_DIEPO > lsk$HOR_DIEPO))
)

lsk <- lsk[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - LSK

# Subset removing NA records in LUTUM column
lsk <- subset(x = lsk,subset = !is.na(lsk$LUTUM))

lsk$s_thick <- lsk$MON_DIEPO - lsk$MON_DIEPB
lsk$s_thick.clay <- lsk$s_thick * lsk$LUTUM

lsk$HOR_ID <- as.character(lsk$LAAG_ID)

sum1 <- tapply(X = lsk$s_thick.clay, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
```

```
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_clay <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Weighted averaging of bulk density

lsk$s_thick.bulk <- lsk$s_thick * lsk$bulk

sum1 <- tapply(X = lsk$s_thick.bulk, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_bulk <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
lsk <- subset(x = lsk,subset = duplicated(lsk$LAAG_ID)==FALSE)

# Horizons selection - LSK

lsk <- subset(x = lsk,subset =
    (HOR_DIEPB> d5 & HOR_DIEPO< d6) |
    (HOR_DIEPB< d5 & HOR_DIEPO> d5 & HOR_DIEPO< d6) |
    (HOR_DIEPB> d5 & HOR_DIEPB< d6 & HOR_DIEPO> d6) |
    (HOR_DIEPB<=d5 & HOR_DIEPO>=d6) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
lsk$h_thick <- 0

s1 <- ((lsk$HOR_DIEPB > d5) & (lsk$HOR_DIEPO < d6))
lsk[s1,]$h_thick <- lsk[s1,]$HOR_DIEPO - lsk[s1,]$HOR_DIEPB

s2 <- ((lsk$HOR_DIEPB < d5) & (lsk$HOR_DIEPO > d5) & (lsk$HOR_DIEPO < d6))
lsk[s2,]$h_thick <- lsk[s2,]$HOR_DIEPO - d5

s3 <- ((lsk$HOR_DIEPB > d5) & (lsk$HOR_DIEPB < d6) & (lsk$HOR_DIEPO > d6))
lsk[s3,]$h_thick <- d6 - lsk[s3,]$HOR_DIEPB

s4 <- ((lsk$HOR_DIEPB <= d5) & (lsk$HOR_DIEPO >= d6))
lsk[s4,]$h_thick <- d6 - d5

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
lsk$ratio <- (lsk$h_thick/(d6-d5))

# The sum of ratios
lsk$PROF_ID <- as.character(lsk$PROFIEL_ID)
sum_ratio <- tapply(X = lsk$ratio, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum_ratio <- sum_ratio[lsk$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
lsk <- subset(x = lsk,subset = sum_ratio > 0.999)

# Profile weighted averaging - LSK

lsk$SoilMass <- lsk$h_thick * lsk$l_bulk
lsk$SoilMass.l_clay <- lsk$SoilMass * lsk$l_clay

sum1 <- tapply(X = lsk$SoilMass.l_clay, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$PROF_ID]

sum2 <- tapply(X = lsk$SoilMass, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$PROF_ID]

lsk$wa_clay <- lsk$sum1/ lsk$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
```

```
lsk <- subset(x = lsk, subset = duplicated(lsk$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = lsk$X, Y = lsk$Y)
dup2 <- duplicated(dup)

lsk$dupl_loc <- dup2
rm(dup,dup2)

lsk <- subset(x = lsk, subset = (dupl_loc==FALSE))

# Available clay content data for the depth d5 - d6
lsk <- data.frame(X = lsk$X, Y = lsk$Y, clay = lsk$wa_clay)

# Pawn soil types - LSK

# overlay sampling locaations with pawn-map
coordinates(lsk) <- ~ X+Y
o <- overlay(map,lsk)

lsk$pawn <- o$PAWN_CODE
rm(o)

# Merge tables

clay <- merge(pfb,lsk,all=TRUE)
clay <- merge(clay,augering,all=TRUE)
rm(pfb,lsk,augering)

# exlude locations which fall into the water,build up areas, and NA
clay <- subset(x = clay, subset = (pawn < 22) & (!is.na(pawn)))

# Rename table
clay3 <- clay
rm(clay)

# Write tableto the text file
write.table(clay3,"clay3.txt",sep="\t",row.names=FALSE)

# 26 Save Box-plots
png("box_plot3.png",width=800,height=600,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
boxplot(split(clay3$clay,clay3$pawn),xlab="Pawn soil type",main="Clay content",
     cex.main=2)
dev.off()

# 27 Histograms

# histogram of original dataset
png("hist3.png",width=400,height=400,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(clay3$clay,main="",xlab="Clay content",col="lightblue")
dev.off()
```

C.4 Cross-variography

```
# Cross-variography

# Requires clay1, clay2, clay3 loaded in R workspace

clay1 <- read.table("clay1.txt",sep="\t",header=TRUE,na.strings="NA")
clay2 <- read.table("clay2.txt",sep="\t",header=TRUE,na.strings="NA")
clay3 <- read.table("clay3.txt",sep="\t",header=TRUE,na.strings="NA")

# Compute standartized residuals

mean1 <- tapply(clay1$clay,clay1$pawn,mean)
clay1$mean <- mean1[clay1$pawn]

sd1 <- tapply(clay1$clay,clay1$pawn,sd)
```

```
clay1$sd <- sd1[clay1$pawn]

clay1$residua <- (clay1$clay - clay1$mean) / clay1$sd
#
mean2 <- tapply(clay2$clay,clay2$pawn,mean)
clay2$mean <- mean2[clay2$pawn]

sd2 <- tapply(clay2$clay,clay2$pawn,sd)
clay2$sd <- sd2[clay2$pawn]

clay2$residua <- (clay2$clay - clay2$mean) / clay2$sd
#
mean3 <- tapply(clay3$clay,clay3$pawn,mean)
clay3$mean <- mean3[clay3$pawn]

sd3 <- tapply(clay3$clay,clay3$pawn,sd)
clay3$sd <- sd3[clay3$pawn]

clay3$residua <- (clay3$clay - clay3$mean) / clay3$sd

# Random pick to speed up variogram estimation
clay11 <- clay1[sample(nrow(clay1),size=5000,replace=F),]
clay22 <- clay2[sample(nrow(clay2),size=5000,replace=F),]
clay33 <- clay3[sample(nrow(clay3),size=5000,replace=F),]


# Define the gstat object and LMC fitting

gCLAY <- gstat(NULL ,id="clay.1", formula=residua ~ 1, locations= ~X+Y,
     data=clay11)
gCLAY <- gstat(gCLAY, id="clay.2", formula=residua ~ 1, locations= ~X+Y,
     data=clay22)
gCLAY <- gstat(gCLAY, id="clay.3", formula=residua ~ 1, locations= ~X+Y,
     data=clay33)
gCLAY <- gstat(gCLAY, model=vgm(psill= 0.2,model="Sph",range=50000,nugget=0.8),
     fill.all=TRUE)

# Compute experimental variogram
v <- variogram(gCLAY)

# LMC fitting
fit <- fit.lmc(v, gCLAY)

# Plot variograms
plot(v)
plot(v,fit)

# Save cross-variograms figure
png("lmc2.png",width=600,height=600,units="px",pointsize=12,bg="white",res=NA)
fontsize <- trellis.par.get("fontsize")
fontsize$text <- 16
fontsize$points <- 16
trellis.par.set("fontsize",fontsize)
plot(v,fit,main="Clay content - LMC",pch=20)
dev.off()
```

C.5 Define interpolation grid

```
# Define interpolation grid - chose one option only

# 1) rectangular grid

######### Set parameters ##########

# Output map extent
x.min <- 10000
x.max <- 280000
y.min <- 300000
y.max <- 620000

# Prediction grid cell size (output map resolution)
```

```
cs <- 1000

###################################

x <- seq(x.min,x.max,cs)
y <- seq(y.min,y.max,cs)
grid <- expand.grid(x1=x,x2=y)
coordinates(grid) <- ~x1+x2
o <- overlay(map,grid)
grid <- data.frame(grid,pawn=o$PAWN_CODE)
grid <- subset(grid,(pawn < 22 & !is.na(pawn)))
coordinates(grid) <- ~x1+x2
rm(o)


# 2) sampling within polygon(s)

######### Set parameters ##########

# Prediction grid cell size (output map resolution)
cs <- 1000

###################################

# Read ESRI shapefile

# the whole NL boundary (en example)
nl_poly <- readShapePoly("NL.shp")

grid <- spsample(nl_poly,cellsize=cs,type="regular")
o <- overlay(map,grid)
grid <- data.frame(grid,pawn=o$PAWN_CODE)
grid <- subset(grid,(pawn < 22 & !is.na(pawn)))
coordinates(grid)<- ~x1+x2
rm(o)
```

C.6 Co-kriging prediction

```
# Co-kriging prediction

############ Set parameters ############

# Point or block kriging
# "block <- 0" ... point kriging
# "block <- 1" ... block kriging
block <- 0

# Block size (for block kriging only)
bs <- c(10,10)

########################################

# Redefine the gstat object

gCLAY <- gstat(NULL ,id="clay.1", formula=residua ~ 1, locations= ~X+Y,
     data=clay1,nmax=30,beta=0,maxdist=20000)
gCLAY <- gstat(gCLAY, id="clay.2", formula=residua ~ 1, locations= ~X+Y,
     data=clay2,nmax=30,beta=0,maxdist=20000)
gCLAY <- gstat(gCLAY, id="clay.3", formula=residua ~ 1, locations= ~X+Y,
     data=clay3,nmax=30,beta=0,maxdist=20000)
gCLAY <- gstat(gCLAY, model=vgm(psill= 0.2,model="Sph",range=50000,nugget=0.8),
     fill.all=TRUE)

# LMC fitting again
fit <- fit.lmc(v, gCLAY)

# Cokriging predictions
if(block==0)    blup2 <- predict.gstat(fit,grid) else
            blup2 <- predict.gstat(fit,grid,block=bs)

# Prediction for the first layer
```

```
clay1.pred <- data.frame(coordinates(blup2),residua=blup2[[1]],pawn=grid$pawn)
clay1.pred$mean <- mean1[clay1.pred$pawn]
clay1.pred$sd <- sd1[clay1.pred$pawn]
clay1.pred$pred <- clay1.pred$residua * clay1.pred$sd + clay1.pred$mean

coordinates(clay1.pred) <- ~x1+x2
gridded(clay1.pred) <- TRUE

# Prediction for the second layer
clay2.pred <- data.frame(coordinates(blup2),residua=blup2[[3]],pawn=grid$pawn)
clay2.pred$mean <- mean2[clay2.pred$pawn]
clay2.pred$sd <- sd2[clay2.pred$pawn]
clay2.pred$pred <- clay2.pred$residua * clay2.pred$sd + clay2.pred$mean

coordinates(clay2.pred) <- ~x1+x2
gridded(clay2.pred) <- TRUE

# Prediction for the third layer
clay3.pred <- data.frame(coordinates(blup2),residua=blup2[[5]],pawn=grid$pawn)
clay3.pred$mean <- mean3[clay3.pred$pawn]
clay3.pred$sd <- sd3[clay3.pred$pawn]
clay3.pred$pred <- clay3.pred$residua * clay3.pred$sd + clay3.pred$mean

coordinates(clay3.pred) <- ~x1+x2
gridded(clay3.pred) <- TRUE

# Save figures
nl_poly <- readShapePoly("NL.shp")
nl <- list("sp.polygons", nl_poly)
rgb.palette <- colorRampPalette(c("bisque","coral","coral2","darkred"),
     space = "rgb")

# First layer
text <- list("sp.text",c(60000,550000),paste(d1,"-",d2,"cm"),cex=1.5)

png("clay1_copred.png",width=600,height=600,units="px",pointsize=12,
     bg="white",res=NA)
spplot(clay1.pred["pred"],main=list(label="Clay content - co-prediction",cex=1.5),
xlab=list(paste(d1,"-",d2,"cm"),cex=1.5),
col.regions=rgb.palette(17),
at=c(seq(0,40,2.5),70),
colorkey=list(space="right",width=2,at=1:18,labels=list(cex=2,at=1:18,
labels=c("0","","","","10","","","","20","","","","30","","","","40 >",""))),
sp.layout=list(nl,text))
dev.off()

# Second layer
text <- list("sp.text",c(60000,550000),paste(d3,"-",d4,"cm"),cex=1.5)

png("clay2_copred.png",width=600,height=600,units="px",pointsize=12,
     bg="white",res=NA)
spplot(clay2.pred["pred"],main=list(label="Clay content - co-prediction",cex=1.5),
xlab=list(paste(d3,"-",d4,"cm"),cex=1.5),
col.regions=rgb.palette(17),
at=c(seq(0,40,2.5),70),
colorkey=list(space="right",width=2,at=1:18,labels=list(cex=2,at=1:18,
labels=c("0","","","","10","","","","20","","","","30","","","","40 >",""))),
sp.layout=list(nl,text))
dev.off()

# Third layer
text <- list("sp.text",c(60000,550000),paste(d5,"-",d6,"cm"),cex=1.5)

png("clay3_copred.png",width=600,height=600,units="px",pointsize=12,
     bg="white",res=NA)
spplot(clay3.pred["pred"],main=list(label="Clay content - co-prediction",cex=1.5),
xlab=list(paste(d5,"-",d6,"cm"),cex=1.5),
col.regions=rgb.palette(17),
at=c(seq(0,40,2.5),70),
colorkey=list(space="right",width=2,at=1:18,labels=list(cex=2,at=1:18,
labels=c("0","","","","10","","","","20","","","","30","","","","40 >",""))),
sp.layout=list(nl,text))
dev.off()
```

C.7 Stochastic co-simulation

```
# Stochastic co-simulation (based on simple cokriging)

############ Set parameters ##############

# Number of simulations
ns <- 2

# Point or block kriging
# "block <- 0" ... point kriging
# "block <- 1" ... block kriging
block <- 0

# Block size (for block kriging only)
bs <- c(10,10)

########################################

# Redefine the gstat object

gCLAY <- gstat(NULL ,id="clay.1", formula=residua ~ 1, locations= ~X+Y,
     data=clay1,nmax=30,beta=0,maxdist=20000)
gCLAY <- gstat(gCLAY, id="clay.2", formula=residua ~ 1, locations= ~X+Y,
     data=clay2,nmax=30,beta=0,maxdist=20000)
gCLAY <- gstat(gCLAY, id="clay.3", formula=residua ~ 1, locations= ~X+Y,
     data=clay3,nmax=30,beta=0,maxdist=20000)
gCLAY <- gstat(gCLAY, model=vgm(psill= 0.2,model="Sph",range=50000,nugget=0.8),
     fill.all=TRUE)

# LMC fitting again
fit <- fit.lmc(v, gCLAY)

# Cokriging simulations
if(block==0)    blup <- predict.gstat(fit,grid,nsim=ns) else
            blup <- predict.gstat(fit,grid,nsim=ns,block=bs)

# Simulation for the first layer
clay1.sim <- data.frame(blup[,1:ns],pawn=grid$pawn)
clay1.sim$mean <- mean1[clay1.sim$pawn]
clay1.sim$sd <- sd1[clay1.sim$pawn]
clay1.sim_ns <- data.frame(coordinates(blup))
clay1.sim_ns[,3:(ns+2)] <- clay1.sim[,3:(ns+2)] * clay1.sim$sd + clay1.sim$mean

coordinates(clay1.sim_ns) <- ~x1+x2
gridded(clay1.sim_ns) <- TRUE

# Simulation for the second layer
clay2.sim <- data.frame(blup[,(ns+1):(ns*2)],pawn=grid$pawn)
clay2.sim$mean <- mean2[clay2.sim$pawn]
clay2.sim$sd <- sd2[clay2.sim$pawn]
clay2.sim_ns <- data.frame(coordinates(blup))
clay2.sim_ns[,3:(ns+2)] <- clay2.sim[,3:(ns+2)] * clay2.sim$sd + clay2.sim$mean

coordinates(clay2.sim_ns) <- ~x1+x2
gridded(clay2.sim_ns) <- TRUE

# Simulation for the third layer
clay3.sim <- data.frame(blup[,(ns*2+1):(ns*3)],pawn=grid$pawn)
clay3.sim$mean <- mean3[clay3.sim$pawn]
clay3.sim$sd <- sd3[clay3.sim$pawn]
clay3.sim_ns <- data.frame(coordinates(blup))
clay3.sim_ns[,3:(ns+2)] <- clay3.sim[,3:(ns+2)] * clay3.sim$sd + clay3.sim$mean

coordinates(clay3.sim_ns) <- ~x1+x2
gridded(clay3.sim_ns) <- TRUE

# Save figures
nl_poly <- readShapePoly("NL.shp")
nl <- list("sp.polygons", nl_poly)
rgb.palette <- colorRampPalette(c("bisque","coral","coral2","darkred"),
     space = "rgb")
```

```
# First layer
text <- list("sp.text",c(60000,550000),paste(d1,"-",d2,"cm"),cex=1.5)

for (i in 1:ns){
png(filename = paste("clay1.cosim",i,".png",sep=""), width=600,
     height=600, units="px",pointsize=12, bg="white", res=NA)
print(spplot(clay1.sim_ns[,i], main = list(label=paste("Clay content -
     co-simulation",i),cex=1.5),
col.regions=rgb.palette(17),
xlab=list(paste(d1,"-",d2,"cm"),cex=1.5),
at=c(seq(0,40,2.5),70),
colorkey=list(space="right",width=2,at=1:18,labels=list(cex=2,at=1:18,
labels=c("0","","","","10","","","","20","","","","30","","","","40 >","")),
sp.layout=list(nl,text)))
dev.off()
}

# Second layer
text <- list("sp.text",c(60000,550000),paste(d3,"-",d4,"cm"),cex=1.5)

for (i in 1:ns){
png(filename = paste("clay2.cosim",i,".png",sep=""), width=600, height=600,
     units="px",pointsize=12, bg="white", res=NA)
print(spplot(clay2.sim_ns[,i], main = list(label=paste("Clay content -
     co-simulation",i),cex=1.5),
col.regions=rgb.palette(17),
xlab=list(paste(d3,"-",d4,"cm"),cex=1.5),
at=c(seq(0,40,2.5),70),
colorkey=list(space="right",width=2,at=1:18,labels=list(cex=2,at=1:18,
labels=c("0","","","","10","","","","20","","","","30","","","","40 >","")),
sp.layout=list(nl,text)))
dev.off()
}

# Third layer
text <- list("sp.text",c(60000,550000),paste(d5,"-",d6,"cm"),cex=1.5)

for (i in 1:ns){
png(filename = paste("clay3.cosim",i,".png",sep=""), width=600, height=600,
     units="px",pointsize=12, bg="white", res=NA)
print(spplot(clay3.sim_ns[,i], main = list(label=paste("Clay content -
     co-simulation",i),cex=1.5),
col.regions=rgb.palette(17),
xlab=list(paste(d5,"-",d6,"cm"),cex=1.5),
at=c(seq(0,40,2.5),70),
colorkey=list(space="right",width=2,at=1:18,labels=list(cex=2,at=1:18,
labels=c("0","","","","10","","","","20","","","","30","","","","40 >","")),
sp.layout=list(nl,text)))
dev.off()
}
```

Appendix D

R code Organic Matter Content, single layer analysis

D.1 Data extraction and preparation

```
# Data extraction and preparation

###### Set parameters #################
# Load libraries

library(RODBC)
library(foreign)
library(sp)
library(maptools)
library(gstat)
library(lattice)

setwd("O:/Radim/Orgstof/Finalorgstofcode/Singlelayeranalysis2")

# The thickness of soil layer (cm)
d1 <- 0
d2 <- 30

# Data filtering (spatial extent of input data)
X_MIN <- 0
X_MAX <- 300000
Y_MIN <- 300000
Y_MAX <- 620000

# Time filtering: yyyy-mm-dd

from <- as.POSIXct("1000-01-01")
to   <- as.POSIXct("3000-01-01")


#######################################

# View extraction - V_Augering_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_augering_deltabis view
augering <- sqlFetch(channel = channel, sqtable = "V_AUGERING_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)
```

```
# Time filtering - Augering
augering$MON_DATUM <- paste(augering$MON_DATUM,"-01-01",sep="")
augering$MON_DATUM <- as.POSIXct(augering$MON_DATUM)

augering <- subset(augering,(!is.na(augering$MON_DATUM) & augering$MON_DATUM > from &
     augering$MON_DATUM < to))

# Spatial extent filtering - Augering
augering <- subset(augering,(!is.na(augering$X) & augering$X > X_MIN & augering$X < X_MAX))
augering <- subset(augering,(!is.na(augering$Y) & augering$Y > Y_MIN & augering$Y < Y_MAX))

# Horizons selection - Augering

# Exclude records with missing values
augering <- subset(x = augering, subset = (!is.na(ORGSTOF) & ORGSTOF > 0))

# Horizons selection
augering <- subset(x = augering,subset =
    (HOR_DIEPB > d1 & HOR_DIEPO < d2) |
    (HOR_DIEPB < d1 & HOR_DIEPO > d1 & HOR_DIEPO < d2) |
    (HOR_DIEPB > d1 & HOR_DIEPB < d2 & HOR_DIEPO > d2) |
    (HOR_DIEPB <= d1 & HOR_DIEPO >= d2) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
augering$h_thick <- 0

s1 <- ((augering$HOR_DIEPB > d1) & (augering$HOR_DIEPO < d2))
augering[s1,]$h_thick <- augering[s1,]$HOR_DIEPO - augering[s1,]$HOR_DIEPB

s2 <- ((augering$HOR_DIEPB < d1) & (augering$HOR_DIEPO > d1) & (augering$HOR_DIEPO < d2))
augering[s2,]$h_thick <- augering[s2,]$HOR_DIEPO - d1

s3 <- ((augering$HOR_DIEPB > d1) & (augering$HOR_DIEPB < d2) & (augering$HOR_DIEPO > d2))
augering[s3,]$h_thick <- d2 - augering[s3,]$HOR_DIEPB

s4 <- ((augering$HOR_DIEPB <= d1) & (augering$HOR_DIEPO >= d2))
augering[s4,]$h_thick <- d2 - d1

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
augering$ratio <- (augering$h_thick/(d2-d1))

# The sum of ratios
augering$PROF_ID <- as.character(augering$PROFIEL_ID)
sum_ratio <- tapply(X = augering$ratio, INDEX = augering$PROF_ID, FUN = sum)
augering$sum_ratio <- sum_ratio[augering$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
augering <- subset(x = augering,subset = sum_ratio > 0.999)

# Bulk density - Augering

# Split the table into two dataset based on geocode value
# to apply two different aproaches

augering1 <- subset(x = augering, subset = ((GEO_FOR_C > 0) & (augering$GEO_FOR_C <=
     690)))

augering2 <- subset(x = augering, subset = ((GEO_FOR_C > 690) |
     (augering$GEO_FOR_C == 0) | (is.na(augering$GEO_FOR_C))))
rm(augering)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")

top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
augering1$bulk <- NA
augering1$GEO_FOR_C <- as.character(augering1$GEO_FOR_C)
```

```
t <- (!is.na(augering1$A_HORIZON))
augering1[t,]$bulk <- top[augering1[t,]$GEO_FOR_C]
augering1[!t,]$bulk <- sub[augering1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
augering2$bulk <- NA
augering2 <- subset(augering2,!is.na(augering2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.4

s2 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.8

s4 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.65

s5 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 50) )
augering2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.5

s2 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.65

s4 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.6

s5 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 50))
augering2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
augering <- merge(augering1,augering2,all=TRUE)
rm(augering1,augering2)

# Profile Weighted averaging - Augering

augering$SoilMass <- augering$h_thick * augering$bulk
augering$SoilMass.ORGSTOF <- augering$SoilMass * augering$ORGSTOF

sum1 <- tapply(X = augering$SoilMass.ORGSTOF, INDEX = augering$PROF_ID, FUN = sum)
augering$sum1 <- sum1[augering$PROF_ID]

sum2 <- tapply(X = augering$SoilMass, INDEX = augering$PROF_ID, FUN = sum)
augering$sum2 <- sum2[augering$PROF_ID]

augering$wa_org <- augering$sum1/ augering$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
augering <- subset(x = augering, subset = duplicated(augering$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = augering$X, Y = augering$Y)
dup2 <- duplicated(dup)
```

```
augering$dupl_loc <- dup2
rm(dup,dup2)

augering <- subset(x = augering, subset = (dupl_loc==FALSE))

# All available organic matter data for the depth d1 - d2
augering <- data.frame(X = augering$X, Y = augering$Y, org = augering$wa_org)


# Pawn soil types - Augering

# read the pawn_map08.shp file
map <- readShapePoly("pawn-map08.shp",IDvar="PAWN_ID")

# overlay sampling locaations with pawn-map
coordinates(augering) <- ~X+Y
o <- overlay(map,augering)

augering$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_PFB_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_pfb_deltabis view
pfb <- sqlFetch(channel = channel, sqtable = "V_PFB_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$MON_DATUM) & pfb$MON_DATUM > from &
     pfb$MON_DATUM < to))

# Spatial extent filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$X) & pfb$X > X_MIN & pfb$X < X_MAX))
pfb <- subset(pfb,(!is.na(pfb$Y) & pfb$Y > Y_MIN & pfb$Y < Y_MAX))

# Bulk density - PFB

# Split the table into two dataset based on geocode value
# to apply two different aproaches

pfb1 <- subset(pfb,((GEO_FOR_C > 0) & (pfb$GEO_FOR_C <= 690)))

pfb2 <- subset(pfb,((GEO_FOR_C > 690) | (pfb$GEO_FOR_C == 0) |
     (is.na(pfb$GEO_FOR_C))))
rm(pfb)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")

top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
pfb1$bulk <- NA
pfb1$GEO_FOR_C <- as.character(pfb1$GEO_FOR_C)
t <- (!is.na(pfb1$A_HORIZON))
pfb1[t,]$bulk <- top[pfb1[t,]$GEO_FOR_C]
pfb1[!t,]$bulk <- sub[pfb1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
pfb2$bulk <- NA
pfb2 <- subset(pfb2,!is.na(pfb2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.4
```

```
s2 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.8

s4 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.65

s5 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50) )
pfb2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.5

s2 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.65

s4 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.6

s5 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50))
pfb2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
pfb <- merge(pfb1,pfb2,all=TRUE)
rm(pfb1,pfb2)

# Exclude samples collected from more than 1 soil horizon - PFB

isCrossing <- (!is.na(pfb$MON_VNR)  &
    ((pfb$MON_DIEPB < pfb$HOR_DIEPB) & (pfb$MON_DIEPO > pfb$HOR_DIEPB)) |
    ((pfb$MON_DIEPB < pfb$HOR_DIEPO) & (pfb$MON_DIEPO > pfb$HOR_DIEPO))
)

pfb <- pfb[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - PFB

# Weighted average of organic matter

# Subset removing NA records in ORGSTOF column
pfb <- subset(x = pfb,subset = !is.na(pfb$ORGSTOF))

# Weighted averiging of organic carbon content

pfb$s_thick <- pfb$MON_DIEPO - pfb$MON_DIEPB
pfb$s_thick.org <- pfb$s_thick * pfb$ORGSTOF

pfb$HOR_ID <- as.character(pfb$LAAG_ID)

sum1 <- tapply(X = pfb$s_thick.org, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_org <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Weighted average of bulk density

pfb$s_thick.bulk <- pfb$s_thick * pfb$bulk

sum1 <- tapply(X = pfb$s_thick.bulk, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]
```

```
sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_bulk <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
pfb <- subset(x = pfb,subset = duplicated(pfb$LAAG_ID)==FALSE)

# Horizons selection - PFB

pfb <- subset(x = pfb,subset =
    (HOR_DIEPB> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB< d1 & HOR_DIEPO> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB> d1 & HOR_DIEPB< d2 & HOR_DIEPO> d2) |
    (HOR_DIEPB<=d1 & HOR_DIEPO>=d2) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
pfb$h_thick <- 0

s1 <- ((pfb$HOR_DIEPB > d1) & (pfb$HOR_DIEPO < d2))
pfb[s1,]$h_thick <- pfb[s1,]$HOR_DIEPO - pfb[s1,]$HOR_DIEPB

s2 <- ((pfb$HOR_DIEPB < d1) & (pfb$HOR_DIEPO > d1) & (pfb$HOR_DIEPO < d2))
pfb[s2,]$h_thick <- pfb[s2,]$HOR_DIEPO - d1

s3 <- ((pfb$HOR_DIEPB > d1) & (pfb$HOR_DIEPB < d2) & (pfb$HOR_DIEPO > d2))
pfb[s3,]$h_thick <- d2 - pfb[s3,]$HOR_DIEPB

s4 <- ((pfb$HOR_DIEPB <= d1) & (pfb$HOR_DIEPO >= d2))
pfb[s4,]$h_thick <- d2 - d1

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
pfb$ratio <- (pfb$h_thick/(d2-d1))

# The sum of ratios
pfb$PROF_ID <- as.character(pfb$PROFIEL_ID)
sum_ratio <- tapply(X = pfb$ratio, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum_ratio <- sum_ratio[pfb$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
pfb <- subset(x = pfb, subset = sum_ratio > 0.999)

# Profile weighted averaging - PFB

pfb$SoilMass <- pfb$h_thick * pfb$l_bulk
pfb$SoilMass.l_org <- pfb$SoilMass * pfb$l_org

sum1 <- tapply(X = pfb$SoilMass.l_org, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$PROF_ID]

sum2 <- tapply(X = pfb$SoilMass, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$PROF_ID]

pfb$wa_org <- pfb$sum1/ pfb$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
pfb <- subset(x = pfb, subset = duplicated(pfb$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = pfb$X, Y = pfb$Y)
dup2 <- duplicated(dup)

pfb$dupl_loc <- dup2
rm(dup,dup2)

pfb <- subset(x = pfb, subset = (dupl_loc==FALSE))
```

```
# All available organic matter data for the depth d1 - d2
pfb <- data.frame(X = pfb$X, Y = pfb$Y, org = pfb$wa_org)


# Pawn soil types - PFB table

# overlay sampling locaations with pawn-map
coordinates(pfb) <- ~ X+Y
o <- overlay(map,pfb)

pfb$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_LSK_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_lsk_deltabis view
lsk <- sqlFetch(channel = channel, sqtable = "V_LSK_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$MON_DATUM) & lsk$MON_DATUM > from &
     lsk$MON_DATUM < to))

# Spatial extent filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$X) & lsk$X > X_MIN & lsk$X < X_MAX))
lsk <- subset(lsk,(!is.na(lsk$Y) & lsk$Y > Y_MIN & lsk$Y < Y_MAX))

# Bulk density - LSK

# Split the table into two dataset based on geocode value
# to apply two different aproaches

lsk1 <- subset(lsk,((GEO_FOR_C > 0) & (lsk$GEO_FOR_C <= 690)))

lsk2 <- subset(lsk,((GEO_FOR_C > 690) | (lsk$GEO_FOR_C == 0) |
     (is.na(lsk$GEO_FOR_C))))
rm(lsk)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")

top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
lsk1$bulk <- NA
lsk1$GEO_FOR_C <- as.character(lsk1$GEO_FOR_C)
t <- (!is.na(lsk1$A_HORIZON))
lsk1[t,]$bulk <- top[lsk1[t,]$GEO_FOR_C]
lsk1[!t,]$bulk <- sub[lsk1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
lsk2$bulk <- NA
lsk2 <- subset(lsk2,!is.na(lsk2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.4

s2 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.8

s4 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.65
```

```
s5 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50) )
lsk2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.5

s2 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.65

s4 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.6

s5 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50))
lsk2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
lsk <- merge(lsk1,lsk2,all=TRUE)
rm(lsk1,lsk2)

# Exclude samples collected from more than 1 soil horizon - LSK

isCrossing <- (!is.na(lsk$MON_VNR)  &
    ((lsk$MON_DIEPB < lsk$HOR_DIEPB) & (lsk$MON_DIEPO > lsk$HOR_DIEPB)) |
    ((lsk$MON_DIEPB < lsk$HOR_DIEPO) & (lsk$MON_DIEPO > lsk$HOR_DIEPO))
)

lsk <- lsk[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - LSK

# Subset removing NA records in ORGSTOF column
lsk <- subset(x = lsk,subset = !is.na(lsk$ORGSTOF))

# Weighted averiging

lsk$s_thick <- lsk$MON_DIEPO - lsk$MON_DIEPB
lsk$s_thick.org <- lsk$s_thick * lsk$ORGSTOF

lsk$HOR_ID <- as.character(lsk$LAAG_ID)

sum1 <- tapply(X = lsk$s_thick.org, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_org <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Weighted averaging of bulk density

lsk$s_thick.bulk <- lsk$s_thick * lsk$bulk

sum1 <- tapply(X = lsk$s_thick.bulk, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_bulk <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
lsk <- subset(x = lsk,subset = duplicated(lsk$LAAG_ID)==FALSE)

# Horizons selection - LSK
```

```
lsk <- subset(x = lsk,subset =
    (HOR_DIEPB> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB< d1 & HOR_DIEPO> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB> d1 & HOR_DIEPB< d2 & HOR_DIEPO> d2) |
    (HOR_DIEPB<=d1 & HOR_DIEPO>=d2) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
lsk$h_thick <- 0

s1 <- ((lsk$HOR_DIEPB > d1) & (lsk$HOR_DIEPO < d2))
lsk[s1,]$h_thick <- lsk[s1,]$HOR_DIEPO - lsk[s1,]$HOR_DIEPB

s2 <- ((lsk$HOR_DIEPB < d1) & (lsk$HOR_DIEPO > d1) & (lsk$HOR_DIEPO < d2))
lsk[s2,]$h_thick <- lsk[s2,]$HOR_DIEPO - d1

s3 <- ((lsk$HOR_DIEPB > d1) & (lsk$HOR_DIEPB < d2) & (lsk$HOR_DIEPO > d2))
lsk[s3,]$h_thick <- d2 - lsk[s3,]$HOR_DIEPB

s4 <- ((lsk$HOR_DIEPB <= d1) & (lsk$HOR_DIEPO >= d2))
lsk[s4,]$h_thick <- d2 - d1

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
lsk$ratio <- (lsk$h_thick/(d2-d1))

# The sum of ratios
lsk$PROF_ID <- as.character(lsk$PROFIEL_ID)
sum_ratio <- tapply(X = lsk$ratio, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum_ratio <- sum_ratio[lsk$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
lsk <- subset(x = lsk,subset = sum_ratio > 0.999)

# Profile weighted averaging - LSK

lsk$SoilMass <- lsk$h_thick * lsk$l_bulk
lsk$SoilMass.l_org <- lsk$SoilMass * lsk$l_org

sum1 <- tapply(X = lsk$SoilMass.l_org, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$PROF_ID]

sum2 <- tapply(X = lsk$SoilMass, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$PROF_ID]

lsk$wa_org <- lsk$sum1/ lsk$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
lsk <- subset(x = lsk, subset = duplicated(lsk$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = lsk$X, Y = lsk$Y)
dup2 <- duplicated(dup)

lsk$dupl_loc <- dup2
rm(dup,dup2)

lsk <- subset(x = lsk, subset = (dupl_loc==FALSE))

# Available organic matter data for the depth d1 - d2
lsk <- data.frame(X = lsk$X, Y = lsk$Y, org = lsk$wa_org)


# Pawn soil types - LSK

# Overlay sampling locations with pawn map
coordinates(lsk) <- ~ X+Y
o <- overlay(map,lsk)

lsk$pawn <- o$PAWN_CODE
```

```
rm(o)

# Merge tables

org <- merge(pfb,lsk,all=TRUE)
org <- merge(org,augering,all=TRUE)
rm(pfb,lsk,augering)

# Exlude locations which fall into the water,build up areas, and NA
org <- subset(x = org, subset = (pawn < 22) & (!is.na(pawn)))

# Replace organic matter where 0 with 0.2
s <- (org$org==0)
org[s,]$org <- 0.2
rm(s)

# Write table to the text file
write.table(org,"org.txt",sep="\t",row.names=FALSE)

# 26 Save Box-plots
png("box_plot.png",width=800,height=600,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
boxplot(split(org$org,org$pawn),xlab="Pawn soil type",main="Organic matter",
     cex.main=2)
dev.off()


# 27 Histograms

# histogram of original dataset
png("hist.png",width=400,height=400,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(org$org,main="",xlab="Organic matter",col="lightblue")
dev.off()
```

D.2 Variography

```
# Variography

# Log transformation
org$log_org <- log(org$org+1)

# Compute standartized residuals

mean <- tapply(org$log_org,org$pawn,mean)
org$mean <- mean[org$pawn]

sd <- tapply(org$log_org,org$pawn,sd)
org$sd <- sd[org$pawn]

org$residua <- (org$log_org - org$mean) / org$sd

# Random pick to speed up variogram estimation
org1 <- org[sample(nrow(org),size=10000,replace=F),]

# Variography
coordinates(org1) <- ~X+Y

gORG <- gstat(id = c("ORG"), formula = residua ~ 1, data = org1)
gORG.vg <- variogram(gORG,boundaries=c(500,700,1000,1500,2500,10000,20000,30000,
     40000,50000,60000,70000,80000,90000,100000,110000))

# Fit nested variogram model automatically
vgm <- fit.variogram(gORG.vg, vgm(psill= 1,model="Exp",range=100000,
add.to=vgm(psill= 1,model="Sph",range=1000,nugget=1)),fit.sill=TRUE,fit.ranges=TRUE)

# Save residual variogram
png("variogram.png",width=600,height=600,units="px",pointsize=12,bg="white",res=NA)
fontsize <- trellis.par.get("fontsize")
fontsize$text <- 20
fontsize$points <- 20
```

```
trellis.par.set("fontsize",fontsize)
plot(gORG.vg,vgm,main="Organic matter - residual variogram",pch=20)
dev.off()
```

D.3 Define interpolation grid

```
# Define interpolation grid - chose one option only

# 1) rectangular grid

######### Set parameters ##########

# Output map extent
x.min <- 10000
x.max <- 280000
y.min <- 300000
y.max <- 620000

# Prediction grid cell size (output map resolution)
cs <- 1000

####################################

x <- seq(x.min,x.max,cs)
y <- seq(y.min,y.max,cs)
grid <- expand.grid(x1=x,x2=y)
coordinates(grid) <- ~x1+x2
o <- overlay(map,grid)
grid <- data.frame(grid,pawn=o$PAWN_CODE)
grid <- subset(grid,(pawn < 22 & !is.na(pawn)))
coordinates(grid) <- ~x1+x2
rm(o)


# 2) sampling within polygon(s)

######### Set parameters ##########

# Prediction grid cell size (output map resolution)
cs <- 1000

####################################

# Read ESRI shapefile

# the whole NL boundary (en example)
nl_poly <- readShapePoly("NL.shp")

grid <- spsample(nl_poly,cellsize=cs,type="regular")
o <- overlay(map,grid)
grid <- data.frame(grid,pawn=o$PAWN_CODE)
grid <- subset(grid,(pawn < 22 & !is.na(pawn)))
coordinates(grid)<- ~x1+x2
rm(o)
```

D.4 Stochastic simulation

```
# Stochastic simulation

######## Set parameters ################

# Number of simulations
ns <- 1

# Point or block kriging
# if "block <- 0" ... point kriging
```

```
# if "block <- 1" ... block kriging
block <- 0

# Block size (for block kriging only)
bs <- c(10,10)

########################################

coordinates(org) <- ~X+Y

# Define gstat object to predict residuals

gstatORG <- gstat(id = c("ORG"), formula = residua ~ 1, data = org, model = vgm, beta = 0, nmax = 30)

# Ready to predict organic matter residuals using Gaussian simulation based on
# simple kriging
if (block==0)   blup <- predict.gstat(gstatORG, newdata = grid, BLUE = FALSE,
   nsim = ns) else
            blup <- predict.gstat(gstatORG, newdata = grid, BLUE = FALSE,
   nsim = ns,block=bs)

# Simulations computation
sim <- data.frame(blup, pawn = grid$pawn)

sim$mean <- mean[sim$pawn]

sim$sd <- sd[sim$pawn]

sim_ns <- data.frame(x1 = sim$x1, x2 = sim$x2)

sim_ns[,3:(ns+2)] <- sim[,3:(ns+2)] * sim$sd + sim$mean

# Back transformation
sim_ns[,3:(ns+2)] <- exp(sim_ns[,3:(ns+2)]) - 1

# Save figures
coordinates(sim_ns) <- ~x1+x2
gridded(sim_ns) <- TRUE

nl_poly <- readShapePoly("NL.shp")
text <- list("sp.text",c(60000,550000),paste(d1,"-",d2,"cm"),cex=1.5)
nl <- list("sp.polygons", nl_poly)
rgb.palette <- colorRampPalette(c("lightyellow","green","orange","brown"),
     space = "rgb")

for (i in 1:ns){
png(filename = paste("sim",i,".png",sep=""), width=600, height=600, units="px",
     pointsize=12, bg="white", res=NA)
print(spplot(sim_ns[,i], main = list(label=paste("Organic matter -
     simulation",i),cex=1.5),
col.regions=rgb.palette(22),
xlab=list(paste(d1,"-",d2,"cm"),cex=1.5),
at=c(seq(0,10,0.5),120),
colorkey=list(space="right",width=2,at=1:22,labels=list(cex=2,at=1:22,
labels=c("","","1","","","","3","","","","5","","","","7","","","",
     "9 >","","","",""))),
sp.layout=list(nl,text)))
dev.off()
}
```

D.5 Prediction

```
# Prediction

############ Set parameters #############

# Point or block kriging
# "block <- 0" ... point kriging
# "block <- 1" ... block kriging
block <- 0
```

```
# Block size (for block kriging only)
bs <- c(10,10)

########################################

coordinates(org) <- ~X+Y

# Define gstat object to predict residuals

gstatORG <- gstat(id = c("ORG"), formula = residua ~ 1, data = org, model = vgm,
     beta = 0, nmax = 30)

# Ready to make pH residual prediction using simple kriging
if(block==0)    blup2 <- predict.gstat(gstatORG, newdata = grid, BLUE = FALSE) else
            blup2 <- predict.gstat(gstatORG, newdata = grid, BLUE = FALSE,block=bs)

# Compute prediction
pred <- data.frame(blup2, pawn = grid$pawn)

pred$mean <- mean[pred$pawn]

pred$sd <- sd[pred$pawn]


pred$pred <- pred$ORG.pred * pred$sd + pred$mean

# Back transformation
pred$pred <- exp(pred$pred) - 1

coordinates(pred) <- ~x1+x2
gridded(pred) <- TRUE

# Standart deviation map
pred$pred_sd <- sqrt(pred$ORG.var)
pred$map_sd <- pred$sd * pred$pred_sd

# 95 lower limit map
pred$lower95 <- pred$pred - 1.96 * pred$map_sd

# 95 upper limit map
pred$upper95 <- pred$pred + 1.96 * pred$map_sd

# Save figures
nl_poly <- readShapePoly("NL.shp")
text <- list("sp.text",c(60000,550000),paste(d1,"-",d2,"cm"),cex=1.5)
nl <- list("sp.polygons", nl_poly)
rgb.palette <- colorRampPalette(c("lightyellow","green","orange","brown"),
     space = "rgb")

# Prediction map
png(filename="pred.png",width=600,height=600,units="px",pointsize=12,bg="white",
     res=NA)
spplot(pred["pred"],main=list(label="Organic matter - prediction",cex=1.5),
col.regions=rgb.palette(22),
xlab=list(paste(d1,"-",d2,"cm"),cex=1.5),
at=c(seq(0,10,0.5),120),
colorkey=list(space="right",width=2,at=1:22,labels=list(cex=2,at=1:22,
labels=c("","","1","","","","3","","","","5","","","","7","","","",
     "9 >","","","",""))),
sp.layout=list(nl,text))
dev.off()

# Standart deviation map
png(filename="stdev.png",width=600,height=600,units="px",pointsize=12,bg="white",
     res=NA)
spplot(pred["map_sd"],main=list(label="Organic matter - standart deviation",
     cex=1.5),
col.regions=rgb.palette(22),
xlab=list(paste(d1,"-",d2,"cm"),cex=1.5),
colorkey=list(space="right",width=2),
sp.layout=list(nl,text))
dev.off()

# 95 lower limit  map
png(filename="lower95.png",width=600,height=600,units="px",pointsize=12,
```

```
     bg="white",res=NA)
spplot(pred["lower95"],main=list(label="Organic matter - 95 lower limit",cex=1.5),
col.regions=rgb.palette(22),
xlab=list(paste(d1,"-",d2,"cm"),cex=1.5),
at=c(seq(0,10,0.5),120),
colorkey=list(space="right",width=2,at=1:22,labels=list(cex=2,at=1:22,
labels=c("","","1","","","","3","","","","5","","","","7","","","",
     "9 >","","","",""))),
sp.layout=list(nl,text))
dev.off()

# 95 upper limit  map
png(filename="upper95.png",width=600,height=600,units="px",pointsize=12,
     bg="white",res=NA)
spplot(pred["upper95"],main=list(label="Organic matter - 95 upper limit",cex=1.5),
col.regions=rgb.palette(22),
xlab=list(paste(d1,"-",d2,"cm"),cex=1.5),
at=c(seq(0,10,0.5),120),
colorkey=list(space="right",width=2,at=1:22,labels=list(cex=2,at=1:22,
labels=c("","","1","","","","3","","","","5","","","","7","","","",
     "9 >","","","",""))),
sp.layout=list(nl,text))
dev.off()
```

Appendix E

R code Organic Matter Content, multiple layer analysis

E.1 First layer data extraction and preparation

```
# First layer data extraction and preparation

############ Set parameters ###############

# Load libraries

library(RODBC)
library(foreign)
library(sp)
library(maptools)
library(gstat)
library(lattice)

# The thickness of the first soil layer (cm)
d1 <- 10
d2 <- 15

# Data filtering (spatial extent of input data)
X_MIN <- 0
X_MAX <- 300000
Y_MIN <- 300000
Y_MAX <- 620000

# Time filtering: yyyy-mm-dd

from <- as.POSIXct("1000-01-01")
to   <- as.POSIXct("3000-01-01")


##########################################

# View extraction - V_Augering_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_augering_deltabis view
augering <- sqlFetch(channel = channel, sqtable = "V_AUGERING_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - Augering
```

```
augering$MON_DATUM <- paste(augering$MON_DATUM,"-01-01",sep="")
augering$MON_DATUM <- as.POSIXct(augering$MON_DATUM)

augering <- subset(augering,(!is.na(augering$MON_DATUM) & augering$MON_DATUM >
     from & augering$MON_DATUM < to))

# Spatial extent filtering - Augering
augering <- subset(augering,(!is.na(augering$X) & augering$X > X_MIN & augering$X < X_MAX))
augering <- subset(augering,(!is.na(augering$Y) & augering$Y > Y_MIN & augering$Y < Y_MAX))

# Horizons selection - Augering

# Exclude records with missing values
augering <- subset(x = augering, subset = (!is.na(ORGSTOF) & ORGSTOF > 0))

# Horizons selection
augering <- subset(x = augering,subset =
    (HOR_DIEPB > d1 & HOR_DIEPO < d2) |
    (HOR_DIEPB < d1 & HOR_DIEPO > d1 & HOR_DIEPO < d2) |
    (HOR_DIEPB > d1 & HOR_DIEPB < d2 & HOR_DIEPO > d2) |
    (HOR_DIEPB <= d1 & HOR_DIEPO >= d2) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
augering$h_thick <- 0

s1 <- ((augering$HOR_DIEPB > d1) & (augering$HOR_DIEPO < d2))
augering[s1,]$h_thick <- augering[s1,]$HOR_DIEPO - augering[s1,]$HOR_DIEPB

s2 <- ((augering$HOR_DIEPB < d1) & (augering$HOR_DIEPO > d1) & (augering$HOR_DIEPO < d2))
augering[s2,]$h_thick <- augering[s2,]$HOR_DIEPO - d1

s3 <- ((augering$HOR_DIEPB > d1) & (augering$HOR_DIEPB < d2) & (augering$HOR_DIEPO > d2))
augering[s3,]$h_thick <- d2 - augering[s3,]$HOR_DIEPB

s4 <- ((augering$HOR_DIEPB <= d1) & (augering$HOR_DIEPO >= d2))
augering[s4,]$h_thick <- d2 - d1

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
augering$ratio <- (augering$h_thick/(d2-d1))

# The sum of ratios
augering$PROF_ID <- as.character(augering$PROFIEL_ID)
sum_ratio <- tapply(X = augering$ratio, INDEX = augering$PROF_ID, FUN = sum)
augering$sum_ratio <- sum_ratio[augering$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
augering <- subset(x = augering,subset = sum_ratio > 0.999)

# Bulk density - Augering

# Split the table into two dataset based on geocode value
# to apply two different aproaches

augering1 <- subset(x = augering, subset = ((GEO_FOR_C > 0) & (augering$GEO_FOR_C
     <= 690)))

augering2 <- subset(x = augering, subset = ((GEO_FOR_C > 690) |
     (augering$GEO_FOR_C == 0) | (is.na(augering$GEO_FOR_C))))
rm(augering)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")

top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
augering1$bulk <- NA
augering1$GEO_FOR_C <- as.character(augering1$GEO_FOR_C)
t <- (!is.na(augering1$A_HORIZON))
```

```
augering1[t,]$bulk <- top[augering1[t,]$GEO_FOR_C]
augering1[!t,]$bulk <- sub[augering1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
augering2$bulk <- NA
augering2 <- subset(augering2,!is.na(augering2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.4

s2 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.8

s4 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.65

s5 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 50) )
augering2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.5

s2 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.65

s4 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.6

s5 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 50))
augering2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
augering <- merge(augering1,augering2,all=TRUE)
rm(augering1,augering2)

# Profile Weighted averaging - Augering

augering$SoilMass <- augering$h_thick * augering$bulk
augering$SoilMass.ORGSTOF <- augering$SoilMass * augering$ORGSTOF

sum1 <- tapply(X = augering$SoilMass.ORGSTOF, INDEX = augering$PROF_ID, FUN = sum)
augering$sum1 <- sum1[augering$PROF_ID]

sum2 <- tapply(X = augering$SoilMass, INDEX = augering$PROF_ID, FUN = sum)
augering$sum2 <- sum2[augering$PROF_ID]

augering$wa_org <- augering$sum1/ augering$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
augering <- subset(x = augering, subset = duplicated(augering$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = augering$X, Y = augering$Y)
dup2 <- duplicated(dup)
```

```
augering$dupl_loc <- dup2
rm(dup,dup2)

augering <- subset(x = augering, subset = (dupl_loc==FALSE))

# All available organic matter data for the depth d1 - d2
augering <- data.frame(X = augering$X, Y = augering$Y, org = augering$wa_org)


# Pawn soil types - Augering

# read the pawn_map08.shp file
map <- readShapePoly("pawn-map08.shp",IDvar="PAWN_ID")

# overlay sampling locaations with pawn-map
coordinates(augering) <- ~X+Y
o <- overlay(map,augering)

augering$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_PFB_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_pfb_deltabis view
pfb <- sqlFetch(channel = channel, sqtable = "V_PFB_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$MON_DATUM) & pfb$MON_DATUM > from & pfb$MON_DATUM
     < to))

# Spatial extent filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$X) & pfb$X > X_MIN & pfb$X < X_MAX))
pfb <- subset(pfb,(!is.na(pfb$Y) & pfb$Y > Y_MIN & pfb$Y < Y_MAX))

# Bulk density - PFB

# Split the table into two dataset based on geocode value
# to apply two different aproaches

pfb1 <- subset(pfb,((GEO_FOR_C > 0) & (pfb$GEO_FOR_C <= 690)))

pfb2 <- subset(pfb,((GEO_FOR_C > 690) | (pfb$GEO_FOR_C == 0) |
     (is.na(pfb$GEO_FOR_C))))
rm(pfb)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")

top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
pfb1$bulk <- NA
pfb1$GEO_FOR_C <- as.character(pfb1$GEO_FOR_C)
t <- (!is.na(pfb1$A_HORIZON))
pfb1[t,]$bulk <- top[pfb1[t,]$GEO_FOR_C]
pfb1[!t,]$bulk <- sub[pfb1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
pfb2$bulk <- NA
pfb2 <- subset(pfb2,!is.na(pfb2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.4
```

```
s2 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.8

s4 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.65

s5 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50) )
pfb2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.5

s2 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.65

s4 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.6

s5 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50))
pfb2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
pfb <- merge(pfb1,pfb2,all=TRUE)
rm(pfb1,pfb2)

# Exclude samples collected from more than 1 soil horizon - PFB

isCrossing <- (!is.na(pfb$MON_VNR)  &
    ((pfb$MON_DIEPB < pfb$HOR_DIEPB) & (pfb$MON_DIEPO > pfb$HOR_DIEPB)) |
    ((pfb$MON_DIEPB < pfb$HOR_DIEPO) & (pfb$MON_DIEPO > pfb$HOR_DIEPO))
)

pfb <- pfb[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - PFB

# Weighted average of organic matter

# Subset removing NA records in ORGSTOF column
pfb <- subset(x = pfb,subset = !is.na(pfb$ORGSTOF))

# Weighted averiging of organic carbon content

pfb$s_thick <- pfb$MON_DIEPO - pfb$MON_DIEPB
pfb$s_thick.org <- pfb$s_thick * pfb$ORGSTOF

pfb$HOR_ID <- as.character(pfb$LAAG_ID)

sum1 <- tapply(X = pfb$s_thick.org, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_org <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Weighted average of bulk density

pfb$s_thick.bulk <- pfb$s_thick * pfb$bulk

sum1 <- tapply(X = pfb$s_thick.bulk, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
```

```
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_bulk <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
pfb <- subset(x = pfb,subset = duplicated(pfb$LAAG_ID)==FALSE)

# Horizons selection - PFB

pfb <- subset(x = pfb,subset =
    (HOR_DIEPB> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB< d1 & HOR_DIEPO> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB> d1 & HOR_DIEPB< d2 & HOR_DIEPO> d2) |
    (HOR_DIEPB<=d1 & HOR_DIEPO>=d2) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
pfb$h_thick <- 0

s1 <- ((pfb$HOR_DIEPB > d1) & (pfb$HOR_DIEPO < d2))
pfb[s1,]$h_thick <- pfb[s1,]$HOR_DIEPO - pfb[s1,]$HOR_DIEPB

s2 <- ((pfb$HOR_DIEPB < d1) & (pfb$HOR_DIEPO > d1) & (pfb$HOR_DIEPO < d2))
pfb[s2,]$h_thick <- pfb[s2,]$HOR_DIEPO - d1

s3 <- ((pfb$HOR_DIEPB > d1) & (pfb$HOR_DIEPB < d2) & (pfb$HOR_DIEPO > d2))
pfb[s3,]$h_thick <- d2 - pfb[s3,]$HOR_DIEPB

s4 <- ((pfb$HOR_DIEPB <= d1) & (pfb$HOR_DIEPO >= d2))
pfb[s4,]$h_thick <- d2 - d1

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
pfb$ratio <- (pfb$h_thick/(d2-d1))

# The sum of ratios
pfb$PROF_ID <- as.character(pfb$PROFIEL_ID)
sum_ratio <- tapply(X = pfb$ratio, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum_ratio <- sum_ratio[pfb$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
pfb <- subset(x = pfb, subset = sum_ratio > 0.999)

# Profile weighted averaging - PFB

pfb$SoilMass <- pfb$h_thick * pfb$l_bulk
pfb$SoilMass.l_org <- pfb$SoilMass * pfb$l_org

sum1 <- tapply(X = pfb$SoilMass.l_org, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$PROF_ID]

sum2 <- tapply(X = pfb$SoilMass, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$PROF_ID]

pfb$wa_org <- pfb$sum1/ pfb$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
pfb <- subset(x = pfb, subset = duplicated(pfb$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = pfb$X, Y = pfb$Y)
dup2 <- duplicated(dup)

pfb$dupl_loc <- dup2
rm(dup,dup2)

pfb <- subset(x = pfb, subset = (dupl_loc==FALSE))

# All available organic matter data for the depth d1 - d2
```

```
pfb <- data.frame(X = pfb$X, Y = pfb$Y, org = pfb$wa_org)


# Pawn soil types - PFB table

# overlay sampling locaations with pawn-map
coordinates(pfb) <- ~ X+Y
o <- overlay(map,pfb)

pfb$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_LSK_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_lsk_deltabis view
lsk <- sqlFetch(channel = channel, sqtable = "V_LSK_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$MON_DATUM) & lsk$MON_DATUM > from & lsk$MON_DATUM
     < to))

# Spatial extent filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$X) & lsk$X > X_MIN & lsk$X < X_MAX))
lsk <- subset(lsk,(!is.na(lsk$Y) & lsk$Y > Y_MIN & lsk$Y < Y_MAX))

# Bulk density - LSK

# Split the table into two dataset based on geocode value
# to apply two different aproaches

lsk1 <- subset(lsk,((GEO_FOR_C > 0) & (lsk$GEO_FOR_C <= 690)))

lsk2 <- subset(lsk,((GEO_FOR_C > 690) | (lsk$GEO_FOR_C == 0) |
     (is.na(lsk$GEO_FOR_C))))
rm(lsk)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")

top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
lsk1$bulk <- NA
lsk1$GEO_FOR_C <- as.character(lsk1$GEO_FOR_C)
t <- (!is.na(lsk1$A_HORIZON))
lsk1[t,]$bulk <- top[lsk1[t,]$GEO_FOR_C]
lsk1[!t,]$bulk <- sub[lsk1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
lsk2$bulk <- NA
lsk2 <- subset(lsk2,!is.na(lsk2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.4

s2 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.8

s4 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.65
```

```
s5 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50) )
lsk2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.5

s2 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.65

s4 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.6

s5 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50))
lsk2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
lsk <- merge(lsk1,lsk2,all=TRUE)
rm(lsk1,lsk2)

# Exclude samples collected from more than 1 soil horizon - LSK

isCrossing <- (!is.na(lsk$MON_VNR)  &
    ((lsk$MON_DIEPB < lsk$HOR_DIEPB) & (lsk$MON_DIEPO > lsk$HOR_DIEPB)) |
    ((lsk$MON_DIEPB < lsk$HOR_DIEPO) & (lsk$MON_DIEPO > lsk$HOR_DIEPO))
)

lsk <- lsk[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - LSK

# Subset removing NA records in ORGSTOF column
lsk <- subset(x = lsk,subset = !is.na(lsk$ORGSTOF))

# Weighted averiging

lsk$s_thick <- lsk$MON_DIEPO - lsk$MON_DIEPB
lsk$s_thick.org <- lsk$s_thick * lsk$ORGSTOF

lsk$HOR_ID <- as.character(lsk$LAAG_ID)

sum1 <- tapply(X = lsk$s_thick.org, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_org <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Weighted averaging of bulk density

lsk$s_thick.bulk <- lsk$s_thick * lsk$bulk

sum1 <- tapply(X = lsk$s_thick.bulk, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_bulk <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
lsk <- subset(x = lsk,subset = duplicated(lsk$LAAG_ID)==FALSE)

# Horizons selection - LSK

lsk <- subset(x = lsk,subset =
```

```
    (HOR_DIEPB> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB< d1 & HOR_DIEPO> d1 & HOR_DIEPO< d2) |
    (HOR_DIEPB> d1 & HOR_DIEPB< d2 & HOR_DIEPO> d2) |
    (HOR_DIEPB<=d1 & HOR_DIEPO>=d2) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
lsk$h_thick <- 0

s1 <- ((lsk$HOR_DIEPB > d1) & (lsk$HOR_DIEPO < d2))
lsk[s1,]$h_thick <- lsk[s1,]$HOR_DIEPO - lsk[s1,]$HOR_DIEPB

s2 <- ((lsk$HOR_DIEPB < d1) & (lsk$HOR_DIEPO > d1) & (lsk$HOR_DIEPO < d2))
lsk[s2,]$h_thick <- lsk[s2,]$HOR_DIEPO - d1

s3 <- ((lsk$HOR_DIEPB > d1) & (lsk$HOR_DIEPB < d2) & (lsk$HOR_DIEPO > d2))
lsk[s3,]$h_thick <- d2 - lsk[s3,]$HOR_DIEPB

s4 <- ((lsk$HOR_DIEPB <= d1) & (lsk$HOR_DIEPO >= d2))
lsk[s4,]$h_thick <- d2 - d1

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
lsk$ratio <- (lsk$h_thick/(d2-d1))

# The sum of ratios
lsk$PROF_ID <- as.character(lsk$PROFIEL_ID)
sum_ratio <- tapply(X = lsk$ratio, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum_ratio <- sum_ratio[lsk$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
lsk <- subset(x = lsk,subset = sum_ratio > 0.999)

# Profile weighted averaging - LSK

lsk$SoilMass <- lsk$h_thick * lsk$l_bulk
lsk$SoilMass.l_org <- lsk$SoilMass * lsk$l_org

sum1 <- tapply(X = lsk$SoilMass.l_org, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$PROF_ID]

sum2 <- tapply(X = lsk$SoilMass, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$PROF_ID]

lsk$wa_org <- lsk$sum1/ lsk$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
lsk <- subset(x = lsk, subset = duplicated(lsk$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = lsk$X, Y = lsk$Y)
dup2 <- duplicated(dup)

lsk$dupl_loc <- dup2
rm(dup,dup2)

lsk <- subset(x = lsk, subset = (dupl_loc==FALSE))

# Available organic matter data for the depth d1 - d2
lsk <- data.frame(X = lsk$X, Y = lsk$Y, org = lsk$wa_org)


# Pawn soil types - LSK

# Overlay sampling locations with pawn map
coordinates(lsk) <- ~ X+Y
o <- overlay(map,lsk)

lsk$pawn <- o$PAWN_CODE
rm(o)
```

```
# Merge tables

org <- merge(pfb,lsk,all=TRUE)
org <- merge(org,augering,all=TRUE)
rm(pfb,lsk,augering)

# Exlude locations which fall into the water,build up areas, and NA
org <- subset(x = org, subset = (pawn < 22) & (!is.na(pawn)))

# Replace organic matter where 0 with 0.2
s <- (org$org==0)
org[s,]$org <- 0.2
rm(s)

# Rename table
org1 <- org
rm(org)

# Write table to the text file
write.table(org1,"org1.txt",sep="\t",row.names=FALSE)

# 26 Save Box-plots
png("box_plot.png",width=800,height=600,units="px",pointsize=12,bg="white",
     res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
boxplot(split(org1$org,org1$pawn),xlab="Pawn soil type",main="Organic matter",
     cex.main=2)
dev.off()


# 27 Histograms

# histogram of original dataset
png("hist.png",width=400,height=400,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(org1$org,main="",xlab="Organic matter",col="lightblue")
dev.off()
```

E.2 Second layer data extraction and preparation

```
# Second layer data extraction and preparation

############ Set parameters ###############

# The thickness of the second soil layer (cm)
d3 <- 30
d4 <- 45

##########################################

# View extraction - V_Augering_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_augering_deltabis view
augering <- sqlFetch(channel = channel, sqtable = "V_AUGERING_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - Augering
augering$MON_DATUM <- paste(augering$MON_DATUM,"-01-01",sep="")
augering$MON_DATUM <- as.POSIXct(augering$MON_DATUM)

augering <- subset(augering,(!is.na(augering$MON_DATUM) & augering$MON_DATUM >
     from & augering$MON_DATUM < to))

# Spatial extent filtering - Augering
```

```
augering <- subset(augering,(!is.na(augering$X) & augering$X > X_MIN & augering$X < X_MAX))
augering <- subset(augering,(!is.na(augering$Y) & augering$Y > Y_MIN & augering$Y < Y_MAX))

# Horizons selection - Augering

# Exclude records with missing values
augering <- subset(x = augering, subset = (!is.na(ORGSTOF) & ORGSTOF > 0))

# Horizons selection
augering <- subset(x = augering,subset =
    (HOR_DIEPB > d3 & HOR_DIEPO < d4) |
    (HOR_DIEPB < d3 & HOR_DIEPO > d3 & HOR_DIEPO < d4) |
    (HOR_DIEPB > d3 & HOR_DIEPB < d4 & HOR_DIEPO > d4) |
    (HOR_DIEPB <= d3 & HOR_DIEPO >= d4) )

# Computing the thickness of a part of horizon which contribute to the
# required soil layer
augering$h_thick <- 0

s1 <- ((augering$HOR_DIEPB > d3) & (augering$HOR_DIEPO < d4))
augering[s1,]$h_thick <- augering[s1,]$HOR_DIEPO - augering[s1,]$HOR_DIEPB

s2 <- ((augering$HOR_DIEPB < d3) & (augering$HOR_DIEPO > d3) & (augering$HOR_DIEPO < d4))
augering[s2,]$h_thick <- augering[s2,]$HOR_DIEPO - d3

s3 <- ((augering$HOR_DIEPB > d3) & (augering$HOR_DIEPB < d4) & (augering$HOR_DIEPO > d4))
augering[s3,]$h_thick <- d4 - augering[s3,]$HOR_DIEPB

s4 <- ((augering$HOR_DIEPB <= d3) & (augering$HOR_DIEPO >= d4))
augering[s4,]$h_thick <- d4 - d3

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
augering$ratio <- (augering$h_thick/(d4-d3))

# The sum of ratios
augering$PROF_ID <- as.character(augering$PROFIEL_ID)
sum_ratio <- tapply(X = augering$ratio, INDEX = augering$PROF_ID, FUN = sum)
augering$sum_ratio <- sum_ratio[augering$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
augering <- subset(x = augering,subset = sum_ratio > 0.999)

# Bulk density - Augering

# Split the table into two dataset based on geocode value
# to apply two different aproaches

augering1 <- subset(x = augering, subset = ((GEO_FOR_C > 0) & (augering$GEO_FOR_C
     <= 690)))

augering2 <- subset(x = augering, subset = ((GEO_FOR_C > 690) |
     (augering$GEO_FOR_C == 0) | (is.na(augering$GEO_FOR_C))))
rm(augering)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")

top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
augering1$bulk <- NA
augering1$GEO_FOR_C <- as.character(augering1$GEO_FOR_C)
t <- (!is.na(augering1$A_HORIZON))
augering1[t,]$bulk <- top[augering1[t,]$GEO_FOR_C]
augering1[!t,]$bulk <- sub[augering1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
augering2$bulk <- NA
```

```
augering2 <- subset(augering2,!is.na(augering2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.4

s2 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.8

s4 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.65

s5 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 50) )
augering2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.5

s2 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.65

s4 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.6

s5 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 50))
augering2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
augering <- merge(augering1,augering2,all=TRUE)
rm(augering1,augering2)

# Profile Weighted averaging - Augering

augering$SoilMass <- augering$h_thick * augering$bulk
augering$SoilMass.ORGSTOF <- augering$SoilMass * augering$ORGSTOF

sum1 <- tapply(X = augering$SoilMass.ORGSTOF, INDEX = augering$PROF_ID, FUN = sum)
augering$sum1 <- sum1[augering$PROF_ID]

sum2 <- tapply(X = augering$SoilMass, INDEX = augering$PROF_ID, FUN = sum)
augering$sum2 <- sum2[augering$PROF_ID]

augering$wa_org <- augering$sum1/ augering$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
augering <- subset(x = augering, subset = duplicated(augering$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = augering$X, Y = augering$Y)
dup2 <- duplicated(dup)

augering$dupl_loc <- dup2
rm(dup,dup2)

augering <- subset(x = augering, subset = (dupl_loc==FALSE))

# All available organic matter data for the depth d3 - d4
augering <- data.frame(X = augering$X, Y = augering$Y, org = augering$wa_org)
```

```
# Pawn soil types - Augering

# read the pawn_map08.shp file
map <- readShapePoly("pawn-map08.shp",IDvar="PAWN_ID")

# overlay sampling locaations with pawn-map
coordinates(augering) <- ~X+Y
o <- overlay(map,augering)

augering$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_PFB_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_pfb_deltabis view
pfb <- sqlFetch(channel = channel, sqtable = "V_PFB_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$MON_DATUM) & pfb$MON_DATUM > from &
     pfb$MON_DATUM < to))

# Spatial extent filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$X) & pfb$X > X_MIN & pfb$X < X_MAX))
pfb <- subset(pfb,(!is.na(pfb$Y) & pfb$Y > Y_MIN & pfb$Y < Y_MAX))

# Bulk density - PFB

# Split the table into two dataset based on geocode value
# to apply two different aproaches

pfb1 <- subset(pfb,((GEO_FOR_C > 0) & (pfb$GEO_FOR_C <= 690)))

pfb2 <- subset(pfb,((GEO_FOR_C > 690) | (pfb$GEO_FOR_C == 0) |
     (is.na(pfb$GEO_FOR_C))))
rm(pfb)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")

top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
pfb1$bulk <- NA
pfb1$GEO_FOR_C <- as.character(pfb1$GEO_FOR_C)
t <- (!is.na(pfb1$A_HORIZON))
pfb1[t,]$bulk <- top[pfb1[t,]$GEO_FOR_C]
pfb1[!t,]$bulk <- sub[pfb1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
pfb2$bulk <- NA
pfb2 <- subset(pfb2,!is.na(pfb2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.4

s2 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.8

s4 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
```

```
pfb2[s4,]$bulk <- 0.65

s5 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50) )
pfb2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.5

s2 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.65

s4 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.6

s5 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50))
pfb2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
pfb <- merge(pfb1,pfb2,all=TRUE)
rm(pfb1,pfb2)

# Exclude samples collected from more than 1 soil horizon - PFB

isCrossing <- (!is.na(pfb$MON_VNR)  &
    ((pfb$MON_DIEPB < pfb$HOR_DIEPB) & (pfb$MON_DIEPO > pfb$HOR_DIEPB)) |
    ((pfb$MON_DIEPB < pfb$HOR_DIEPO) & (pfb$MON_DIEPO > pfb$HOR_DIEPO))
)

pfb <- pfb[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - PFB

# Weighted average of organic matter

# Subset removing NA records in ORGSTOF column
pfb <- subset(x = pfb,subset = !is.na(pfb$ORGSTOF))

# Weighted averiging of organic carbon content

pfb$s_thick <- pfb$MON_DIEPO - pfb$MON_DIEPB
pfb$s_thick.org <- pfb$s_thick * pfb$ORGSTOF

pfb$HOR_ID <- as.character(pfb$LAAG_ID)

sum1 <- tapply(X = pfb$s_thick.org, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_org <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Weighted average of bulk density

pfb$s_thick.bulk <- pfb$s_thick * pfb$bulk

sum1 <- tapply(X = pfb$s_thick.bulk, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_bulk <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
pfb <- subset(x = pfb,subset = duplicated(pfb$LAAG_ID)==FALSE)
```

```
# Horizons selection - PFB

pfb <- subset(x = pfb,subset =
    (HOR_DIEPB> d3 & HOR_DIEPO< d4) |
    (HOR_DIEPB< d3 & HOR_DIEPO> d3 & HOR_DIEPO< d4) |
    (HOR_DIEPB> d3 & HOR_DIEPB< d4 & HOR_DIEPO> d4) |
    (HOR_DIEPB<=d3 & HOR_DIEPO>=d4) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
pfb$h_thick <- 0

s1 <- ((pfb$HOR_DIEPB > d3) & (pfb$HOR_DIEPO < d4))
pfb[s1,]$h_thick <- pfb[s1,]$HOR_DIEPO - pfb[s1,]$HOR_DIEPB

s2 <- ((pfb$HOR_DIEPB < d3) & (pfb$HOR_DIEPO > d3) & (pfb$HOR_DIEPO < d4))
pfb[s2,]$h_thick <- pfb[s2,]$HOR_DIEPO - d3

s3 <- ((pfb$HOR_DIEPB > d3) & (pfb$HOR_DIEPB < d4) & (pfb$HOR_DIEPO > d4))
pfb[s3,]$h_thick <- d4 - pfb[s3,]$HOR_DIEPB

s4 <- ((pfb$HOR_DIEPB <= d3) & (pfb$HOR_DIEPO >= d4))
pfb[s4,]$h_thick <- d4 - d3

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
pfb$ratio <- (pfb$h_thick/(d4-d3))

# The sum of ratios
pfb$PROF_ID <- as.character(pfb$PROFIEL_ID)
sum_ratio <- tapply(X = pfb$ratio, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum_ratio <- sum_ratio[pfb$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
pfb <- subset(x = pfb, subset = sum_ratio > 0.999)

# Profile weighted averaging - PFB

pfb$SoilMass <- pfb$h_thick * pfb$l_bulk
pfb$SoilMass.l_org <- pfb$SoilMass * pfb$l_org

sum1 <- tapply(X = pfb$SoilMass.l_org, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$PROF_ID]

sum2 <- tapply(X = pfb$SoilMass, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$PROF_ID]

pfb$wa_org <- pfb$sum1/ pfb$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
pfb <- subset(x = pfb, subset = duplicated(pfb$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = pfb$X, Y = pfb$Y)
dup2 <- duplicated(dup)

pfb$dupl_loc <- dup2
rm(dup,dup2)

pfb <- subset(x = pfb, subset = (dupl_loc==FALSE))

# All available organic matter data for the depth d3 - d4
pfb <- data.frame(X = pfb$X, Y = pfb$Y, org = pfb$wa_org)


# Pawn soil types - PFB table

# overlay sampling locaations with pawn-map
coordinates(pfb) <- ~ X+Y
```

```
o <- overlay(map,pfb)

pfb$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_LSK_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_lsk_deltabis view
lsk <- sqlFetch(channel = channel, sqtable = "V_LSK_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$MON_DATUM) & lsk$MON_DATUM > from &
     lsk$MON_DATUM < to))

# Spatial extent filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$X) & lsk$X > X_MIN & lsk$X < X_MAX))
lsk <- subset(lsk,(!is.na(lsk$Y) & lsk$Y > Y_MIN & lsk$Y < Y_MAX))

# Bulk density - LSK

# Split the table into two dataset based on geocode value
# to apply two different aproaches

lsk1 <- subset(lsk,((GEO_FOR_C > 0) & (lsk$GEO_FOR_C <= 690)))

lsk2 <- subset(lsk,((GEO_FOR_C > 690) | (lsk$GEO_FOR_C == 0) |
     (is.na(lsk$GEO_FOR_C))))
rm(lsk)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")

top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
lsk1$bulk <- NA
lsk1$GEO_FOR_C <- as.character(lsk1$GEO_FOR_C)
t <- (!is.na(lsk1$A_HORIZON))
lsk1[t,]$bulk <- top[lsk1[t,]$GEO_FOR_C]
lsk1[!t,]$bulk <- sub[lsk1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
lsk2$bulk <- NA
lsk2 <- subset(lsk2,!is.na(lsk2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.4

s2 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.8

s4 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.65

s5 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50) )
lsk2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.5
```

```
s2 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.65

s4 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.6

s5 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50))
lsk2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
lsk <- merge(lsk1,lsk2,all=TRUE)
rm(lsk1,lsk2)

# Exclude samples collected from more than 1 soil horizon - LSK

isCrossing <- (!is.na(lsk$MON_VNR)  &
    ((lsk$MON_DIEPB < lsk$HOR_DIEPB) & (lsk$MON_DIEPO > lsk$HOR_DIEPB)) |
    ((lsk$MON_DIEPB < lsk$HOR_DIEPO) & (lsk$MON_DIEPO > lsk$HOR_DIEPO))
)

lsk <- lsk[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - LSK

# Subset removing NA records in ORGSTOF column
lsk <- subset(x = lsk,subset = !is.na(lsk$ORGSTOF))

# Weighted averiging

lsk$s_thick <- lsk$MON_DIEPO - lsk$MON_DIEPB
lsk$s_thick.org <- lsk$s_thick * lsk$ORGSTOF

lsk$HOR_ID <- as.character(lsk$LAAG_ID)

sum1 <- tapply(X = lsk$s_thick.org, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_org <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Weighted averaging of bulk density

lsk$s_thick.bulk <- lsk$s_thick * lsk$bulk

sum1 <- tapply(X = lsk$s_thick.bulk, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_bulk <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
lsk <- subset(x = lsk,subset = duplicated(lsk$LAAG_ID)==FALSE)

# Horizons selection - LSK

lsk <- subset(x = lsk,subset =
    (HOR_DIEPB> d3 & HOR_DIEPO< d4) |
    (HOR_DIEPB< d3 & HOR_DIEPO> d3 & HOR_DIEPO< d4) |
    (HOR_DIEPB> d3 & HOR_DIEPB< d4 & HOR_DIEPO> d4) |
    (HOR_DIEPB<=d3 & HOR_DIEPO>=d4) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
```

```
lsk$h_thick <- 0

s1 <- ((lsk$HOR_DIEPB > d3) & (lsk$HOR_DIEPO < d4))
lsk[s1,]$h_thick <- lsk[s1,]$HOR_DIEPO - lsk[s1,]$HOR_DIEPB

s2 <- ((lsk$HOR_DIEPB < d3) & (lsk$HOR_DIEPO > d3) & (lsk$HOR_DIEPO < d4))
lsk[s2,]$h_thick <- lsk[s2,]$HOR_DIEPO - d3

s3 <- ((lsk$HOR_DIEPB > d3) & (lsk$HOR_DIEPB < d4) & (lsk$HOR_DIEPO > d4))
lsk[s3,]$h_thick <- d4 - lsk[s3,]$HOR_DIEPB

s4 <- ((lsk$HOR_DIEPB <= d3) & (lsk$HOR_DIEPO >= d4))
lsk[s4,]$h_thick <- d4 - d3

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
lsk$ratio <- (lsk$h_thick/(d4-d3))

# The sum of ratios
lsk$PROF_ID <- as.character(lsk$PROFIEL_ID)
sum_ratio <- tapply(X = lsk$ratio, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum_ratio <- sum_ratio[lsk$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
lsk <- subset(x = lsk,subset = sum_ratio > 0.999)

# Profile weighted averaging - LSK

lsk$SoilMass <- lsk$h_thick * lsk$l_bulk
lsk$SoilMass.l_org <- lsk$SoilMass * lsk$l_org

sum1 <- tapply(X = lsk$SoilMass.l_org, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$PROF_ID]

sum2 <- tapply(X = lsk$SoilMass, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$PROF_ID]

lsk$wa_org <- lsk$sum1/ lsk$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
lsk <- subset(x = lsk, subset = duplicated(lsk$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = lsk$X, Y = lsk$Y)
dup2 <- duplicated(dup)

lsk$dupl_loc <- dup2
rm(dup,dup2)

lsk <- subset(x = lsk, subset = (dupl_loc==FALSE))

# Available organic matter data for the depth d3 - d4
lsk <- data.frame(X = lsk$X, Y = lsk$Y, org = lsk$wa_org)


# Pawn soil types - LSK

# Overlay sampling locations with pawn map
coordinates(lsk) <- ~ X+Y
o <- overlay(map,lsk)

lsk$pawn <- o$PAWN_CODE
rm(o)

# Merge tables

org <- merge(pfb,lsk,all=TRUE)
org <- merge(org,augering,all=TRUE)
rm(pfb,lsk,augering)
```

```
# Exlude locations which fall into the water,build up areas, and NA
org <- subset(x = org, subset = (pawn < 22) & (!is.na(pawn)))

# Replace organic matter where 0 with 0.2
s <- (org$org==0)
org[s,]$org <- 0.2
rm(s)

# Rename table
org2 <- org
rm(org)

# Write table to the text file
write.table(org2,"org2.txt",sep="\t",row.names=FALSE)

# 26 Save Box-plots
png("box_plot2.png",width=800,height=600,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
boxplot(split(org2$org,org2$pawn),xlab="Pawn soil type",main="Organic matter",
     cex.main=2)
dev.off()


# 27 Histograms

# histogram of original dataset
png("hist2.png",width=400,height=400,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(org2$org,main="",xlab="Organic matter",col="lightblue")
dev.off()
```

E.3 Third layer data extraction and preparation

```
# Third layer data extraction and preparation

############ Set parameters ##############

# The thickness of the second soil layer (cm)
d5 <- 50
d6 <- 60

##########################################

# View extraction - V_Augering_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_augering_deltabis view
augering <- sqlFetch(channel = channel, sqtable = "V_AUGERING_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - Augering
augering$MON_DATUM <- paste(augering$MON_DATUM,"-01-01",sep="")
augering$MON_DATUM <- as.POSIXct(augering$MON_DATUM)

augering <- subset(augering,(!is.na(augering$MON_DATUM) & augering$MON_DATUM > from &
     augering$MON_DATUM < to))

# Spatial extent filtering - Augering
augering <- subset(augering,(!is.na(augering$X) & augering$X > X_MIN & augering$X < X_MAX))
augering <- subset(augering,(!is.na(augering$Y) & augering$Y > Y_MIN & augering$Y < Y_MAX))

# Horizons selection - Augering

# Exclude records with missing values
augering <- subset(x = augering, subset = (!is.na(ORGSTOF) & ORGSTOF > 0))
```

```
# Horizons selection
augering <- subset(x = augering,subset =
    (HOR_DIEPB > d5 & HOR_DIEPO < d6) |
    (HOR_DIEPB < d5 & HOR_DIEPO > d5 & HOR_DIEPO < d6) |
    (HOR_DIEPB > d5 & HOR_DIEPB < d6 & HOR_DIEPO > d6) |
    (HOR_DIEPB <= d5 & HOR_DIEPO >= d6) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
augering$h_thick <- 0

s1 <- ((augering$HOR_DIEPB > d5) & (augering$HOR_DIEPO < d6))
augering[s1,]$h_thick <- augering[s1,]$HOR_DIEPO - augering[s1,]$HOR_DIEPB

s2 <- ((augering$HOR_DIEPB < d5) & (augering$HOR_DIEPO > d5) & (augering$HOR_DIEPO < d6))
augering[s2,]$h_thick <- augering[s2,]$HOR_DIEPO - d5

s3 <- ((augering$HOR_DIEPB > d5) & (augering$HOR_DIEPB < d6) & (augering$HOR_DIEPO > d6))
augering[s3,]$h_thick <- d6 - augering[s3,]$HOR_DIEPB

s4 <- ((augering$HOR_DIEPB <= d5) & (augering$HOR_DIEPO >= d6))
augering[s4,]$h_thick <- d6 - d5

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
augering$ratio <- (augering$h_thick/(d6-d5))

# The sum of ratios
augering$PROF_ID <- as.character(augering$PROFIEL_ID)
sum_ratio <- tapply(X = augering$ratio, INDEX = augering$PROF_ID, FUN = sum)
augering$sum_ratio <- sum_ratio[augering$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
augering <- subset(x = augering,subset = sum_ratio > 0.999)

# Bulk density - Augering

# Split the table into two dataset based on geocode value
# to apply two different aproaches

augering1 <- subset(x = augering, subset = ((GEO_FOR_C > 0) &
     (augering$GEO_FOR_C <= 690)))

augering2 <- subset(x = augering, subset = ((GEO_FOR_C > 690) |
     (augering$GEO_FOR_C == 0) | (is.na(augering$GEO_FOR_C))))
rm(augering)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")

top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
augering1$bulk <- NA
augering1$GEO_FOR_C <- as.character(augering1$GEO_FOR_C)
t <- (!is.na(augering1$A_HORIZON))
augering1[t,]$bulk <- top[augering1[t,]$GEO_FOR_C]
augering1[!t,]$bulk <- sub[augering1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
augering2$bulk <- NA
augering2 <- subset(augering2,!is.na(augering2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.4

s2 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
```

```
augering2[s2,]$bulk <- 1.15

s3 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.8

s4 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.65

s5 <- ((!is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 50) )
augering2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF < 8))
augering2[s1,]$bulk <- 1.5

s2 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 8) &
     (augering2$ORGSTOF < 17.5))
augering2[s2,]$bulk <- 1.15

s3 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 17.5) &
     (augering2$ORGSTOF < 30))
augering2[s3,]$bulk <- 0.65

s4 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 30) &
     (augering2$ORGSTOF < 50))
augering2[s4,]$bulk <- 0.6

s5 <- ((is.na(augering2$A_HORIZON)) & (augering2$ORGSTOF >= 50))
augering2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
augering <- merge(augering1,augering2,all=TRUE)
rm(augering1,augering2)

# Profile Weighted averaging - Augering

augering$SoilMass <- augering$h_thick * augering$bulk
augering$SoilMass.ORGSTOF <- augering$SoilMass * augering$ORGSTOF

sum1 <- tapply(X = augering$SoilMass.ORGSTOF, INDEX = augering$PROF_ID, FUN = sum)
augering$sum1 <- sum1[augering$PROF_ID]

sum2 <- tapply(X = augering$SoilMass, INDEX = augering$PROF_ID, FUN = sum)
augering$sum2 <- sum2[augering$PROF_ID]

augering$wa_org <- augering$sum1/ augering$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
augering <- subset(x = augering, subset = duplicated(augering$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = augering$X, Y = augering$Y)
dup2 <- duplicated(dup)

augering$dupl_loc <- dup2
rm(dup,dup2)

augering <- subset(x = augering, subset = (dupl_loc==FALSE))

# All available organic matter data for the depth d5 - d6
augering <- data.frame(X = augering$X, Y = augering$Y, org = augering$wa_org)


# Pawn soil types - Augering

# read the pawn_map08.shp file
map <- readShapePoly("pawn-map08.shp",IDvar="PAWN_ID")

# overlay sampling locaations with pawn-map
```

```
coordinates(augering) <- ~X+Y
o <- overlay(map,augering)

augering$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_PFB_DELTABIS

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_pfb_deltabis view
pfb <- sqlFetch(channel = channel, sqtable = "V_PFB_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$MON_DATUM) & pfb$MON_DATUM > from &
     pfb$MON_DATUM < to))

# Spatial extent filtering - PFB
pfb <- subset(pfb,(!is.na(pfb$X) & pfb$X > X_MIN & pfb$X < X_MAX))
pfb <- subset(pfb,(!is.na(pfb$Y) & pfb$Y > Y_MIN & pfb$Y < Y_MAX))

# Bulk density - PFB

# Split the table into two dataset based on geocode value
# to apply two different aproaches

pfb1 <- subset(pfb,((GEO_FOR_C > 0) & (pfb$GEO_FOR_C <= 690)))

pfb2 <- subset(pfb,((GEO_FOR_C > 690) | (pfb$GEO_FOR_C == 0) |
     (is.na(pfb$GEO_FOR_C))))
rm(pfb)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")

top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
pfb1$bulk <- NA
pfb1$GEO_FOR_C <- as.character(pfb1$GEO_FOR_C)
t <- (!is.na(pfb1$A_HORIZON))
pfb1[t,]$bulk <- top[pfb1[t,]$GEO_FOR_C]
pfb1[!t,]$bulk <- sub[pfb1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
pfb2$bulk <- NA
pfb2 <- subset(pfb2,!is.na(pfb2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.4

s2 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.8

s4 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.65

s5 <- ((!is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50) )
pfb2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF < 8))
pfb2[s1,]$bulk <- 1.5
```

```
s2 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 8) & (pfb2$ORGSTOF < 17.5))
pfb2[s2,]$bulk <- 1.15

s3 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 17.5) & (pfb2$ORGSTOF < 30))
pfb2[s3,]$bulk <- 0.65

s4 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 30) & (pfb2$ORGSTOF < 50))
pfb2[s4,]$bulk <- 0.6

s5 <- ((is.na(pfb2$A_HORIZON)) & (pfb2$ORGSTOF >= 50))
pfb2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
pfb <- merge(pfb1,pfb2,all=TRUE)
rm(pfb1,pfb2)

# Exclude samples collected from more than 1 soil horizon - PFB

isCrossing <- (!is.na(pfb$MON_VNR)  &
    ((pfb$MON_DIEPB < pfb$HOR_DIEPB) & (pfb$MON_DIEPO > pfb$HOR_DIEPB)) |
    ((pfb$MON_DIEPB < pfb$HOR_DIEPO) & (pfb$MON_DIEPO > pfb$HOR_DIEPO))
)

pfb <- pfb[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - PFB

# Weighted average of organic matter

# Subset removing NA records in ORGSTOF column
pfb <- subset(x = pfb,subset = !is.na(pfb$ORGSTOF))

# Weighted averiging of organic carbon content

pfb$s_thick <- pfb$MON_DIEPO - pfb$MON_DIEPB
pfb$s_thick.org <- pfb$s_thick * pfb$ORGSTOF

pfb$HOR_ID <- as.character(pfb$LAAG_ID)

sum1 <- tapply(X = pfb$s_thick.org, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_org <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Weighted average of bulk density

pfb$s_thick.bulk <- pfb$s_thick * pfb$bulk

sum1 <- tapply(X = pfb$s_thick.bulk, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$HOR_ID]

sum2 <- tapply(X = pfb$s_thick, INDEX = pfb$HOR_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$HOR_ID]

pfb$l_bulk <- (pfb$sum1/pfb$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
pfb <- subset(x = pfb,subset = duplicated(pfb$LAAG_ID)==FALSE)

# Horizons selection - PFB

pfb <- subset(x = pfb,subset =
    (HOR_DIEPB> d5 & HOR_DIEPO< d6) |
    (HOR_DIEPB< d5 & HOR_DIEPO> d5 & HOR_DIEPO< d6) |
    (HOR_DIEPB> d5 & HOR_DIEPB< d6 & HOR_DIEPO> d6) |
    (HOR_DIEPB<=d5 & HOR_DIEPO>=d6) )
```

```
# Computing the thickness of a part of horizon which contribute to the required
# soil layer
pfb$h_thick <- 0

s1 <- ((pfb$HOR_DIEPB > d5) & (pfb$HOR_DIEPO < d6))
pfb[s1,]$h_thick <- pfb[s1,]$HOR_DIEPO - pfb[s1,]$HOR_DIEPB

s2 <- ((pfb$HOR_DIEPB < d5) & (pfb$HOR_DIEPO > d5) & (pfb$HOR_DIEPO < d6))
pfb[s2,]$h_thick <- pfb[s2,]$HOR_DIEPO - d5

s3 <- ((pfb$HOR_DIEPB > d5) & (pfb$HOR_DIEPB < d6) & (pfb$HOR_DIEPO > d6))
pfb[s3,]$h_thick <- d6 - pfb[s3,]$HOR_DIEPB

s4 <- ((pfb$HOR_DIEPB <= d5) & (pfb$HOR_DIEPO >= d6))
pfb[s4,]$h_thick <- d6 - d5

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
pfb$ratio <- (pfb$h_thick/(d6-d5))

# The sum of ratios
pfb$PROF_ID <- as.character(pfb$PROFIEL_ID)
sum_ratio <- tapply(X = pfb$ratio, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum_ratio <- sum_ratio[pfb$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
pfb <- subset(x = pfb, subset = sum_ratio > 0.999)

# Profile weighted averaging - PFB

pfb$SoilMass <- pfb$h_thick * pfb$l_bulk
pfb$SoilMass.l_org <- pfb$SoilMass * pfb$l_org

sum1 <- tapply(X = pfb$SoilMass.l_org, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum1 <- sum1[pfb$PROF_ID]

sum2 <- tapply(X = pfb$SoilMass, INDEX = pfb$PROF_ID, FUN = sum)
pfb$sum2 <- sum2[pfb$PROF_ID]

pfb$wa_org <- pfb$sum1/ pfb$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
pfb <- subset(x = pfb, subset = duplicated(pfb$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = pfb$X, Y = pfb$Y)
dup2 <- duplicated(dup)

pfb$dupl_loc <- dup2
rm(dup,dup2)

pfb <- subset(x = pfb, subset = (dupl_loc==FALSE))

# All available organic matter data for the depth d5 - d6
pfb <- data.frame(X = pfb$X, Y = pfb$Y, org = pfb$wa_org)


# Pawn soil types - PFB table

# overlay sampling locaations with pawn-map
coordinates(pfb) <- ~ X+Y
o <- overlay(map,pfb)

pfb$pawn <- o$PAWN_CODE
rm(o)

# View extraction - V_LSK_DELTABIS

# connect to database
```

```
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# fetch all records from V_lsk_deltabis view
lsk <- sqlFetch(channel = channel, sqtable = "V_LSK_DELTABIS")

# disconnect from database
odbcClose(channel = channel)
rm(channel)

# Time filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$MON_DATUM) & lsk$MON_DATUM > from &
     lsk$MON_DATUM < to))

# Spatial extent filtering - LSK
lsk <- subset(lsk,(!is.na(lsk$X) & lsk$X > X_MIN & lsk$X < X_MAX))
lsk <- subset(lsk,(!is.na(lsk$Y) & lsk$Y > Y_MIN & lsk$Y < Y_MAX))

# Bulk density - LSK

# Split the table into two dataset based on geocode value
# to apply two different aproaches

lsk1 <- subset(lsk,((GEO_FOR_C > 0) & (lsk$GEO_FOR_C <= 690)))

lsk2 <- subset(lsk,((GEO_FOR_C > 690) | (lsk$GEO_FOR_C == 0) |
     (is.na(lsk$GEO_FOR_C))))
rm(lsk)

# Read table with bulk density values
BD <- read.table("BD_table.txt",header=T,sep="\t")

top <- array(data=BD$top,dim=length(BD$top),dimnames=list(as.character(BD$geocode)))
sub <- array(data=BD$sub,dim=length(BD$sub),dimnames=list(as.character(BD$geocode)))

# The bulk density for the first subset
lsk1$bulk <- NA
lsk1$GEO_FOR_C <- as.character(lsk1$GEO_FOR_C)
t <- (!is.na(lsk1$A_HORIZON))
lsk1[t,]$bulk <- top[lsk1[t,]$GEO_FOR_C]
lsk1[!t,]$bulk <- sub[lsk1[!t,]$GEO_FOR_C]

rm(top,sub,t,BD)

# The bulk density for the second subset
lsk2$bulk <- NA
lsk2 <- subset(lsk2,!is.na(lsk2$ORGSTOF))

# Top horizon
s1 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.4

s2 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.8

s4 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.65

s5 <- ((!is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50) )
lsk2[s5,]$bulk <- 0.5

# Sub horizon
s1 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF < 8))
lsk2[s1,]$bulk <- 1.5

s2 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 8) & (lsk2$ORGSTOF < 17.5))
lsk2[s2,]$bulk <- 1.15

s3 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 17.5) & (lsk2$ORGSTOF < 30))
lsk2[s3,]$bulk <- 0.65

s4 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 30) & (lsk2$ORGSTOF < 50))
lsk2[s4,]$bulk <- 0.6
```

```
s5 <- ((is.na(lsk2$A_HORIZON)) & (lsk2$ORGSTOF >= 50))
lsk2[s5,]$bulk <- 0.35

rm(s1,s2,s3,s4,s5)

# Merge tables
lsk <- merge(lsk1,lsk2,all=TRUE)
rm(lsk1,lsk2)

# Exclude samples collected from more than 1 soil horizon - LSK

isCrossing <- (!is.na(lsk$MON_VNR)  &
    ((lsk$MON_DIEPB < lsk$HOR_DIEPB) & (lsk$MON_DIEPO > lsk$HOR_DIEPB)) |
    ((lsk$MON_DIEPB < lsk$HOR_DIEPO) & (lsk$MON_DIEPO > lsk$HOR_DIEPO))
)

lsk <- lsk[!isCrossing, ]
rm(isCrossing)

# Horizon weighted averaging - LSK

# Subset removing NA records in ORGSTOF column
lsk <- subset(x = lsk,subset = !is.na(lsk$ORGSTOF))

# Weighted averiging

lsk$s_thick <- lsk$MON_DIEPO - lsk$MON_DIEPB
lsk$s_thick.org <- lsk$s_thick * lsk$ORGSTOF

lsk$HOR_ID <- as.character(lsk$LAAG_ID)

sum1 <- tapply(X = lsk$s_thick.org, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_org <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Weighted averaging of bulk density

lsk$s_thick.bulk <- lsk$s_thick * lsk$bulk

sum1 <- tapply(X = lsk$s_thick.bulk, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$HOR_ID]

sum2 <- tapply(X = lsk$s_thick, INDEX = lsk$HOR_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$HOR_ID]

lsk$l_bulk <- (lsk$sum1/lsk$sum2)
rm(sum1,sum2)

# Remove duplicated layers - no more needed
lsk <- subset(x = lsk,subset = duplicated(lsk$LAAG_ID)==FALSE)

# Horizons selection - LSK

lsk <- subset(x = lsk,subset =
    (HOR_DIEPB> d5 & HOR_DIEPO< d6) |
    (HOR_DIEPB< d5 & HOR_DIEPO> d5 & HOR_DIEPO< d6) |
    (HOR_DIEPB> d5 & HOR_DIEPB< d6 & HOR_DIEPO> d6) |
    (HOR_DIEPB<=d5 & HOR_DIEPO>=d6) )

# Computing the thickness of a part of horizon which contribute to the required
# soil layer
lsk$h_thick <- 0

s1 <- ((lsk$HOR_DIEPB > d5) & (lsk$HOR_DIEPO < d6))
lsk[s1,]$h_thick <- lsk[s1,]$HOR_DIEPO - lsk[s1,]$HOR_DIEPB

s2 <- ((lsk$HOR_DIEPB < d5) & (lsk$HOR_DIEPO > d5) & (lsk$HOR_DIEPO < d6))
lsk[s2,]$h_thick <- lsk[s2,]$HOR_DIEPO - d5
```

```
s3 <- ((lsk$HOR_DIEPB > d5) & (lsk$HOR_DIEPB < d6) & (lsk$HOR_DIEPO > d6))
lsk[s3,]$h_thick <- d6 - lsk[s3,]$HOR_DIEPB

s4 <- ((lsk$HOR_DIEPB <= d5) & (lsk$HOR_DIEPO >= d6))
lsk[s4,]$h_thick <- d6 - d5

rm(s1,s2,s3,s4)

# The ratio of soil horizon length to the entire required soil layer
lsk$ratio <- (lsk$h_thick/(d6-d5))

# The sum of ratios
lsk$PROF_ID <- as.character(lsk$PROFIEL_ID)
sum_ratio <- tapply(X = lsk$ratio, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum_ratio <- sum_ratio[lsk$PROF_ID]

rm(sum_ratio)

# Excluding soil profiles with uncomplete information
lsk <- subset(x = lsk,subset = sum_ratio > 0.999)

# Profile weighted averaging - LSK

lsk$SoilMass <- lsk$h_thick * lsk$l_bulk
lsk$SoilMass.l_org <- lsk$SoilMass * lsk$l_org

sum1 <- tapply(X = lsk$SoilMass.l_org, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum1 <- sum1[lsk$PROF_ID]

sum2 <- tapply(X = lsk$SoilMass, INDEX = lsk$PROF_ID, FUN = sum)
lsk$sum2 <- sum2[lsk$PROF_ID]

lsk$wa_org <- lsk$sum1/ lsk$sum2
rm(sum1,sum2)

# Subset with unique soil profiles
lsk <- subset(x = lsk, subset = duplicated(lsk$PROFIEL_ID)==FALSE)

# Find and exclude records with the same coordinates

dup <- data.frame(X = lsk$X, Y = lsk$Y)
dup2 <- duplicated(dup)

lsk$dupl_loc <- dup2
rm(dup,dup2)

lsk <- subset(x = lsk, subset = (dupl_loc==FALSE))

# Available organic matter data for the depth d5 - d6
lsk <- data.frame(X = lsk$X, Y = lsk$Y, org = lsk$wa_org)


# Pawn soil types - LSK

# Overlay sampling locations with pawn map
coordinates(lsk) <- ~ X+Y
o <- overlay(map,lsk)

lsk$pawn <- o$PAWN_CODE
rm(o)

# Merge tables

org <- merge(pfb,lsk,all=TRUE)
org <- merge(org,augering,all=TRUE)
rm(pfb,lsk,augering)

# Exlude locations which fall into the water,build up areas, and NA
org <- subset(x = org, subset = (pawn < 22) & (!is.na(pawn)))

# Replace organic matter where 0 with 0.2
s <- (org$org==0)
org[s,]$org <- 0.2
rm(s)
```

```
# Rename table
org3 <- org
rm(org)

# Write table to the text file
write.table(org3,"org3.txt",sep="\t",row.names=FALSE)

# 26 Save Box-plots
png("box_plot3.png",width=800,height=600,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
boxplot(split(org3$org,org3$pawn),xlab="Pawn soil type",main="Organic matter",
     cex.main=2)
dev.off()


# 27 Histograms

# histogram of original dataset
png("hist3.png",width=400,height=400,units="px",pointsize=12,bg="white",res=NA)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(org3$org,main="",xlab="Organic matter",col="lightblue")
dev.off()
```

E.4 Cross-variography

```
# ---- Organic matter multi layer spatial analysis ---- #

# Requires three tables of three different soil layers
org1 <- read.table("org1.txt",sep="\t",header=TRUE,na.strings="NA")
org2 <- read.table("org2.txt",sep="\t",header=TRUE,na.strings="NA")
org3 <- read.table("org3.txt",sep="\t",header=TRUE,na.strings="NA")

# Log transformation
org1$log_org <- log(org1$org+1)
org2$log_org <- log(org2$org+1)
org3$log_org <- log(org3$org+1)

# Residuals

mean1 <- tapply(org1$log_org,org1$pawn,mean)
org1$mean <- mean1[org1$pawn]

sd1 <- tapply(org1$log_org,org1$pawn,sd)
org1$sd <- sd1[org1$pawn]

org1$residua <- (org1$log_org-org1$mean)/org1$sd

mean2 <- tapply(org2$log_org,org2$pawn,mean)
org2$mean <- mean2[org2$pawn]

sd2 <- tapply(org2$log_org,org2$pawn,sd)
org2$sd <- sd2[org2$pawn]

org2$residua <- (org2$log_org-org2$mean)/org2$sd

mean3 <- tapply(org3$log_org,org3$pawn,mean)
org3$mean <- mean3[org3$pawn]

sd3 <- tapply(org3$log_org,org3$pawn,sd)
org3$sd <- sd3[org3$pawn]

org3$residua <- (org3$log_org-org3$mean)/org3$sd

# Random pick to speed up variogram estimation
org11 <- org1[sample(nrow(org1),size=5000,replace=F),]
org22 <- org2[sample(nrow(org2),size=5000,replace=F),]
org33 <- org3[sample(nrow(org3),size=5000,replace=F),]

# Define the gstat object and LMC fitting

gORG <- gstat(NULL ,id="org.1", formula=residua ~ 1, locations= ~X+Y, data=org11)
```

```
gORG <- gstat(gORG, id="org.2", formula=residua ~ 1, locations= ~X+Y, data=org22)
gORG <- gstat(gORG, id="org.3", formula=residua ~ 1, locations= ~X+Y, data=org33)

gORG <- gstat(gORG, model=vgm(psill= 0.2,model="Exp",range=40000,nugget=0.8),
     fill.all=TRUE)

v <- variogram(gORG,boundaries=c(15000,30000,50000,70000,90000,110000,130000,
     150000))

fit <- fit.lmc(v, gORG)

plot(v)
plot(v,fit)

# Save variogram figure
png("lmc.png",width=600,height=600,units="px",pointsize=12,bg="white",res=NA)
fontsize <- trellis.par.get("fontsize")
fontsize$text <- 16
fontsize$points <- 16
trellis.par.set("fontsize",fontsize)
plot(v,fit,main="Organic matter - LMC",pch=20)
dev.off()
```

E.5 Define interpolation grid

```
# Define interpolation grid - chose one option only

# 1) rectangular grid

######### Set parameters ##########

# Output map extent
x.min <- 10000
x.max <- 280000
y.min <- 300000
y.max <- 620000

# Prediction grid cell size (output map resolution)
cs <- 1000

###################################

x <- seq(x.min,x.max,cs)
y <- seq(y.min,y.max,cs)
grid <- expand.grid(x1=x,x2=y)
coordinates(grid) <- ~x1+x2
o <- overlay(map,grid)
grid <- data.frame(grid,pawn=o$PAWN_CODE)
grid <- subset(grid,(pawn < 22 & !is.na(pawn)))
coordinates(grid) <- ~x1+x2
rm(o)


# 2) sampling within polygon(s)

######### Set parameters ##########

# Prediction grid cell size (output map resolution)
cs <- 1000

###################################

# Read ESRI shapefile

# the whole NL boundary (en example)
nl_poly <- readShapePoly("NL.shp")

grid <- spsample(nl_poly,cellsize=cs,type="regular")
o <- overlay(map,grid)
grid <- data.frame(grid,pawn=o$PAWN_CODE)
grid <- subset(grid,(pawn < 22 & !is.na(pawn)))
```

```
coordinates(grid)<- ~x1+x2
rm(o)
```

E.6 Co-kriging prediction

```
# Co-kriging prediction

############ Set parameters ############

# Point or block kriging
# "block <- 0" ... point kriging
# "block <- 1" ... block kriging
block <- 0

# Block size (for block kriging only)
bs <- c(10,10)

########################################

# Redefine the gstat object and LMC fit
gORG <- gstat(NULL ,id="org.1", formula=residua ~ 1, locations= ~X+Y,
     data=org1,nmax=30,beta=0)
gORG <- gstat(gORG, id="org.2", formula=residua ~ 1, locations= ~X+Y,
     data=org2,nmax=30,beta=0)
gORG <- gstat(gORG, id="org.3", formula=residua ~ 1, locations= ~X+Y,
     data=org3,nmax=30,beta=0)
gORG <- gstat(gORG, model=vgm(psill= 0.2,model="Exp",range=40000,nugget=0.8),
     fill.all=TRUE)

fit <- fit.lmc(v, gORG)

# Cokriging
if(block==0)    blup2 <- predict.gstat(fit,grid) else
            blup2 <- predict.gstat(fit,grid,block=bs)

# Prediction for the first layer
org1.pred <- data.frame(coordinates(blup2),residua=blup2[[1]],pawn=grid$pawn)

org1.pred$mean <- mean1[org1.pred$pawn]

org1.pred$sd <- sd1[org1.pred$pawn]

org1.pred$pred <- org1.pred$residua * org1.pred$sd + org1.pred$mean

# Back transformation
org1.pred$pred <- exp(org1.pred$pred) - 1

coordinates(org1.pred) <- ~x1+x2
gridded(org1.pred) <- TRUE

# Prediction for the second layer
org2.pred <- data.frame(coordinates(blup2),residua=blup2[[3]],pawn=grid$pawn)

org2.pred$mean <- mean2[org2.pred$pawn]

org2.pred$sd <- sd2[org2.pred$pawn]

org2.pred$pred <- org2.pred$residua * org2.pred$sd + org2.pred$mean

# Back transformation
org2.pred$pred <- exp(org2.pred$pred) - 1

coordinates(org2.pred) <- ~x1+x2
gridded(org2.pred) <- TRUE

# Prediction for the third layer
org3.pred <- data.frame(coordinates(blup2),residua=blup2[[5]],pawn=grid$pawn)

org3.pred$mean <- mean3[org3.pred$pawn]

org3.pred$sd <- sd3[org3.pred$pawn]
```

```
org3.pred$pred <- org3.pred$residua * org3.pred$sd + org3.pred$mean

# Back transformation
org3.pred$pred <- exp(org3.pred$pred) - 1

coordinates(org3.pred) <- ~x1+x2
gridded(org3.pred) <- TRUE

# Save figures

nl_poly <- readShapePoly("NL.shp")
text <- list("sp.text",c(60000,550000),paste(d1,"-",d2,"cm"),cex=1.5)
nl <- list("sp.polygons", nl_poly)
rgb.palette <- colorRampPalette(c("lightyellow","green","orange","brown"),
     space = "rgb")

png(filename="org1_copred.png",width=600,height=600,units="px",pointsize=12,
     bg="white",res=NA)
spplot(org1.pred["pred"],main=list(label="Organic matter - cokriging prediction",
     cex=1.5),
col.regions=rgb.palette(22),
xlab=list(paste(d1,"-",d2,"cm"),cex=1.5),
at=c(seq(0,10,0.5),120),
colorkey=list(space="right",width=2,at=1:22,labels=list(cex=2,at=1:22,
labels=c("","","1","","","","3","","","","5","","","","7","","","","9 >","",
     "",""))),
sp.layout=list(nl,text))
dev.off()

text <- list("sp.text",c(60000,550000),paste(d3,"-",d4,"cm"),cex=1.5)

png(filename="org2_copred.png",width=600,height=600,units="px",pointsize=12,
     bg="white",res=NA)
spplot(org2.pred["pred"],main=list(label="Organic matter - cokriging prediction",
     cex=1.5),
col.regions=rgb.palette(22),
xlab=list(paste(d3,"-",d4,"cm"),cex=1.5),
at=c(seq(0,10,0.5),120),
colorkey=list(space="right",width=2,at=1:22,labels=list(cex=2,at=1:22,
labels=c("","","1","","","","3","","","","5","","","","7","","","","9 >",
     "","","")),
sp.layout=list(nl,text))
dev.off()

text <- list("sp.text",c(60000,550000),paste(d5,"-",d6,"cm"),cex=1.5)

png(filename="org3_copred.png",width=600,height=600,units="px",pointsize=12,
     bg="white",res=NA)
spplot(org3.pred["pred"],main=list(label="Organic matter - cokriging prediction",
     cex=1.5),
col.regions=rgb.palette(22),
xlab=list(paste(d5,"-",d6,"cm"),cex=1.5),
at=c(seq(0,10,0.5),120),
colorkey=list(space="right",width=2,at=1:22,labels=list(cex=2,at=1:22,
labels=c("","","1","","","","3","","","","5","","","","7","","","","9 >",
     "","","")),
sp.layout=list(nl,text))
dev.off()
```

E.7 Stochastic co-simulation (based on simple cokriging)

```
# Stochastic co-simulation (based on simple cokriging)

############ Set parameters ###############

# Number of simulations
ns <- 2
```

```
# Point or block kriging
# "block <- 0" ... point kriging
# "block <- 1" ... block kriging
block <- 0

# Block size (for block kriging only)
bs <- c(10,10)

########################################

# Redefine the gstat object and LMC fit
gORG <- gstat(NULL ,id="org.1", formula=residua ~ 1, locations= ~X+Y,
     data=org1,nmax=30,beta=0)
gORG <- gstat(gORG, id="org.2", formula=residua ~ 1, locations= ~X+Y,
     data=org2,nmax=30,beta=0)
gORG <- gstat(gORG, id="org.3", formula=residua ~ 1, locations= ~X+Y,
     data=org3,nmax=30,beta=0)
gORG <- gstat(gORG, model=vgm(psill= 0.2,model="Exp",range=40000,nugget=0.8),
     fill.all=TRUE)

fit <- fit.lmc(v, gORG)

# Run stochastic simulations
if(block==0)    blup <- predict.gstat(fit,grid,nsim=ns) else
            blup <- predict.gstat(fit,grid,nsim=ns,block=bs)

# Simulation for the first layer
org1.sim <- data.frame(blup[,1:ns],pawn=grid$pawn)
org1.sim$mean <- mean1[org1.sim$pawn]
org1.sim$sd <- sd1[org1.sim$pawn]

org1.sim_ns <- data.frame(coordinates(blup))
org1.sim_ns[,3:(ns+2)] <- org1.sim[,3:(ns+2)] * org1.sim$sd + org1.sim$mean

# Back transformation
org1.sim_ns[,3:(ns+2)] <- exp(org1.sim_ns[,3:(ns+2)]) - 1

coordinates(org1.sim_ns) <- ~x1+x2
gridded(org1.sim_ns) <- TRUE

# Simulation for the second layer
org2.sim <- data.frame(blup[,(ns+1):(ns*2)],pawn=grid$pawn)
org2.sim$mean <- mean2[org2.sim$pawn]
org2.sim$sd <- sd2[org2.sim$pawn]

org2.sim_ns <- data.frame(coordinates(blup))
org2.sim_ns[,3:(ns+2)] <- org2.sim[,3:(ns+2)] * org2.sim$sd + org2.sim$mean

# Back transformation
org2.sim_ns[,3:(ns+2)] <- exp(org2.sim_ns[,3:(ns+2)]) - 1

coordinates(org2.sim_ns) <- ~x1+x2
gridded(org2.sim_ns) <- TRUE

# Simulation for the third layer
org3.sim <- data.frame(blup[,(ns*2+1):(ns*3)],pawn=grid$pawn)
org3.sim$mean <- mean3[org3.sim$pawn]
org3.sim$sd <- sd3[org3.sim$pawn]

org3.sim_ns <- data.frame(coordinates(blup))
org3.sim_ns[,3:(ns+2)] <- org3.sim[,3:(ns+2)] * org3.sim$sd + org3.sim$mean

# Back transformation
org3.sim_ns[,3:(ns+2)] <- exp(org3.sim_ns[,3:(ns+2)]) - 1

coordinates(org3.sim_ns) <- ~x1+x2
gridded(org3.sim_ns) <- TRUE


# Save figures for first layer

nl_poly <- readShapePoly("NL.shp")
text <- list("sp.text",c(60000,550000),paste(d1,"-",d2,"cm"),cex=1.5)
nl <- list("sp.polygons", nl_poly)
```

```
rgb.palette <- colorRampPalette(c("lightyellow","green","orange","brown"),
     space = "rgb")

for (i in 1:ns){
png(filename = paste("org1.sim",i,".png",sep=""), width=600, height=600,
     units="px",pointsize=12, bg="white", res=NA)
print(spplot(org1.sim_ns[,i], main = list(label=paste("Co-simulation",i),cex=1.5),
col.regions=rgb.palette(22),
xlab=list(paste(d1,"-",d2,"cm"),cex=1.5),
at=c(seq(0,10,0.5),120),
colorkey=list(space="right",width=2,at=1:22,labels=list(cex=2,at=1:22,
labels=c("","","1","","","","3","","","","5","","","","7","","","","9 >",
     "","","","")),
sp.layout=list(nl,text)))
dev.off()
}

# Save figures for first layer

text <- list("sp.text",c(60000,550000),paste(d3,"-",d4,"cm"),cex=1.5)

for (i in 1:ns){
png(filename = paste("org2.sim",i,".png",sep=""), width=600, height=600,
     units="px",pointsize=12, bg="white", res=NA)
print(spplot(org2.sim_ns[,i], main = list(label=paste("Co-simulation",i),cex=1.5),
col.regions=rgb.palette(22),
xlab=list(paste(d3,"-",d4,"cm"),cex=1.5),
at=c(seq(0,10,0.5),120),
colorkey=list(space="right",width=2,at=1:22,labels=list(cex=2,at=1:22,
labels=c("","","1","","","","3","","","","5","","","","7","","","","9 >",
     "","","","")),
sp.layout=list(nl,text)))
dev.off()
}

# Save figures for first layer

text <- list("sp.text",c(60000,550000),paste(d5,"-",d6,"cm"),cex=1.5)

for (i in 1:ns){
png(filename = paste("org3.sim",i,".png",sep=""), width=600, height=600,
     units="px",pointsize=12, bg="white", res=NA)
print(spplot(org3.sim_ns[,i], main = list(label=paste("Co-simulation",i),cex=1.5),
col.regions=rgb.palette(22),
xlab=list(paste(d5,"-",d6,"cm"),cex=1.5),
at=c(seq(0,10,0.5),120),
colorkey=list(space="right",width=2,at=1:22,labels=list(cex=2,at=1:22,
labels=c("","","1","","","","3","","","","5","","","","7","","","","9 >",
     "","","","")),
sp.layout=list(nl,text)))
dev.off()
}
```

Appendix F

R code Mean spring water table depth (MSW)

```
#1. Libraries needed to perform all computations

library(foreign)
library(sp)
library(rgdal)
library(gstat)
library(lattice)
library(maptools)
library(RColorBrewer)
library(RODBC)

# 2. Table LSK_ALG data extraction and erasing of records with duplicated coordinates

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")

# extract the LSK_ALG table with 7 columns (ID,ST_NR,DATUM,X,Y,MHW,MLW)
LSK <- sqlQuery(channel=channel,
query = "SELECT DISTINCT LSK_ID, ST_NR, DATUM, X, Y, MHW, MLW FROM LSK_ALG")

# erase duplicated records where ST_NR > 90
sp=split(LSK,LSK$ST_NR>90)
LSK_01=sp$"FALSE"

# erase other duplicated records
sp01=split(LSK_01,LSK_01$LSK_ID==3363 | LSK_01$LSK_ID==3192 |
    LSK_01$LSK_ID==3113 | LSK_01$LSK_ID==3115 |LSK_01$LSK_ID==3138 |
    LSK_01$LSK_ID==3114 |LSK_01$LSK_ID==350 |LSK_01$LSK_ID==657|
    LSK_01$LSK_ID==809|LSK_01$LSK_ID==2702|LSK_01$LSK_ID==1658|
    LSK_01$LSK_ID==3281|LSK_01$LSK_ID==3194)

sp02=sp01$"FALSE"

# data frame with unique records only
LSK_nondupl=data.frame(LSK_ID=sp02$LSK_ID,X=sp02$X,Y=sp02$Y,
    MHW=sp02$MHW,MLW=sp02$MLW)

# write data frame to a file
write.table(LSK_nondupl,"LSK_nondupl.txt",sep="\t",row.names=FALSE)

# disconnect from database
odbcClose(channel = channel)

# 3. Table PFB_ALG data extraction and erasing of records with duplicated coordinates

# connect to database
channel <- odbcConnect(dsn = "deltaBIS", uid = "BISUSER", pwd = "BISUSER")
```

```
# extract PFB_ALG table with 6 columns (ID,X,Y,MHW,HLG,JAAR)
PFB <- sqlQuery(channel=channel,
query = "SELECT DISTINCT  PFB_ID, X, Y, MHW, MLW, JAAR FROM PFB_ALG")

PFB[,2]=round(PFB[,2])  # round coordinates
PFB[,3]=round(PFB[,3])

# identify duplicated records
PFB1=data.frame(X=PFB$X,Y=PFB$Y)
PFB11=data.frame(PFB,duplicated=duplicated(PFB1))

# split into unique and duplicated
sp=split(PFB11,PFB11$duplicated==TRUE)

PFB_01=sp$"FALSE"   # extract unique records

# data frame with unique records only
PFB_nondupl=data.frame(PFB_ID=PFB_01$PFB_ID,X=PFB_01$X,Y=PFB_01$Y,
    MHW=PFB_01$MHW,MLW=PFB_01$MLW)

# write the table to a file
write.table(PFB_nondupl,"PFB_nondupl.txt",sep="\t",row.names=FALSE)

# disconnect from database
odbcClose(channel = channel)

# 4. Data preparation and transformation

# read tables with unique records
PFB=read.table("PFB_nondupl.txt",sep="\t",header=TRUE,na.strings="NA")
LSK=read.table("LSK_nondupl.txt",sep="\t",header=TRUE,na.strings="999")

# merge the two tables
m1=merge(PFB,LSK,all=TRUE)
m2=data.frame(X=m1$X,Y=m1$Y,MHW=m1$MHW,MLW=m1$MLW)

# compute the mean water-table level
MSW=data.frame(m2,MSW=(5.4+0.83*m2$MHW+0.19*m2$MLW))

# read the bispnt.txt file
bispnt=read.table('bispnt.txt',header=TRUE)

# merge tables
M1=merge(MSW,bispnt,by=c("X","Y"))

# remove records with NA values
MSW=subset(M1,!is.na(M1$MSW))

# force MSW values to be between MHW and MLW
for (i in 1:length(MSW$MSW)){
if (MSW$MSW[i]>MSW$MLW[i]) MSW$MSW[i]=MSW$MLW[i]
if (MSW$MSW[i]<MSW$MHW[i]) MSW$MSW[i]=MSW$MHW[i]
}

# log transformation of MSW
MSW=data.frame(MSW,t_MSW=log(MSW$MSW-min(MSW$MSW)+1))

# 5. Histograms (an example for original MSW data)

windows(width = 5,height = 5)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(MSW$MSW,main='',xlab='MSW',col='lightblue')

# 6. Box-plots (an example for transformed MSW data and each of the seven soil types)

windows(width = 10,height = 5)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
boxplot(split(MSW$t_MSW,MSW$S),xlab='Soil type',main='trans_MSW')

# 7. Normal Q-Q Plot for transformed MSW data

windows(width = 5,height = 5)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
QQ=qqnorm(MSW$t_MSW,xlab='t_MSW')
```

```
# 8. Variography

# compute residuals
MSW=data.frame(MSW,res=0)
for (i in 1:length(MSW$res)){
if (MSW$S[i]==1) MSW$res[i]=(MSW$t_MSW[i]-tapply(MSW$t_MSW,MSW$S,mean)[1])/
                    tapply(MSW$t_MSW,MSW$S,sd)[1]
if (MSW$S[i]==2) MSW$res[i]=(MSW$t_MSW[i]-tapply(MSW$t_MSW,MSW$S,mean)[2])/
                    tapply(MSW$t_MSW,MSW$S,sd)[2]
if (MSW$S[i]==3) MSW$res[i]=(MSW$t_MSW[i]-tapply(MSW$t_MSW,MSW$S,mean)[3])/
                    tapply(MSW$t_MSW,MSW$S,sd)[3]
if (MSW$S[i]==4) MSW$res[i]=(MSW$t_MSW[i]-tapply(MSW$t_MSW,MSW$S,mean)[4])/
                    tapply(MSW$t_MSW,MSW$S,sd)[4]
if (MSW$S[i]==5) MSW$res[i]=(MSW$t_MSW[i]-tapply(MSW$t_MSW,MSW$S,mean)[5])/
                    tapply(MSW$t_MSW,MSW$S,sd)[5]
if (MSW$S[i]==6) MSW$res[i]=(MSW$t_MSW[i]-tapply(MSW$t_MSW,MSW$S,mean)[6])/
                    tapply(MSW$t_MSW,MSW$S,sd)[6]
if (MSW$S[i]==7) MSW$res[i]=(MSW$t_MSW[i]-tapply(MSW$t_MSW,MSW$S,mean)[7])/
                    tapply(MSW$t_MSW,MSW$S,sd)[7]
}
coordinates(MSW)=~X+Y

# compute residual t_MSW  variogram zoomed to shorter distances
rMSW=gstat(id=c("t_MSWres"),formula=res~1,data=MSW)
rMSW.vg=variogram(rMSW,boundaries=c(500,700,5000,10000,15000,20000))
rMSW.vgm=fit.variogram(rMSW.vg,vgm(1,'Sph',5000,1))

# plot residual variogram
windows(width = 8, height = 6)
plot(rMSW.vg,rMSW.vgm,main='t_MSW residual',plot.nu=T)

# define variogram model
rMSW.vgm=vgm(0.72,"Sph",5000,0.28)

# 9. Spatial prediction at 500 locations

# read table of target prediction locations with soil types
grid1 <- read.table('sim1.txt',header=TRUE,sep='\t')
coordinates(grid1)=~X+Y

# define gstat object to predict residuals
sMSW=gstat(id=c("t_MSWres"),formula=res~1,data=MSW,model=rMSW.vgm,beta=0,nmax=100)

# ready to predict t_MSW residuals using simple kriging
blup = predict.gstat(sMSW, newdata = grid1, BLUE = FALSE)

# compute t_MSW prediction, multiply by stdev and add the mean value of
# each soil type
pred=data.frame(grid1,sk.pred=blup$t_MSWres.pred,stdev=0,mean=0,trans.pred=0,
   real.pred=0)

for(i in 1:length(pred$stdev)){
if (pred$S[i]==1) pred$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[1]
if (pred$S[i]==2) pred$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[2]
if (pred$S[i]==3) pred$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[3]
if (pred$S[i]==4) pred$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[4]
if (pred$S[i]==5) pred$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[5]
if (pred$S[i]==6) pred$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[6]
if (pred$S[i]==7) pred$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[7]
}

for (i in 1:length(pred$mean)){
if (pred$S[i]==1) pred$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[1]
if (pred$S[i]==2) pred$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[2]
if (pred$S[i]==3) pred$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[3]
if (pred$S[i]==4) pred$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[4]
if (pred$S[i]==5) pred$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[5]
if (pred$S[i]==6) pred$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[6]
if (pred$S[i]==7) pred$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[7]
}

pred$trans.pred=pred$sk.pred*pred$stdev+pred$mean  # compute predictions
pred$real.pred=exp(pred$trans.pred)+min(MSW$MSW)-1 # back transformation

# plot predicted values
```

```
nl_poly=readShapePoly("NL.shp") # read NL boundary
coordinates(pred)=~X+Y

spplot(pred['real.pred'],main='MSW prediction',
    col.regions=bpy.colors(),sp.layout=list("sp.polygons",nl_poly),
    key.space=list(x=0.02,y=0.96,corner=c(0,1)),
    cuts=c(0,30,60,90,120,150,200,250,500,3500),
    legendEntries=c('0-30 [cm]','30-60','60-90','90-120',
    '120-150','150-200','200-250','250-500','500 <'))

# bubble plot
bubble(pred,'real.pred',col=('blue'),sp.layout=list("sp.polygons",nl_poly),
    key.entries=c(30,60,90,120,150,200,250,500),main='MSW prediction')

# 10. Spatial prediction over the whole of the Netherlands

# read predictor maps:
soil = readGDAL("soil1.asc")
soil$SP = soil$band1
soil$SR = readGDAL("soil2.asc")$band1
soil$SC = readGDAL("soil3.asc")$band1
soil$CN = readGDAL("soil4.asc")$band1
soil$CC = readGDAL("soil5.asc")$band1
soil$LN = readGDAL("soil6.asc")$band1
soil$PN = readGDAL("soil7.asc")$band1
soil$band1=NULL

soil2=data.frame(soil,S=0)

for (i in 1:length(soil2$S)){
if (soil2$SP[i]==1) soil2$S[i]=1
if (soil2$SR[i]==1) soil2$S[i]=2
if (soil2$SC[i]==1) soil2$S[i]=3
if (soil2$CN[i]==1) soil2$S[i]=4
if (soil2$CC[i]==1) soil2$S[i]=5
if (soil2$LN[i]==1) soil2$S[i]=6
if (soil2$PN[i]==1) soil2$S[i]=7
}

coordinates(soil2)=~x+y
# define gstat object to predict residuals
sMSW=gstat(id=c("t_MSWres"),formula=res~1,data=MSW,model=rMSW.vgm,beta=0,nmax=100)

# ready to predict t_MSW residuals using simple kriging
blup2 = predict.gstat(sMSW, newdata = soil2, BLUE = FALSE)

pred2=data.frame(soil2,pred.res=blup2$t_MSWres.pred,stdev=0,mean=0,trans.pred=0,
     real.pred=0)

for(i in 1:length(pred2$stdev)){
if (pred2$S[i]==1) pred2$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[1]
if (pred2$S[i]==2) pred2$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[2]
if (pred2$S[i]==3) pred2$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[3]
if (pred2$S[i]==4) pred2$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[4]
if (pred2$S[i]==5) pred2$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[5]
if (pred2$S[i]==6) pred2$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[6]
if (pred2$S[i]==7) pred2$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[7]
}

for (i in 1:length(pred2$mean)){
if (pred2$S[i]==1) pred2$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[1]
if (pred2$S[i]==2) pred2$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[2]
if (pred2$S[i]==3) pred2$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[3]
if (pred2$S[i]==4) pred2$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[4]
if (pred2$S[i]==5) pred2$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[5]
if (pred2$S[i]==6) pred2$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[6]
if (pred2$S[i]==7) pred2$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[7]
}

pred2$trans.pred=pred2$pred.res*pred2$stdev+pred2$mean  # trans_MSW prediction
pred2$real.pred=exp(pred2$trans.pred)+min(MSW$MSW)-1        # back transformation

# map of predicted values
gridded(pred2)=TRUE
nl_poly=readShapePoly("NL.shp") # read NL boundary
```

```
coordinates(pred2)=~x+y
spplot(pred2['real.pred'],main='MSW prediction',
col.regions=c('darkblue','blue','skyblue','lightblue','yellow','orange',
     'orangered','red'),
at=c(30,60,90,120,150,200,250,500,3500),
colorkey=list(space='right',at=1:9,labels=c('0 [cm]','30','60','90','120','150',
     '200','250','500 >')))

# 11. Simulation at 500 target locations

# read table with target prediction locations and soil types
grid1 <- read.table('sim1.txt',header=TRUE,sep='\t')
coordinates(grid1)=~X+Y

# define gstat object to predict residuals
sMSW=gstat(id=c("t_MSWres"),formula=res~1,data=MSW,model=rMSW.vgm,beta=0,nmax=100)

# ready to predict trans_MSW residuals using Gaussian simulation based
# on simple kriging
blup = predict.gstat(sMSW, newdata = grid1, BLUE = FALSE,nsim=3)

# compute final predictions
sim=data.frame(blup,S=grid1$S,stdev=0,mean=0,t.sim1=0,r.sim1=0,t.sim2=0,r.sim2=0,
     t.sim3=0,r.sim3=0)

for(i in 1:length(sim$stdev)){
if (sim$S[i]==1) sim$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[1]
if (sim$S[i]==2) sim$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[2]
if (sim$S[i]==3) sim$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[3]
if (sim$S[i]==4) sim$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[4]
if (sim$S[i]==5) sim$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[5]
if (sim$S[i]==6) sim$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[6]
if (sim$S[i]==7) sim$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[7]
}

for (i in 1:length(sim$mean)){
if (sim$S[i]==1) sim$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[1]
if (sim$S[i]==2) sim$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[2]
if (sim$S[i]==3) sim$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[3]
if (sim$S[i]==4) sim$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[4]
if (sim$S[i]==5) sim$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[5]
if (sim$S[i]==6) sim$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[6]
if (sim$S[i]==7) sim$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[7]
}

# compute simulated values and back transformation
sim$t.sim1=sim$sim1*sim$stdev+sim$mean
sim$r.sim1=exp(sim$t.sim1)+min(MSW$MSW)-1 # back transformation

sim$t.sim2=sim$sim2*sim$stdev+sim$mean
sim$r.sim2=exp(sim$t.sim2)+min(MSW$MSW)-1

sim$t.sim3=sim$sim3*sim$stdev+sim$mean
sim$r.sim3=exp(sim$t.sim3)+min(MSW$MSW)-1

# plot simulation 1
nl_poly=readShapePoly("NL.shp") # read NL boundary
coordinates(sim)=~X+Y
spplot(sim['r.sim1'],main='MSW simulation 1',
col.regions=bpy.colors(),sp.layout=list("sp.polygons",nl_poly),
key.space=list(x=0.02,y=0.96,corner=c(0,1)),
cuts=c(0,30,60,90,120,150,200,250,500,3500),
legendEntries=c('0-30 [cm]','30-60','60-90','90-120','120-150','150-200',
     '200-250','250-500','500 <'))

# histogram of simulation 3 (transformed values)
windows(width = 5,height = 5)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(sim$t.sim3,main='',xlab='t_MSW simulation 3',ylab='Frequency',freq=F,
     col='skyblue',ylim=c(0,0.8))

# 12. Simulation over the whole area of Netherlands

# read predictor maps:
soil = readGDAL("soil1.asc")
```

```
soil$SP = soil$band1
soil$SR = readGDAL("soil2.asc")$band1
soil$SC = readGDAL("soil3.asc")$band1
soil$CN = readGDAL("soil4.asc")$band1
soil$CC = readGDAL("soil5.asc")$band1
soil$LN = readGDAL("soil6.asc")$band1
soil$PN = readGDAL("soil7.asc")$band1
soil$band1=NULL

soil2=data.frame(soil,S=0)

for (i in 1:length(soil2$S)){
if (soil2$SP[i]==1) soil2$S[i]=1
if (soil2$SR[i]==1) soil2$S[i]=2
if (soil2$SC[i]==1) soil2$S[i]=3
if (soil2$CN[i]==1) soil2$S[i]=4
if (soil2$CC[i]==1) soil2$S[i]=5
if (soil2$LN[i]==1) soil2$S[i]=6
if (soil2$PN[i]==1) soil2$S[i]=7
}
coordinates(soil2)=~x+y

# define gstat object to predict residuals
sMSW=gstat(id=c("t_MSWres"),formula=res~1,data=MSW,model=rMSW.vgm,beta=0,nmax=100)

# ready to predict trans_MSW residuals using simple kriging
blup.2 = predict.gstat(sMSW, newdata = soil2, BLUE = FALSE,nsim=3)

sim.2=data.frame(blup.2,S=soil2$S,stdev=0,mean=0,t.sim1=0,r.sim1=0,t.sim2=0,r.sim2=0,
     t.sim3=0,r.sim3=0)

for(i in 1:length(sim.2$stdev)){
if (sim.2$S[i]==1) sim.2$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[1]
if (sim.2$S[i]==2) sim.2$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[2]
if (sim.2$S[i]==3) sim.2$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[3]
if (sim.2$S[i]==4) sim.2$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[4]
if (sim.2$S[i]==5) sim.2$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[5]
if (sim.2$S[i]==6) sim.2$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[6]
if (sim.2$S[i]==7) sim.2$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[7]
}

for (i in 1:length(sim.2$mean)){
if (sim.2$S[i]==1) sim.2$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[1]
if (sim.2$S[i]==2) sim.2$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[2]
if (sim.2$S[i]==3) sim.2$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[3]
if (sim.2$S[i]==4) sim.2$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[4]
if (sim.2$S[i]==5) sim.2$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[5]
if (sim.2$S[i]==6) sim.2$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[6]
if (sim.2$S[i]==7) sim.2$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[7]
}

# compute simulations and back transformation
sim.2$t.sim1=sim.2$sim1*sim.2$stdev+sim.2$mean      # t_MSW simulation
sim.2$r.sim1=exp(sim.2$t.sim1)+min(MSW$MSW)-1   # back transformation

sim.2$t.sim2=sim.2$sim2*sim.2$stdev+sim.2$mean
sim.2$r.sim2=exp(sim.2$t.sim2)+min(MSW$MSW)-1

sim.2$t.sim3=sim.2$sim3*sim.2$stdev+sim.2$mean
sim.2$r.sim3=exp(sim.2$t.sim3)+min(MSW$MSW)-1

# read NL boundary
nl_poly=readShapePoly("NL.shp")

coordinates(sim.2)=~x+y
gridded(sim.2)=TRUE

# plot simulation 1
spplot(sim.2['r.sim1'],main='MSW simulation 1',
col.regions=c('darkblue','blue','skyblue','lightblue','yellow','orange',
     'orangered','red'),
at=c(30,60,90,120,150,200,250,500,3500),
colorkey=list(space='right',at=1:9,labels=c('0 [cm]','30','60','90','120','150',
     '200','250','500 >')))
```

```
# histogram of simulation 1 (transformed values)
windows(width = 5,height = 5)
par(mar=c(4,4,1,1)+0.6,cex.lab=1.4,cex.axis=1.5)
hist(sim.2$t.sim1,main='',xlab='t_MSW simulation 1',ylab='Frequency',freq=F,
     col='skyblue',ylim=c(0,0.8))

# 13. 3000 simulations at 500 target locations

# read table with 3000 soil type simulations
s3000=read.table("simuHardkcond.prn")

# create table to which MSW simulations will be written
MSW3000=data.frame(X=s3000$V1,Y=s3000$V2)

# define gstat object to predict residuals
sMSW=gstat(id=c("t_MSWres"),formula=res~1,data=MSW,model=rMSW.vgm,beta=0,nmax=100)

# make a loop for 3000 simulations
for (i in 3:3002){
grid1=data.frame(X=s3000$V1,Y=s3000$V2,S=s3000[,i])
coordinates(grid1)=~X+Y

# ready to predict trans_MSW residuals using Gaussian simulation based on
# simple kriging
blup = predict.gstat(sMSW, newdata = grid1, BLUE = FALSE,nsim=1)

# compute final predictions
sim=data.frame(blup,S=grid1$S,stdev=0,mean=0)

for(i in 1:length(sim$stdev)){
if (sim$S[i]==1) sim$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[1]
if (sim$S[i]==2) sim$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[2]
if (sim$S[i]==3) sim$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[3]
if (sim$S[i]==4) sim$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[4]
if (sim$S[i]==5) sim$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[5]
if (sim$S[i]==6) sim$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[6]
if (sim$S[i]==7) sim$stdev[i]=tapply(MSW$t_MSW,MSW$S,sd)[7]
}
for (i in 1:length(sim$mean)){
if (sim$S[i]==1) sim$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[1]
if (sim$S[i]==2) sim$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[2]
if (sim$S[i]==3) sim$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[3]
if (sim$S[i]==4) sim$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[4]
if (sim$S[i]==5) sim$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[5]
if (sim$S[i]==6) sim$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[6]
if (sim$S[i]==7) sim$mean[i]=tapply(MSW$t_MSW,MSW$S,mean)[7]
}
# compute simulated values and back transformation
sim$t.sim1=sim$sim1*sim$stdev+sim$mean
sim$r.sim1=exp(sim$t.sim1)+min(MSW$MSW)-1 # back transformation
MSW3000=data.frame(MSW3000,gvg=round(sim$r.sim1,2))
}

write.table(MSW3000,'MSW3000.txt',col.names=F)
```

Verschenen documenten in de reeks Werkdocumenten van de Wettelijke Onderzoekstaken Natuur & Milieu vanaf 2007

Werkdocumenten zijn verkrijgbaar bij het secretariaat van Unit Wettelijke Onderzoekstaken Natuur & Milieu, te Wageningen. T 0317 – 48 54 71; F 0317 – 41 90 00; E info.wnm@wur.nl
De werkdocumenten zijn ook te downloaden via de WOt-website www.wotnatuurenmilieu.wur.nl

2007

47 *Ten Berge, H.F.M., A.M. van Dam, B.H. Janssen & G.L. Velthof.* Mestbeleid en bodemvruchtbaarheid in de Duin- en Bollenstreek; Advies van de CDM-werkgroep Mestbeleid en Bodemvruchtbaarheid in de Duin- en Bollenstreek

48 *Kruit, J. & I.E. Salverda.* Spiegeltje, spiegeltje aan de muur, valt er iets te leren van een andere plannings-cultuur?

49 *Rijk, P.J., E.J. Bos & E.S. van Leeuwen.* Nieuwe activiteiten in het landelijk gebied. Een verkennende studie naar natuur en landschap als vestigingsfactor

50 *Ligthart, S.S.H.* Natuurbeleid met kwaliteit. Het Milieu- en Natuurplanbureau en natuurbeleidsevaluatie in de periode 1998-2006

51 *Kennismarkt 22 maart 2007; van onderbouwend onderzoek Wageningen UR naar producten MNP in 27 posters*

52 *Kuindersma, W., R.I. van Dam & J. Vreke.* Sturen op niveau. Perversies tussen nationaal natuurbeleid en besluitvorming op gebiedsniveau.

53.1 *Reijnen, M.J.S.M.* Indicators for the 'Convention on Biodiversity 2010'. National Capital Index version 2.0

53.3 *Windig, J.J., M.G.P. van Veller & S.J. Hiemstra.* Indicatoren voor 'Convention on Biodiversity 2010'. Biodiversiteit Nederlandse landbouwhuisdieren en gewassen

53.4 *Melman, Th.C.P. & J.P.M. Willemen.* Indicators for the 'Convention on Biodiversity 2010'. Coverage protected areas.

53.6 *Weijden, W.J. van der, R. Leewis & P. Bol.* Indicatoren voor 'Convention on Biodiversity 2010'. Indicatoren voor het invasieproces van exotische organismen in Nederland

53.7a *Nijhof, B.S.J., C.C. Vos & A.J. van Strien.* Indicators for the 'Convention on Biodiversity 2010'. Influence of climate change on biodiversity.

53.7b *Moraal, L.G.* Indicatoren voor 'Convention on Biodiversity 2010'. Effecten van klimaatverandering op insectenplagen bij bomen.

53.8 *Fey-Hofstede, F.E. & H.W.G. Meesters.* Indicators for the 'Convention on Biodiversity 2010'. Exploration of the usefulness of the Marine Trophic Index (MTI) as an indicator for sustainability of marine fisheries in the Dutch part of the North Sea.

53.9 *Reijnen, M.J.S.M.* Indicators for the 'Convention on Biodiversity 2010'. Connectivity/fragmentation of ecosystems: spatial conditions for sustainable biodiversity

53.11 *Gaaff, A. & R.W. Verburg.* Indicators for the 'Convention on Biodiversity 2010' Government expenditure on land acquisition and nature development for the National Ecological Network (EHS) and expenditure for international biodiversity projects

53.12 *Elands, B.H.M. & C.S.A. van Koppen.* Indicators for the 'Convention on Biodiversity 2010'. Public awareness and participation

54 *Broekmeyer, M.E.A. & E.P.A.G. Schouwenberg & M.E. Sanders & R. Pouwels.* Synergie Ecologische Hoofdstructuur en Natura 2000-gebieden. Wat stuurt het beheer?

55 *Bosch, F.J.P. van den.* Draagvlak voor het Natura 2000-gebiedenbeleid. Onder relevante betrokkenen op regionaal niveau

56 *Jong, J.J. & M.N. van Wijk, I.M. Bouwma.* Beheerskosten van Natura 2000-gebieden

57 *Pouwels, R. & M.J.S.M. Reijnen & M. van Adrichem & H. Kuipers.* Ruimtelijke condities voor VHR-soorten

58 Niet verschenen/ vervallen

59 *Schouwenberg, E.P.A.G.* Huidige en toekomstige stikstofbelasting op Natura 2000-gebieden

60 Niet verschenen/ vervallen

61 *Jaarrapportage 2006.* WOT-04-001 – ME-AVP

62 *Jaarrapportage 2006.* WOT-04-002 – Onderbouwend Onderzoek

63 *Jaarrapportage 2006.* WOT-04-003 – Advisering Natuur & Milieu

64 *Jaarrapportage 2006.* WOT-04-385 – Milieuplanbureaufunctie

65 *Jaarrapportage 2006.* WOT-04-394 – Natuurplanbureaufunctie

66 *Brasser E.A., M.F. van de Kerkhof, A.M.E. Groot, L. Bos-Gorter, M.H. Borgstein, H. Leneman* Verslag van de Dialogen over Duurzame Landbouw in 2006

67 *Hinssen, P.J.W.* Wettelijke Onderzoekstaken Natuur & Milieu. Werkplan 2007

68 *Nieuwenhuizen, W. & J. Roos Klein Lankhorst.* Landschap in Natuurbalans 2006; Landschap in verandering tussen 1990 en 2005; Achtergronddocument bij Natuurbalans 2006.

69 *Geelen, J. & H. Leneman.* Belangstelling, motieven en knelpunten van natuuraanleg door grondeigenaren. Uitkomsten van een marktonderzoek.

70 *Didderen, K., P.F.M. Verdonschot, M. Bleeker.* Basiskaart Natuur aquatisch. Deel 1: Beleidskaarten en prototype

71 *Boesten, J.J.T.I, A. Tiktak & R.C. van Leerdam.* Manual of PEARLNEQ v4

72 *Grashof-Bokdam, C.J., J. Frissel, H.A.M. Meeuwsen & M.J.S.M. Reijnen.* Aanpassing graadmeter natuurwaarde voor het agrarisch gebied

73 *Bosch, F.J.P. van den.* Functionele agrobiodiversiteit. Inventarisatie van nut, noodzaak en haalbaarheid van het ontwikkelen van een indicator voor het MNP

74 *Kistenkas, F.H. en M.E.A. Broekmeyer.* Natuur, landschap en de Wet algemene bepalingen omgevingsrecht

75 *Luttik, J., F.R. Veeneklaas, J. Vreke, T.A. de Boer, L.M. van den Berg & P. Luttik.* Investeren in landschapskwaliteit; De toekomstige vraag naar landschappen om in te wonen, te werken en te ontspannen

76 *Vreke, J.* Evaluatie van natuurbeleidsprocessen

77 *Apeldoorn, R.C. van,* Working with biodiversity goals in European directives. A comparison of the implementation of the Birds and Habitats Directives and the Water Framework Directive in the Netherlands, Belgium, France and Germany

78 *Hinssen, P.J.W.* Werkprogramma 2008; Unit Wettelijke Onderzoekstaken Natuur & Milieu (WOT-04). Onderdeel Planbureaufuncties Natuur en Milieu.

79 *Custers, M.H.G.* Betekenissen van Landschap in onderzoek voor het Milieu- en Natuurplanbureau; een bibliografisch overzicht

80 *Vreke,J., J.L.M. Donders, B.H.M. Elands, C.M. Goossen, F. Langers, R. de Niet & S. de Vries.* Natuur en landschap voor mensen Achtergronddocument bij Natuurbalans 2007

81 *Bakel, P.J.T. van, T. Kroon, J.G. Kroes, J. Hoogewoud, R. Pastoors, H.Th.L. Massop, D.J.J. Walvoort.* Reparatie Hydrologie voor STONE 2.1. Beschrijving reparatie-acties, analyse resultaten en beoordeling plausibiliteit.

2008

82 *Kistenkas, F.H. & W. Kuindersma..* Jurisprudentie-monitor natuur 2005-2007; Rechtsontwikkelingen Natura 2000 en Ecologische Hoofdstructuur

83 *Berg, F. van den, P.I. Adriaanse, J. A. te Roller, V.C. Vulto & J.G. Groenwold..* SWASH Manual 2.1; User's Guide version 2

84 *Smits, M.J., M.J. Bogaardt, D. Eaton, P. Roza & T. Selnes..* Tussen de bomen het geld zien. Programma Beheer en vergelijkbare regelingen in het buitenland (een quick-scan)

85 *Dijk, T.A. van, J.J.M. Driessen, P.A.I. Ehlert, P.H. Hotsma, M.H.M.M. Montforts, S.F. Plessius & O. Oenema..* Protocol beoordeling stoffen Meststoffenwet; versie 1.0

86 *Goossen, C.M., H.A.M. Meeuwsen, G.J. Franke & M.C. Kuyper. V*erkenning Europese versie van de website www.daarmoetikzijn.nl.

87 *Helming, J.F.M. & R.A.M. Schrijver.* Economische effecten van inzet van landbouwsubsidies voor milieu, natuur en landschap in Nederland; Achtergrond bij het MNP-rapport 'Opties voor Europese landbouw-subsidies

88 *Hinssen, P.J.W.* Werkprogramma 2008; Unit Wettelijke Onderzoekstaken Natuur & Milieu (WOT-04). Programma 001/003/005

90 *Kramer, H.* Geografisch Informatiesysteem Bestaande Natuur; Beschrijving IBN1990t en pilot ontwikkeling BN2004

92 *Jaarrapportage 2007.* WOT-04-001 – Koepel

93 *Jaarrapportage 2007.* WOT-04-002 – Onderbouwend Onderzoek

94 *Jaarrapportage 2007.* WOT-04-003 – Advisering Natuur & Milieu

95 *Jaarrapportage 2007.* WOT-04-005 – M-AVP

96 *Jaarrapportage 2007.* WOT-04-006 – Natuurplanbureaufunctie

97 *Jaarrapportage 2007.* WOT-04-007 – Milieuplanbureaufunctie

98 *Wamelink, G.W.W.* Gevoeligheids- en onzekerheids-analyse van SUMO

99 *Hoogeveen, M.W., H.H. Luesink, L.J. Mokveld & J.H. Wisman.* Ammoniakemissies uit de landbouw in Milieubalans 2006: uitgangspunten en berekeningen

100 *Kennismarkt 3 april 2008; Van onderbouwend onderzoek Wageningen UR naar producten MNP*

101 *Mansfeld, M.J.M. van & J.A. Klijn.* "Balansen op de weegschaal". Terugblik op acht jaar Natuurbalansen (1996-2005)

102 *Sollart, K.M. & J. Vreke.* Het faciliteren van natuur- en milieueducatie in het basisonderwijs; NME-ondersteuning in de provincies

103 *Berg, F. van den, A. Tiktak, J.G. Groenwold, D.W.G. van Kraalingen, A.M.A. van der Linden & J.J.T.I. Boesten*, Documentation update for GeoPEARL 3.3.3

104 *Wijk, M.N., van (redactie).* Aansturing en kosten van het natuurbeheer. Ecologische effectiviteit regelingen natuurbeheer

105 *Selnes, T. & P. van der Wielen.* Tot elkaar veroordeeld? Het belang van gebiedsprocessen voor de natuur

106 *Annual reports for 2007; Programme WOT-04*

107 *Pouwels, R. J.G.M. van der Greft, M.H.C. van Adrichem, H. Kuiper, R. Jochem & M.J.S.M. Reijnen.* LARCH Status A

108 *Wamelink, G.W.W.* Technical Documentation for SUMO2 v. 3.2.1,

109 *Wamelink, G.W.W., J.P. Mol-Dijkstra & G.J. Reinds.* Herprogrammeren van SUMO2. Verbetering in het kader van de modelkwaliteitsslag

110 *Salm, C. van der, T. Hoogland & D.J.J. Walvoort.* Verkenning van de mogelijkheden voor de ontwikkeling van een metamodel voor de uitspoeling van stikstof uit landbouwgronden

111 *Dobben H.F. van & R.M.A. Wegman.* Relatie tussen bodem, atmosfeer en vegetatie in het Landelijk Meetnet Flora (LMF)

112 *Smits, M.J.W. & M.J. Bogaardt.* Kennis over de effecten van EU-beleid op natuur en landschap

113 *Maas,G.J. & H. van Reuler.* Boomkwekerij en aardkunde in Nederland,

114 *Lindeboom, H.J., R. Witbaard, O.G. Bos & H.W.G. Meesters.* Gebiedsbescherming Noordzee, habitattypen, instandhoudingdoelen en beheermaatregelen

115 *Leneman, H., J. Vader, L.H.G. Slangen, K.H.M. Bommel, N.B.P. Polman, M.W.M. van der Elst & C. Mijnders.* Groene diensten in Nationale Landschappen- Potenties bij een veranderende landbouw,

116 *Groeneveld, R.A. & D.P. Rudrum.* Habitat Allocation to Maximize BiOdiversity, A technical description of the HAMBO model

117 *Kruit, J., M. Brinkhuijzen & H. van Blerck.* Ontwikkelen met kwaliteit. Indicatoren voor culturele vernieuwing en architectonische vormgeving

118 *Roos-Klein Lankhorst, J.* Beheers- en Ontwikkelingsplan 2007: Kennismodel Effecten Landschap Kwaliteit; Monitoring Schaal; BelevingsGIS

119 *Henkens, R.J.H.G.* Kwalitatieve analyse van knelpunten tussen Natura 2000-gebieden en waterrecreatie

120 *Verburg, R.W., I.M. Jorritsma & G.H.P. Dirkx.* Quick scan naar de processen bij het opstellen van beheerplannen van Natura 2000-gebieden. Een eerste verkenning bij provincies, Rijkswaterstaat en Dienst Landelijk Gebied

121 *Daamen, W.P.* Kaart van de oudste bossen in Nederland; Kansen op hot spots voor biodiversiteit

122 *Lange de, H.J., G.H.P. Arts & W.C.E.P. Verberk.* Verkenning CBD 2010-indicatoren zoetwater. Inventarisatie en uitwerking relevante indicatoren voor Nederland

123 *Vreke, J., N.Y. van der Wulp, J.L.M. Donders, C.M. Goossen, T.A. de Boer & R. Henkens.* Recreatief gebruik van water. Achtergronddocument Natuurbalans 2008

124 *Oenema, O. & J.W.H. van der Kolk*. Moet het eenvoudiger? Een essay over de complexiteit van het milieubeleid

125 *Oenema, O. & A. Tiktak.* Niets is zonder grond; Een essay over de manier waarop samenlevingen met hun grond omgaan

2009

126 *Kamphorst, D.A.* Keuzes in het internationale biodiversiteitsbeleid; Verkenning van de beleidstheorie achter de internationale aspecten van het Beleidsprogramma Biodiversiteit (2008-2011)

127 *Dirkx, G.H.P. & F.J.P. van den Bosch.* Quick scan gebruik Catalogus groenblauwe diensten

128 *Loeb, R. & P.F.M. Verdonschot.* Complexiteit van nutriëntenlimitaties in oppervlaktewateren

129 *Kruit, J. & P.M. Veer.* Herfotografie van landschappen; Landschapsfoto's van de 'Collectie de Boer' als uitgangspunt voor het in beeld brengen van ontwikkelingen in het landschap in de periode 1976-2008

130 *Oenema, O., A. Smit & J.W.H. van der Kolk.* Indicatoren Landelijk Gebied; werkwijze en eerste resultaten

131 *Agricola, H.J.A.J. van Strien, J.A. Boone, M.A. Dolman, C.M. Goossen, S. de Vries, N.Y. van der Wulp, L.M.G. Groenemeijer, W.F. Lukey & R.J. van Til.* Achtergrond-document Nulmeting Effectindicatoren Monitor Agenda Vitaal Platteland

132 *Jaarrapportage 2008.* WOT-04-001 – Koepel

133 *Jaarrapportage 2008.* WOT-04-002 – Onderbouwend Onderzoek

134 *Jaarrapportage 2008.* WOT-04-003 – Advisering Natuur & Milieu

135 *Jaarrapportage 2008.* WOT-04-005 – M-AVP

136 *Jaarrapportage 2008.* WOT-04-006 – Natuurplanbureaufunctie

137 *Jaarrapportage 2008.* WOT-04-007 – Milieuplanbureaufunctie

138 *Jong de, J.J., J. van Os & R.A. Smidt.* Inventarisatie en beheerskosten van landschapselementen

139 *Dirkx, G.H.P., R.W. Verburg & P. van der Wielen.* Tegenkrachten Natuur. Korte verkenning van de weerstand tegen aankopen van landbouwgrond voor natuur

140 *Annual reports for 2008; Programme WOT-04*

141 *Vullings, L.A.E., C. Blok, G. Vonk, M. van Heusden, A. Huisman, J.M. van Linge, S. Keijzer, J. Oldengarm & J.D. Bulens.* Omgaan met digitale nationale beleidskaarten

142 *Vreke, J.,A.L. Gerritsen, R.P. Kranendonk, M. Pleijte, P.H. Kersten & F.J.P. van den Bosch.* Maatlat Government – Governance

143 *Gerritsen, A.L., R.P. Kranendonk, J. Vreke, F.J.P. van den Bosch & M. Pleijte.* Verdrogingsbestrijding in het tijdperk van het Investeringsbudget Landelijk Gebied. Een verslag van casusonderzoek in de provincies Drenthe, Noord-Brabant en Noord-Holland.

144 *Luesink, H.H., P.W. Blokland, M.W. Hoogeveen & J.H. Wisman.* Ammoniakemissie uit de landbouw in 2006 en 2007

145 *Bakker de, H.C.M. & C.S.A. van Koppen.* Draagvlakonderzoek in de steigers. Een voorstudie naar indicatoren om maatschappelijk draagvlak voor natuur en landschap te meten

146 *Goossen, C.M.,* Monitoring recreatiegedrag van Nederlanders in landelijke gebieden. Jaar 2006/2007

147 *Hoefs, R.M.A., J. van Os & T.J.A. Gies.* Kavelruil en Landschap. Een korte verkenning naar ruimtelijke effecten van kavelruil.

148 *Klok, T.L., R. Hille Ris Lambers, P. de Vries, J.E. Tamis & J.W.M. Wijsman.* Quick scan model instruments for marine biodiversity policy.

149 *Spruijt, J., P. Spoorenberg & R. Schreuder.* Milieueffectiviteit en kosten van maatregelen gewasbescherming.

150 *Ehlert, P.A.I. (rapporteur).* Advies Bemonstering bodem voor differentiatie van fosfaatgebruiksnormen.

151 *Wulp van der, N.Y.* Storende elementen in het landschap: welke, waar en voor wie? Bijlage bij WOt-paper 1 – Krassen op het landschap

152 *Oltmer, K., K.H.M. van Bommel, J. Clement, J.J. de Jong, D.P. Rudrum & E.P.A.G. Schouwenberg.* Kosten voor habitattypen in Natura 2000-gebieden. Toepassing van de methode Kosteneffectiviteit natuurbeleid.

153 *Adrichem van, M.H.C., F.G. Wortelboer & G.W.W. Wamelink.* MOVE. Model for terrestrial Vegetation. Version 4.0

154 *Wamelink, G.W.W., R.M. Winkler & F.G. Wortelboer.* User documentation MOVE4 v 1.0

155 *Gies de, T.J.A., L.J.J. Jeurissen, I. Staritsky & A. Bleeker.* Leefomgevingsindicatoren Landelijk gebied. Inventarisatie naar stand van zaken over geurhinder, lichthinder en fijn stof.

156 *Tamminga, S., A.W. Jongbloed, P. Bikker, L. Sebek, C. van Bruggen & O. Oenema.* Actualisatie excretiecijfers landbouwhuisdieren voor forfaits regeling Meststoffenwet

157 *Van der Salm, C., L. .M. Boumans, G.B.M. Heuvelink & T.C. van Leeuwen.* Protocol voor validatie van het nutriëntenemissiemodel STONE op meetgegevens uit het Landelijk Meetnet effecten Mestbeleid

158 *Bouwma, I.M.* Quickscan Natura 2000 en Programma Beheer. Een vergelijking van Programma Beheer met de soorten en habitats van Natura 2000

159 *Gerritsen, A.L., D.A. Kamphorst, T.A. Selnes, M. van Veen, F.J.P.van den Bosch, L. van den Broek, M.E.A. Broekmeyer, J.L.M. Donders, R.J. Fontein, S. van Tol, G.W.W. Wamelink & P. van der Wielen.* Dilemma's en barrières in de praktijk van het natuur- en landschapsbeleid; Achtergronddocument bij Natuurbalans 2009.

160 *Fontein R.J, T.A. de Boer, B. Breman, C.M. Goossen, R.J.H.G. Henkens, J. Luttik & S. de Vries.* Relatie recreatie en natuur; Achtergronddocument bij Natuurbalans 2009

161 *Deneer, J.W. & R. Kruijne.* ***(2010).*** Atmosferische depositie van gewasbeschermingsmiddelen. Een verkenning van de literatuur verschenen na 2003.

162 *Verburg, R.W., M.E. Sanders, G.H.P. Dirkx, B. de Knegt & J.W. Kuhlman.* Natuur, landschap en landelijk gebied. Achtergronddocument bij Natuurbalans 2009.

163 *Doorn van, A.M. & M.P.C.P. Paulissen.* Natuurgericht milieubeleid voor Natura 2000-gebieden in Europees perspectief: een verkenning.

164 *Smidt, R.A., J. van Os & I. Staritsky.* Samenstellen van landelijke kaarten met landschapselementen, grondeigendom en beheer. Technisch achtergronddocument bij de opgeleverde bestanden.

165 *Pouwels, R., R.P.B. Foppen, M.F. Wallis de Vries, R. Jochem, M.J.S.M. Reijnen & A. van Kleunen,* Verkenning LARCH: omgaan met kwaliteit binnen ecologische netwerken.

166 *Born van den, G.J., H.H. Luesink, H.A.C. Verkerk, H.J. Mulder, J.N. Bosma, M.J.C. de Bode & O. Oenema,* Protocol voor monitoring landelijke mestmarkt onder een stelsel van gebruiksnormen, versie 2009.

167 *Dijk, T.A. van, J.J.M. Driessen, P.A.I. Ehlert, P.H. Hotsma, M.H.M.M. Montforts, S.F. Plessius & O. Oenema.* Protocol beoordeling stoffen Meststoffenwet- Versie 2.1

168 *Smits, M.J., M.J. Bogaardt, D. Eaton, A. Karbauskas & P. Roza.* De vermaatschappelijking van het Gemeenschappelijk Landbouwbeleid. Een inventarisatie van visies in Brussel en diverse EU-lidstaten.

169 *Vreke, J. & I.E. Salverda.* Kwaliteit leefomgeving en stedelijk groen.

170 *Hengsdijk, H. & J.W.A. Langeveld.* Yield trends and yield gap analysis of major crops in the World.

171 *Horst, M.M.S. ter & J.G. Groenwold.* Tool to determine the coefficient of variation of DegT50 values of plant protection products in water-sediment systems for different values of the sorption coefficient

172 *Boons-Prins, E., P. Leffelaar, L. Bouman & E. Stehfest (2010)* Grassland simulation with the LPJmL model

173 *Smit, A., O. Oenema & J.W.H. van der Kolk.* Indicatoren Kwaliteit Landelijk Gebied

2010

174 *Boer de, S., M.J. Bogaardt, P.H. Kersten, F.H. Kistenkas, M.G.G. Neven & M. van der Zouwen (.* Zoektocht naar nationale beleidsruimte in de EU-richtlijnen voor het milieu- en natuurbeleid. Een vergelijking van de implementatie van de Vogel- en Habitatrichtlijn, de Kaderrichtlijn Water en de Nitraatrichtlijn in Nederland, Engeland en Noordrijn-Westfalen

175 *Jaarrapportage 2009.* WOT-04-001 – Koepel

176 *Jaarrapportage 2009.* WOT-04-002 – Onderbouwend Onderzoek

177 *Jaarrapportage 2009.* WOT-04-003 – Advisering Natuur & Milieu

178 *Jaarrapportage 2009.* WOT-04-005 – M-AVP

179 *Jaarrapportage 2009.* WOT-04-006 – Natuurplanbureaufunctie

180 *Jaarrapportage 2009.* WOT-04-007 – Milieuplanbureaufunctie

181 *Annual reports for 2009; Programme WOT-04*

182 O*enema, O., P. Bikker, J. van Harn, E.A.A. Smolders, L.B. Sebek, M. van den Berg, E. Stehfest & H. Westhoek.* Quickscan opbrengsten en efficiëntie in de gangbare en biologische akkerbouw, melkveehouderij, varkenshouderij en pluimveehouderij. Deelstudie van project 'Duurzame Eiwitvoorziening'.

183 *Smits, M.J.W., N.B.P. Polman & J. Westerink.* Uitbreidingsmogelijkheden voor groene en blauwe diensten in Nederland; Ervaringen uit het buitenland

184 *Dirkx, G.H.P. (red.).* Quick responsefunctie 2009. Verslag van de werkzaamheden.

185 *Kuhlman, J.W., J. Luijt, J. van Dijk, A.D. Schouten & M.J. Voskuilen.* Grondprijskaarten 1998-2008

186 *Slangen, L.H.G., R.A. Jongeneel, N.B.P. Polman, E. Lianouridis, H. Leneman & M.P.W. Sonneveld.* Rol en betekenis van commissies voor gebiedsgericht beleid.

187 *Temme, A.J.A.M. & P.H. Verburg.* Modelling of intensive and extensive farming in CLUE

188 *Vreke, J.* Financieringsconstructies voor landschap

189 *Slangen, L.H.G.* Economische concepten voor beleidsanalyse van milieu, natuur en landschap

190 *Knotters, M., G.B.M. Heuvelink, T. Hoogland & D.J.J. Walvoort.* A disposition of interpolation techniques

191 *Hoogeveen, M.W., P.W. Blokland, H. van Kernebeek, H.H. Luesink & J.H. Wisman.* Ammoniakemissie uit de landbouw in 1990 en 2005-2008

192 *Beekman, V., A. Pronk & A. de Smet.* De consumptie van dierlijke producten. Ontwikkeling, determinanten, actoren en interventies.

193 *Polman, N.B.P., L.H.G. Slangen, A.T. de Blaeij, J. Vader & J. van Dijk.* Baten van de EHS; De locatie van recreatiebedrijven

194 *Veeneklaas, F.R. & J. Vader.* Demografie in de Natuurverkenning 2011; Bijlage bij WOt-paper 3

195 *Wascher, D.M., M. van Eupen, C.A. Mücher & I.R. Geijzendorffer,* Biodiversity of European Agricultural landscapes. Enhancing a High Nature Value Farmland Indicator

196 *Apeldoorn van, R.C., I.M. Bouwma, A.M. van Doorn, H.S.D. Naeff, R.M.A. Hoefs, B.S. Elbersen & B.J.R. van Rooij.* Natuurgebieden in Europa: bescherming en Financiering

197 *Brus, D.J.,, R. Vasat, G. B. M. Heuvelink, M. Knotters, F. de Vries & D. J. J. Walvoort.* Towards a Soil Information System with quantified accuracy; A prototype for mapping continuous soil properties

198 *Groot, A.M.E.& A.L. Gerritsen, m.m.v. M.H. Borgstein, E.J. Bos & P. van der Wielen,* Verantwoording van de methodiek Achtergronddocument bij 'Kwalitatieve monitor Systeeminnovaties verduurzaming landbouw'

199 *Bos, E.J. & M.H. Borgstein,* Monitoring Gesloten voer-mest kringlopen. Achtergronddocument bij 'Kwalitatieve monitor Systeeminnovaties verduurzaming landbouw'

200 *Kennismarkt 27 april 2010;* Van onderbouwend onderzoek Wageningen UR naar producten Planbureau voor de Leefomgeving.

201 *Wielen van der, P.,* Monitoring Integrale duurzame stallen. Achtergronddocument bij 'Kwalitatieve monitor Systeeminnovaties verduurzaming landbouw'

202 *Groot, A.M.E.& A.L. Gerritsen.* Monitoring Functionele agrobiodiversiteit. Achtergrond-document bij 'Kwalitatieve monitor Systeeminnovaties verduurzaming landbouw'