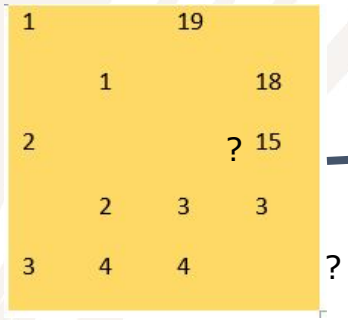
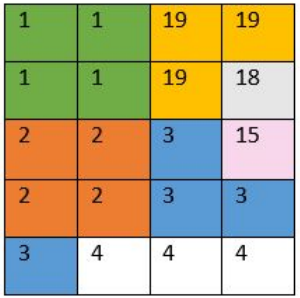

Food and Agriculture
Organization of the
United Nations
AFACI
Asian Food & Agriculture
Cooperation Initiative
Rural Development
Administration
EduSoils
Regression Kriging
Kriging residuals
GLOBAL SOIL
PARTNERSHIP

Interpolation

- In digital soil mapping we mostly work with data in table format and then rasterize this data so that we can make a continuous map
- Interpolation predicts values for cells in a raster from a limited number of sample data points





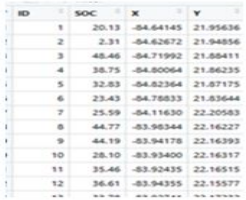
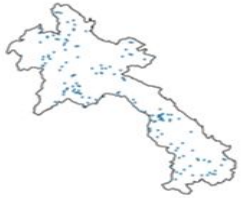


Table with XY coordinates



Point sample data

Modelling/Mapping

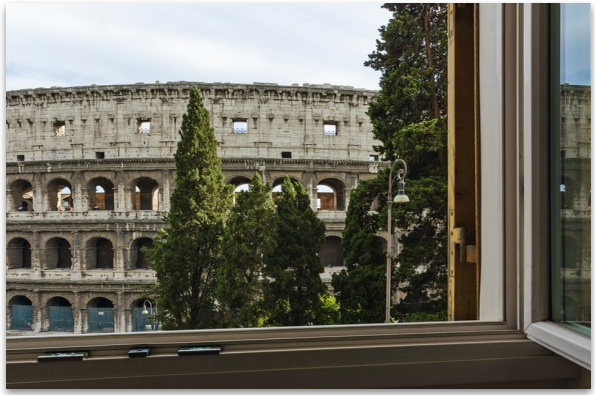
cational platform

Spatial data is special

- Geographer Waldo R. Tobler's stated in the first law of geography:

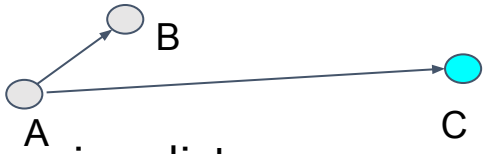
"Everything is related to everything else, but near things are more related than distant things."

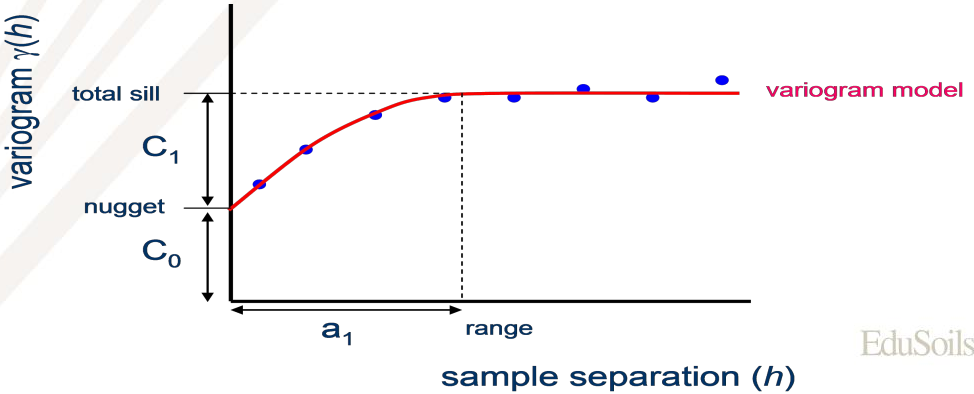
- This is at the bases of spatial autocorrelation
- Housing prices are a good example of this: houses in fancy neighborhoods tend to have similar prices



Spatial data is special

- The Variogram describes the spatial dependance throughout a region
- Taking into consideration spatial autocorrelation there's a good chance that the variance between point A and Point C is greater
- Basic assumption: Variance increases with the increasing distance between points until it becomes constant



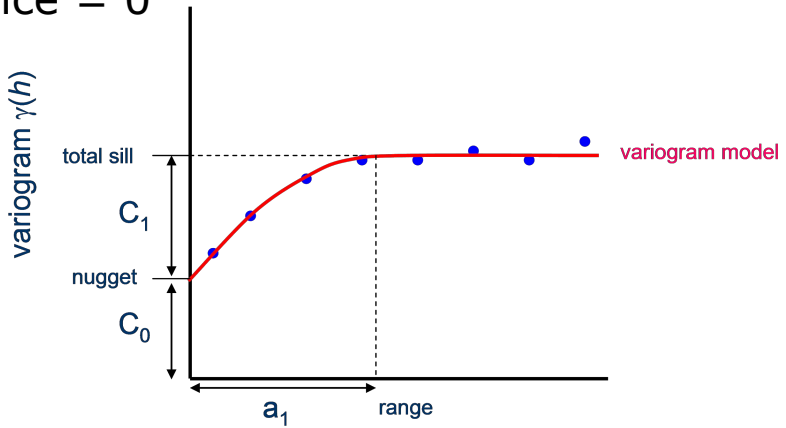


Ordinary Kriging

- In **ordinary** kriging predictions at unsampled locations are made based on a weighted average
- It's an interpolation technique that relies only on point observations of the target variable
- Weights depend on the spatial autocorrelation structure of the variable
- The weights are chosen such that the prediction error variance is minimized
- To make a prediction with the kriging interpolation method, two tasks are necessary:
    - Uncover the dependency rules by creating a variogram and covariance functions to estimate the statistical dependence (called spatial autocorrelation) values that depend on the model of autocorrelation (fitting a model).
    - Make the predictions

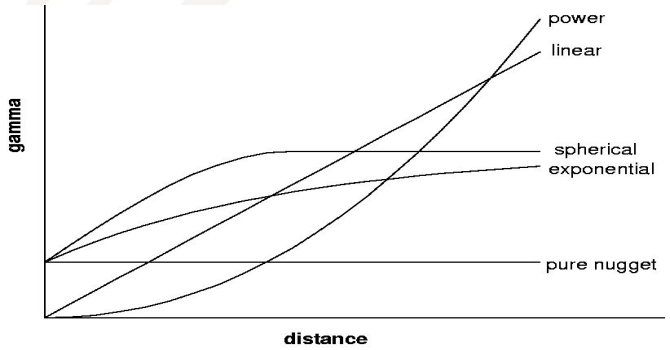
Kriging

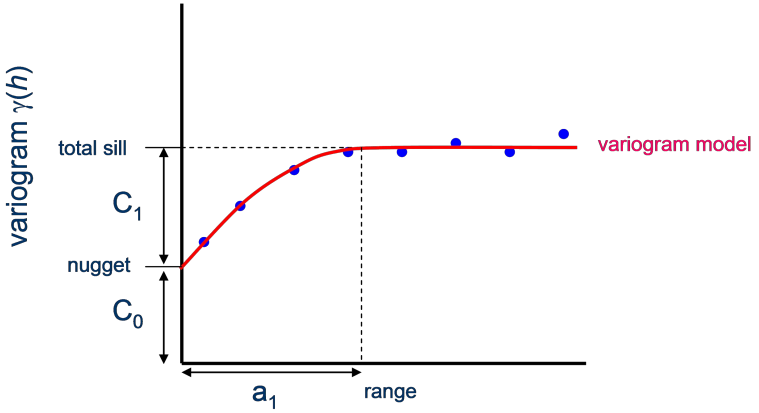
- A variogram has three components:
  1. *Sill* → describes the total variance of the process
  2. *Range* → distance beyond which there's not spatial autocorrelation
  3. *Nugget* → variance at distance = 0



Kriging

- To make predictions at unknown locations a curve needs to be fitted to the variogram
- Abstractly, this is similar to regression analysis, in which a continuous line or curve is fitted to the data points
- There are different variogram-models that can be selected to better fit the data

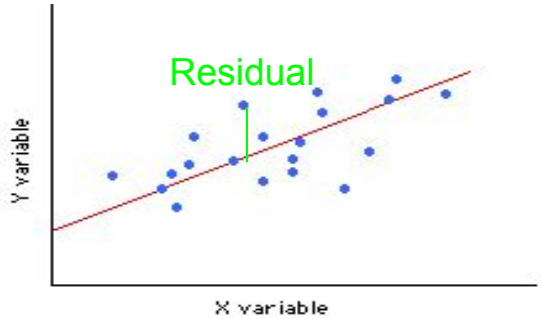




EduSoils | e-learning soil educational platform

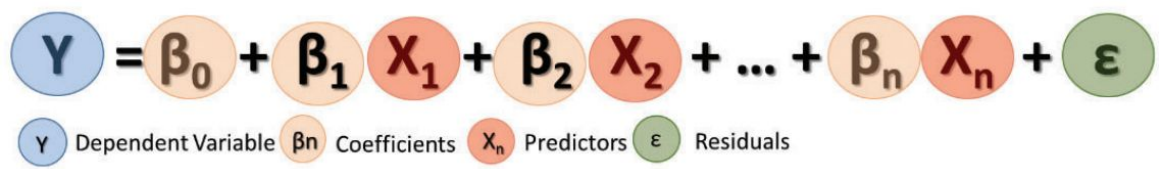
Regression Kriging

- Regression kriging is a hybrid interpolation technique which combines two conceptually different approaches to modelling and mapping spatial variability:
  - interpolation based on regression of the target variable based on spatially auxiliary information (a.k.a covariates)
  - interpolation relying only on point observations of the target variable
- It combines a regression of the dependent variable (target variable) on predictors (the environmental covariates) with kriging of the prediction residuals



Regression Kriging

- Instead of assuming the errors ε(s) are independent, you model them to be autocorrelated
- The residuals show spatial autocorrelation → Semivariogram



Kriging

- Regression kriging is advantageous compared to ordinary kriging when:
  - We don't have that many data points to predict the target variable in our study area
  - We know that there is a strong relationship between the target variable and the independent ones

Regression Kriging

- Kriging residuals of a linear model can be interpreted as creating a correction map that can be combined to the map created with the linear regression in order to take in account the autocorrelation of errors
- The autocorrelation of errors can be interpreted as the way our model tends to overpredict or underpredict our target variable depending on the location

Observed values:

5	?	?	5
?	?	?	?
?	2	?	?
2	?	?	3

Linear Model Prediction:

6	2	4	3
2	3	2	7
3	3	7	7
2	7	7	5

Residual =Predicted - Observed

R1	1
R2	-2
R3	1
R4	0
R5	2

Interpolation of residuals through kriging:

1	0	-1	-2
0.5	0	-0.5	-1
0.5	1	0	0.5
0	1	1.5	2

Regression Kriging

- The final map is created by correcting the predictions of the linear model based on the kriged residuals

Interpolation of residuals through kriging:

1	0	-1	-2
0.5	0	-0.5	-1
0.5	1	0	0.5
0	1	1.5	2

Linear Model Prediction:

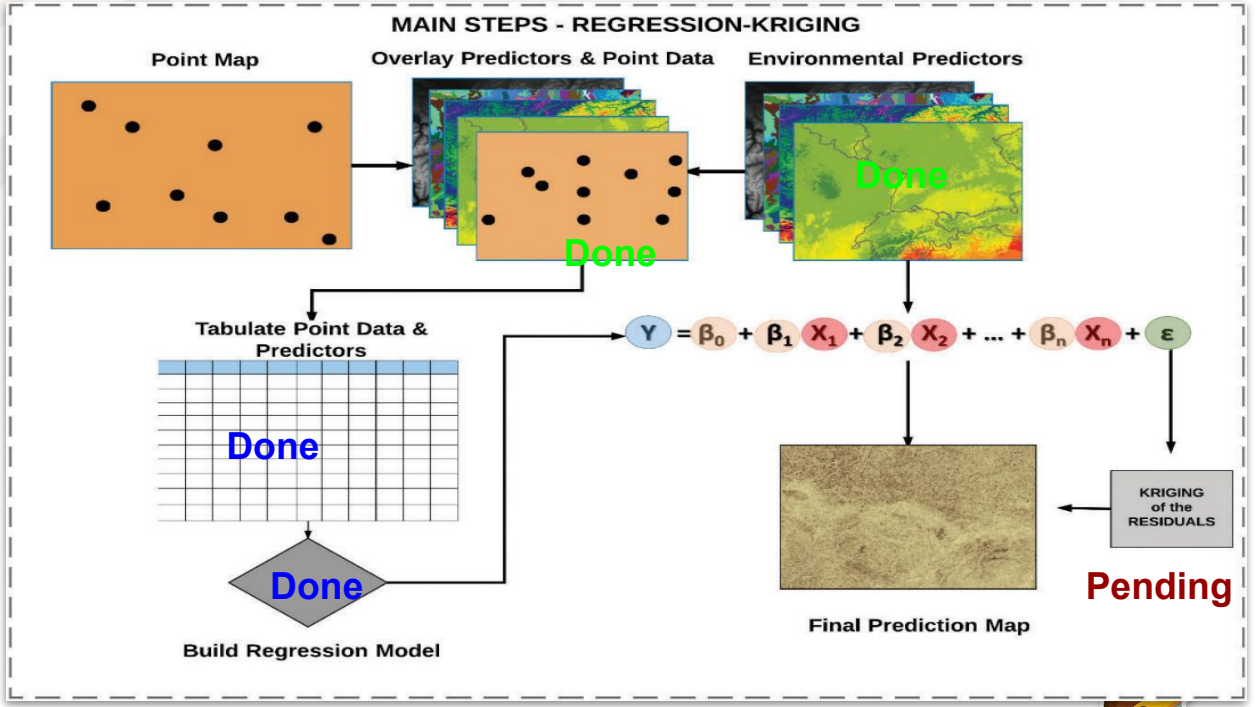
6	2	4	3
2	3	2	7
3	3	7	7
2	7	7	5

Regression Kriging map:

5	2	5	5
1.5	3	2.5	8
2.5	2	7	6.5
2	6	5.5	3

Regression Kriging Workflow

- Extract the values of the covariates and target variables into the single table
- Fit a multiple regression model using the previously created table
- Kriging: Model the residuals taking into account spatial correlation



Regression Kriging

During this part of the workshop we will:

1. Promote covariates to SpatialGridDataframe
2. Compute and explore a variogram
3. Create and tweak the variogram model to create the best interpolation of the residuals
4. Run the regression kriging model

SpatialGridDataframe

- Unlike the SpatialPointDataframe class object, this one has a defined raster resolution
- Since kriging is based on measured distances, all data should be converted to a projected coordinate systems (to meters)

```
# Project point data
coordinates(dat) <- ~ X + Y
proj4string(dat) = CRS("+init=epsg:4326") # WGS84
dat <- spTransform(dat, CRS("+init=epsg:6204")) # Macedonian state #coordinate
system
covs <- projectRaster(covs, crs = CRS("+init=epsg:6204"),
                      method='ngb')
```

SpatialGridDataframe

- Unlike the SpatialPointDataframe class object, this one has a defined raster resolution

```
# Promote covariates to spatial grid dataframe (it can take a lot #of memory)
covs.sp <- as(covs, "SpatialGridDataFrame")
covs.sp@grid

# LandCover and soilmap are categorical variables, they need to be #'factor' type
covs.sp$LandCover <-as.factor(covs.sp$LandCover)
covs.sp$soilmap <-as.factor(covs.sp$soilmap)
```

Semivariogram

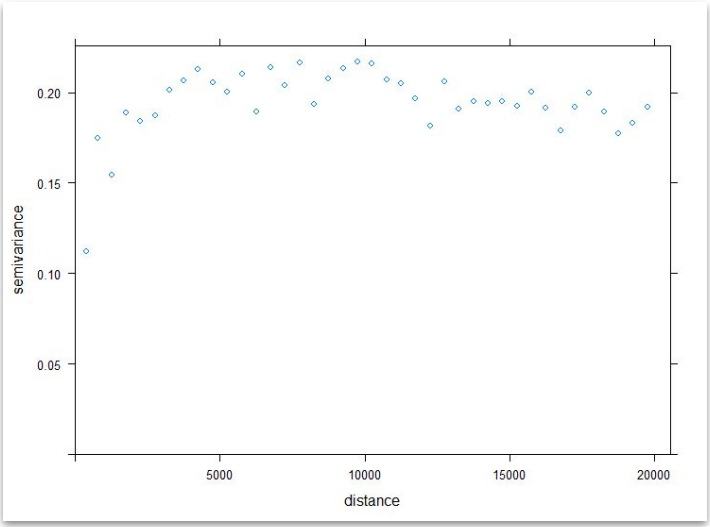
- To compute a variogram to check if the residual exhibit spatial autocorrelation we're going to use the *gstat* package
- The gstat() function creates an object that contains the necessary information to perform regression kriging
- In it we're going to input our previously created linear model

```
library(gstat)
# Define gstat object
gstat_data <- gstat(formula = as.formula(model.MLR.step$call$formula), data = dat)
```

Semivariogram

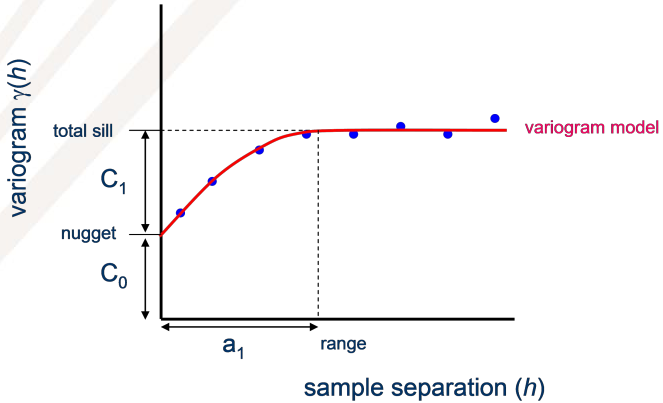
- Let’s plot the residuals in a semivariogram

```
#Compute and explore an experimental
#semivariogram
vario <- variogram(gstat_data, cutoff=20000,
          width=500)
plot(vario, plot.nu=FALSE)
```



Semivariogram

- In order to predict the distribution of the residuals at locations without observed data will will create a first semivariogram model
- We can tweak the nugget, sill and range

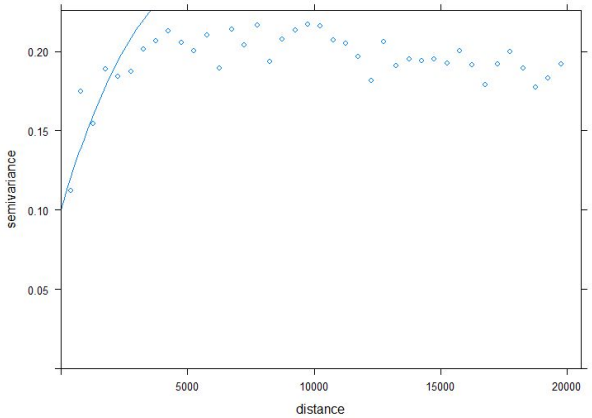


1. *Sill* → describes the total variance of the process
2. *Range* → distance beyond which there's not spatial autocorrelation
3. *Nugget* → variance at distance = 0

Semivariogram

- In order to predict the distribution of the residuals at locations without observed data will will create a first semivariogram model
- Abstractly, this is similar to regression analysis, in which a continuous line or curve is fitted to the data points

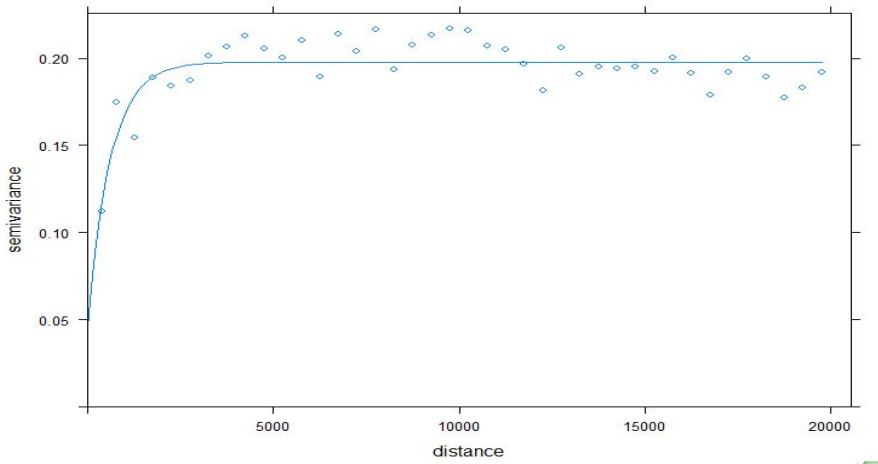
```
# Define initial semivariogram model
vario_mod <- vgm(nugget = 0.1, psill =
 0.20, range = 5000, model = "Ste")
plot(vario, vario_mod)
```



Semivariogram

- Now we're going to use the function fit.variogram() to fit the best model to predict the residuals
- We had to put some initial values, for the algorithm to work

```
# Define the random numbers table
#(to get reproducible result)
#set.seed(12042019)
# Fit semivariogram model for kriging
vario_mod <- fit.variogram(vario,
vario_mod, fit.method=7)
plot(vario, vario_mod)
vario_mod
```



Other packages for regression kriging

- The package *automap* fits the semivariogram model automatically to the residuals (no need to create a semivariogram graph and to tune the model)
- This is the package used in the Soil Organic Carbon Mapping Cookbook 2nd Edition - FAO

```
# RK model
library(automap)

# Run regression-kriging prediction.
# This step can take hours!
OCS.krige <- autoKrige(formula =
                         as.formula(model.MLR.step$call$formula),
                       input_data = dat,
                       new_data = covs.sp,
                       verbose = TRUE,
                       block = c(1000, 1000))

OCS.krige
```

Regression Kriging

- Now we will predict our target variable using the krige() function
- The formula used for the linear mode part of the final regression kriging map is the previously created step-wise linear model
- The locations are defined by the XY coordinates in the object dat
- The model used to krig the residual is our previously created semivariogram model

```
# Make a prediction across all Macedonia using Regression Kriging #model
pred_gstat <- krige(formula = as.formula(model.MLR.step$call$formula),
                    locations = dat,
                    newdata = covs.sp,
                    model = vario_mod,
                    debug.level = -1)
```

Plot our predictions

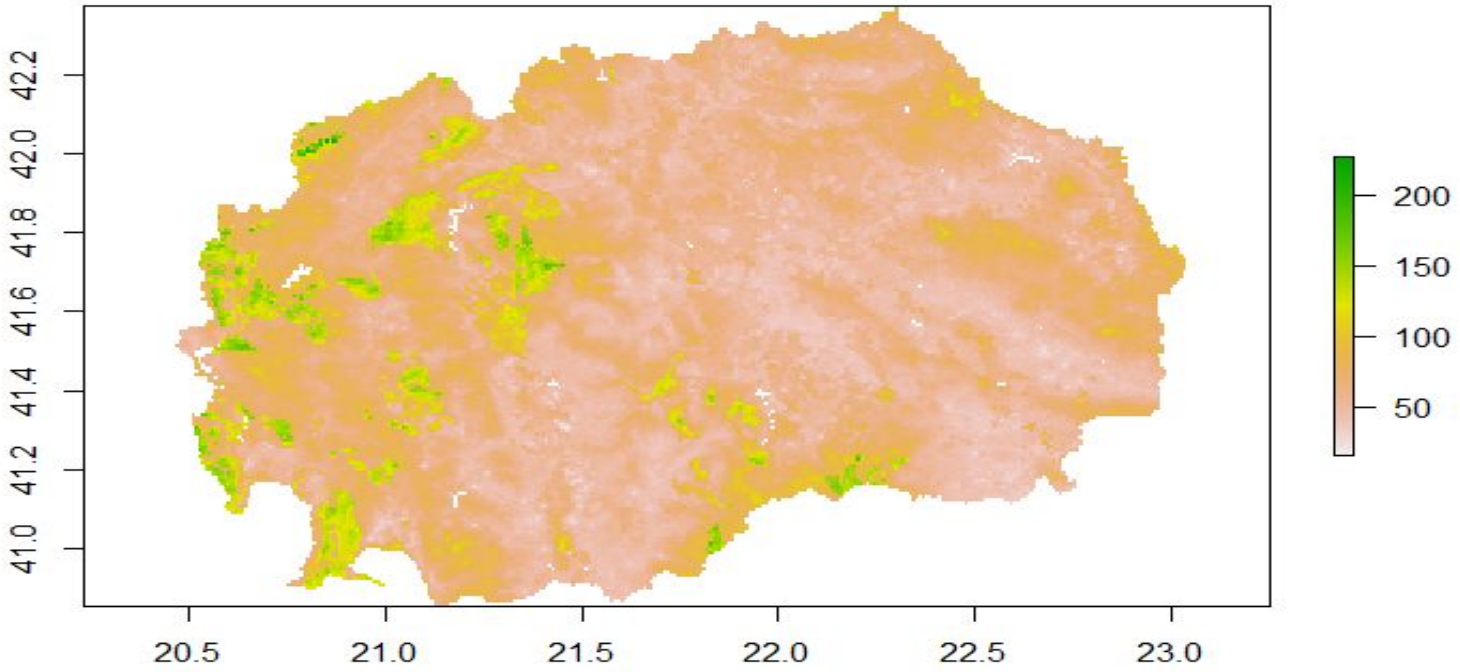
- Our model was made to predict the log of the target variable therefore to we need to back transform it

```
# Back transform predictions log transformed
RKpred <- exp(raster(pred_gstat))

# Back transform the coordinate system to WGS 84, using MLR #prediction as a template
RKpred <- projectRaster(from = RKpred, to = pred, method = "ngb")

# Explore and save the result as a tiff file
plot(RKpred)
writeRaster(RKpred,'02-Outputs/Final Maps/MKD_OCS_RK.tif', overwrite=TRUE)
```

Regression Kriging

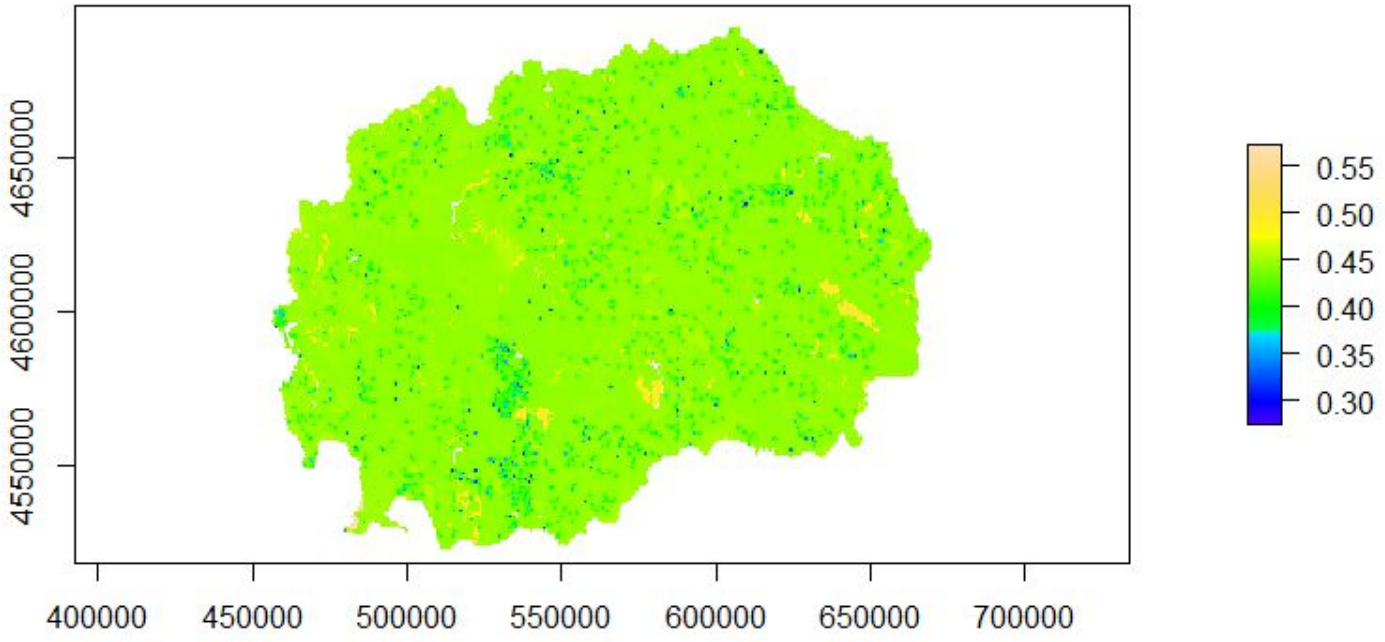


Uncertainty (see lecture 8.Uncertainty and Validation)

- We can display the model's prediction variance with a standard deviation map

```
# Make an uncertainty estimation as a map of standard deviations
# Standard deviation is the square root of kriging variance
RKsd <- sqrt(raster(pred_gstat, layer='var1.var'))
# If data was not log-transformed, than standard deviation map is in same units as
the data (t/ha) and can be reported as uncertainty map
plot(RKsd, col=  topo.colors(255))
# But in our case it was calculated on log-transformed data and cannot be back
transformed to t/ha
writeRaster(RKsd_log,'02-Outputs/MKD_OCS_RK_sd.tif', overwrite=TRUE)
```

Uncertainty (see lecture 8.Uncertainty and Validation)



Uncertainty (see lecture 8.Uncertainty and Validation)

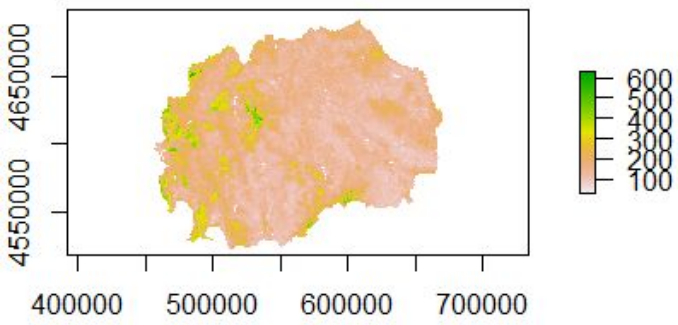
- We can calculate confidence interval from the standard deviations

```
# Calculate 95% confidence interval on
# log-transformed data as pred +-2*sd
RK_ci_high<-raster(pred_gstat)+2*RKsd_log
RK_ci_low<-raster(pred_gstat)-2*RKsd_log
# Confidence interval limits can be back transformed to t/ha
RK_ci_high<-exp(RK_ci_high)
RK_ci_low<-exp(RK_ci_low)
plot(RK_ci_high)
plot(RK_ci_low)
# Export limits of confidence interval as measures of uncertainty
writeRaster(RK_ci_high,'02-Outputs/MKD_OCS_RK_ci95_high.tif', overwrite=TRUE)
writeRaster(RK_ci_low,'02-Outputs/MKD_OCS_RK_ci95_low.tif', overwrite=TRUE)
```

Uncertainty (see lecture 8.Uncertainty and Validation)

Upper limit:

95% confidence interval



Lower limit: