Global prediction of soil saturated hydraulic conductivity using random forest in a Covariate-based Geo Transfer Functions (CoGTF) framework

Surya Gupta, Tom Hengl, Peter Lehmann, Sara Bonetti, Andreas Papritz, Dani Or

Abstract:

Soil saturated hydraulic conductivity (Ksat) is one of the prominent soil hydraulic properties used in the modeling of land surface processes. Ksat is often derived using limited dataset and soil basic properties likely soil texture, bulk density) by means pedotransfer functions (PTFs). We propose here an integrated Predictive Soil Modeling (PSM) framework where soil variables are combined with RS-based covariates using the Random Forest method. We refer to this approach as the “Covariate-based Geo Transfer Functions” (CoGTF). Here, the objective of this report to show the methods used to develop the CoGTF with R code and stepwise description.

```r
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```r
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
library(ranger)
```

```
##
## Attaching package: 'ranger'
```

```
## The following object is masked from 'package:randomForest':
##
##     importance
```

```r
library(mlr)
```

```
## Loading required package: ParamHelpers
```

```
## 'mlr' is in maintenance mode since July 2019. Future development
## efforts will go into its successor 'mlr3' (<https://mlr3.mlr-org.com>).
```

```
##
## Attaching package: 'mlr'
```

```
## The following object is masked from 'package:caret':
##
##     train
```

```r
library(tibble)
library(raster)
```

```
## Loading required package: sp
```

```
##
## Attaching package: 'raster'
```

```
## The following object is masked from 'package:mlr':
##
##     resample
```

```
## The following object is masked from 'package:ParamHelpers':
##
##     getValues
```

```
library(sp)
library(rgdal)
```

```
## rgdal: version: 1.5-12, (SVN revision 1018)
## Geospatial Data Abstraction Library extensions to R successfully loaded
## Loaded GDAL runtime: GDAL 3.0.4, released 2020/01/28
## Path to GDAL shared files: C:/Users/guptasu.D/Documents/R/win-library/3.6/rgdal/gdal
## GDAL binary built with GEOS: TRUE
## Loaded PROJ runtime: Rel. 6.3.1, February 10th, 2020, [PJ_VERSION: 631]
## Path to PROJ shared files: C:/Users/guptasu.D/Documents/R/win-library/3.6/rgdal/proj
## Linking to sp version:1.4-2
## To mute warnings of possible GDAL/OSR exportToProj4() degradation,
## use options("rgdal_show_exportToProj4_warnings"="none") before loading rgdal.
```

```
library(hexbin)
library(lattice)
library(RColorBrewer)
library(viridis)
```

```
## Loading required package: viridisLite
```

```
library(Metrics)
```

```
##
## Attaching package: 'Metrics'
```

```
## The following objects are masked from 'package:caret':
##
##     precision, recall
```

```
tt= read.csv("E:/Ksat_final_dataset.csv")
nrow(tt)
```

```
## [1] 6814
```

```
source("E:/OpenLandMap/R/saveRDS_functions.R")
source("E:/OpenLandMap/R/LandGIS_functions.R")

## saveRDS_functions.R and LandGIS_functions.R available at https://github.com/Envirometrix/LandGISmaps/tree
/477460d1d0099646c508f65e68769b9edf050ce8/functions

## 3D modeling (see Hengl, T., & MacMillan, R. A. (2019). Predictive soil mapping with R. Lulu. com.)

dfs <- hor2xyd(tt, U="hzn_top", L="hzn_bot")

I.vars = make.names(unique(unlist(sapply(c("s.no","clm_", "dtm_", "lcv", "veg_", "olm_c", "olm_s", "olm_bd"
,"DEPTH"), function(i){names(dfs)[grep(i, names(dfs))]}))))

t.vars = c("log_ksat")
sel.n <- c(t.vars,I.vars)
sel.r <- complete.cases(dfs[,sel.n])
PTF_temp2 <- dfs[sel.r,sel.n]


## Hypertunning parameter

#control <- trainControl(method="repeatedcv", number=5, repeats=3, search="random")

#set.seed(1)
#rf_random <-train(log_ksat~.,data=PTF_temp2, method="rf", metric=metric, tuneLength=15, trControl=control)
#print(rf_random)
#plot(rf_random)


## In hypertunning, we selected mtry = 6 because of best RMSE


## Manual Cross-validation and repeated for 3 times

set.seed(4)
ff<- split(PTF_temp2, sample(1:5, nrow(PTF_temp2), replace=T))

df1<- ff$`1`
df2<- ff$`2`
df3<- ff$`3`
df4<- ff$`4`
df5<- ff$`5`

Train1<- rbind(df1, df2, df3, df4)

Train2<- rbind (df2, df3, df4, df5)

Train3<- rbind(df3, df4, df5, df1)

Train4<- rbind(df4, df5, df1,df2)

Train5<- rbind(df5, df1,df2,df3)

grid <- list.files("E:/maps_tests/new_layers/layers_RS/" , pattern = "*.tif$")
All_cov <- raster::stack(paste0("E:/maps_tests/new_layers/layers_RS/", grid))

set.seed(2)
fm.ksat <- as.formula(paste("log_ksat~ ",paste(names(All_cov), collapse = "+")))
fm.ksat
```

```
## log_ksat ~ clm_bioclim.var_chelsa.1_m_1km_s0..0cm_1979..2013_v1.0 +
##     clm_bioclim.var_chelsa.12_m_1km_s0..0cm_1979..2013_v1.0 +
##     clm_bioclim.var_chelsa.13_m_1km_s0..0cm_1979..2013_v1.0 +
##     clm_bioclim.var_chelsa.14_m_1km_s0..0cm_1979..2013_v1.0 +
##     clm_bioclim.var_chelsa.4_m_1km_s0..0cm_1979..2013_v1.0 +
##     clm_bioclim.var_chelsa.5_m_1km_s0..0cm_1979..2013_v1.0 +
##     clm_bioclim.var_chelsa.6_m_1km_s0..0cm_1979..2013_v1.0 +
##     clm_cloud.fraction_earthenv.modis.annual_m_1km_s0..0cm_2000..2015_v1.0 +
##     clm_diffuse.irradiation_solar.atlas.kwhm2.100_m_1km_s0..0cm_2016_v1 +
##     clm_direct.irradiation_solar.atlas.kwhm2.10_m_1km_s0..0cm_2016_v1 +
##     clm_lst_mod11a2.annual.day_m_1km_s0..0cm_2000..2017_v1.0 +
##     clm_lst_mod11a2.annual.day_sd_1km_s0..0cm_2000..2017_v1.0 +
##     clm_precipitation_sm2rain.annual_m_1km_s0..0cm_2007..2018_v0.2 +
##     DEPTH + dtm_aspect.cosine_merit.dem_m_250m_s0..0cm_2018_v1.0 +
##     dtm_elevation_merit.dem_m_250m_s0..0cm_2017_v1.0 + dtm_lithology_usgs.ecotapestry.acid.plutonics_p_25
0m_s0..0cm_2014_v1.0 +
##     dtm_slope_merit.dem_m_1km_s0..0cm_2017_v1.0 + dtm_twi_merit.dem_m_1km_s0..0cm_2017_v1.0 +
##     lcv_landsat.nir_wri.forestwatch_m_250m_s0..0cm_2014_v1.0 +
##     lcv_landsat.red_wri.forestwatch_m_250m_s0..0cm_2018_v1.2 +
##     lcv_landsat.swir2_wri.forestwatch_m_250m_s0..0cm_2018_v1.2 +
##     lcv_snow_probav.lc100_p_250m_s0..0cm_2017_v1.0 + lcv_wetlands.regularly.flooded_upmc.wtd_p_250m_b0..2
00cm_2010..2015_v1.0 +
##     olm_bd + olm_clay + olm_sand + veg_fapar_proba.v.annual_d_250m_s0..0cm_2014..2017_v1.0
```

```
set.seed(2)
rm.ksat <- Train1[complete.cases(Train1[,all.vars(fm.ksat)]),]
m.ksat <- ranger(fm.ksat, rm.ksat, num.trees=200, mtry=6, quantreg = TRUE)
m.ksat
```

```
## Ranger result
##
## Call:
##  ranger(fm.ksat, rm.ksat, num.trees = 200, mtry = 6, quantreg = TRUE)
##
## Type:                             Regression
## Number of trees:                  200
## Sample size:                      14317
## Number of independent variables:  28
## Mtry:                             6
## Target node size:                 5
## Variable importance mode:         none
## Splitrule:                        variance
## OOB prediction error (MSE):       0.532218
## R squared (OOB):                  0.6578845
```

```
df5$prediction<- predict(m.ksat,df5)$predictions

## Ist_part is computed

rm.ksat1 <- Train2[complete.cases(Train2[,all.vars(fm.ksat)]),]
m.ksat1 <- ranger(fm.ksat, rm.ksat1, num.trees=200, mtry=6, quantreg = TRUE)
m.ksat1
```

```
## Ranger result
##
## Call:
##  ranger(fm.ksat, rm.ksat1, num.trees = 200, mtry = 6, quantreg = TRUE)
##
## Type:                             Regression
## Number of trees:                  200
## Sample size:                      14414
## Number of independent variables:  28
## Mtry:                             6
## Target node size:                 5
## Variable importance mode:         none
## Splitrule:                        variance
## OOB prediction error (MSE):       0.5429787
## R squared (OOB):                  0.6592161
```

```r
df1$prediction<- predict(m.ksat1,df1)$predictions

## 2nd_part is computed
rm.ksat2 <- Train3[complete.cases(Train3[,all.vars(fm.ksat)]),]
m.ksat2 <- ranger(fm.ksat, rm.ksat2, num.trees=200, mtry=6, quantreg = TRUE)
m.ksat2
```

```
## Ranger result
##
## Call:
##  ranger(fm.ksat, rm.ksat2, num.trees = 200, mtry = 6, quantreg = TRUE)
##
## Type:                             Regression
## Number of trees:                  200
## Sample size:                      14455
## Number of independent variables:  28
## Mtry:                             6
## Target node size:                 5
## Variable importance mode:         none
## Splitrule:                        variance
## OOB prediction error (MSE):       0.541844
## R squared (OOB):                  0.662298
```

```r
df2$prediction<- predict(m.ksat2,df2)$predictions

## 3rd_part is computed

rm.ksat3 <- Train4[complete.cases(Train4[,all.vars(fm.ksat)]),]
m.ksat3 <- ranger(fm.ksat, rm.ksat3, num.trees=200, mtry=6, quantreg = TRUE)
m.ksat3
```

```
## Ranger result
##
## Call:
##  ranger(fm.ksat, rm.ksat3, num.trees = 200, mtry = 6, quantreg = TRUE)
##
## Type:                             Regression
## Number of trees:                  200
## Sample size:                      14421
## Number of independent variables:  28
## Mtry:                             6
## Target node size:                 5
## Variable importance mode:         none
## Splitrule:                        variance
## OOB prediction error (MSE):       0.5372147
## R squared (OOB):                  0.6622029
```

```r
df3$prediction<- predict(m.ksat3,df3)$predictions

## 4th_part is computed
rm.ksat4 <- Train5[complete.cases(Train5[,all.vars(fm.ksat)]),]
m.ksat4 <- ranger(fm.ksat, rm.ksat4, num.trees=200, mtry=6, quantreg = TRUE)
m.ksat4
```

```
## Ranger result
##
## Call:
##  ranger(fm.ksat, rm.ksat4, num.trees = 200, mtry = 6, quantreg = TRUE)
##
## Type:                             Regression
## Number of trees:                  200
## Sample size:                      14309
## Number of independent variables:  28
## Mtry:                             6
## Target node size:                 5
## Variable importance mode:         none
## Splitrule:                        variance
## OOB prediction error (MSE):       0.5268998
## R squared (OOB):                  0.6644586
```

```
df4$prediction<- predict(m.ksat4,df4)$predictions

Final_data<- rbind(df1,df2,df3,df4,df5)

dd<- aggregate(Final_data[, 1:31], list(Final_data$s.no), mean)


ll<- lm(dd$prediction~ dd$log_ksat)

RMSE(dd$prediction,dd$log_ksat)
```

```
## [1] 0.7212678
```

```
summary(ll)
```

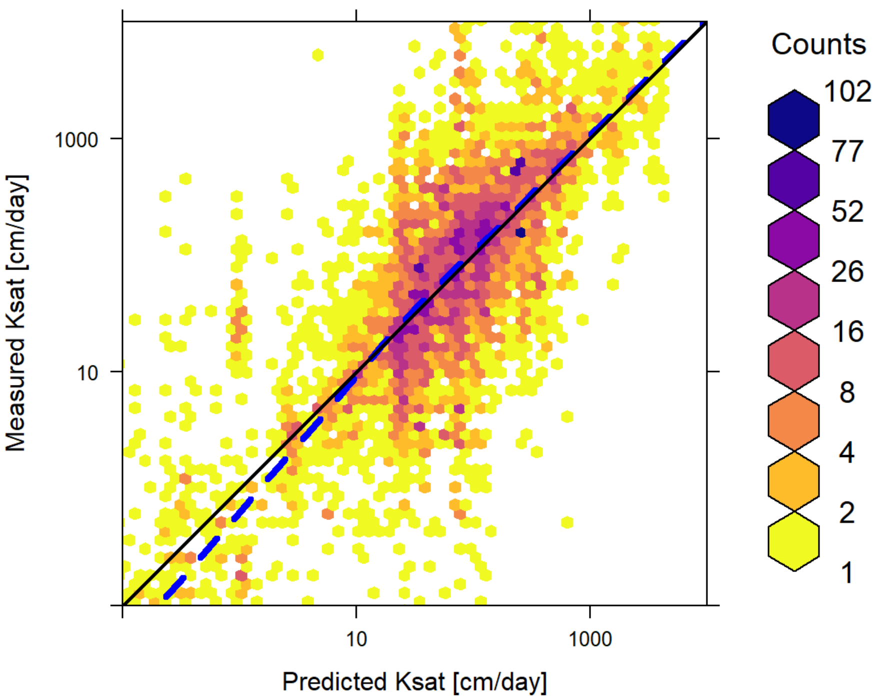
```
##
## Call:
## lm(formula = dd$prediction ~ dd$log_ksat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.8821 -0.2715  0.0296  0.3210  3.6810
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.565895   0.011967   47.29   <2e-16 ***
## dd$log_ksat 0.658594   0.005751  114.52   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5836 on 6679 degrees of freedom
## Multiple R-squared:  0.6626, Adjusted R-squared:  0.6625
## F-statistic: 1.311e+04 on 1 and 6679 DF,  p-value: < 2.2e-16
```

```
dd$log_ksat1<- 10^dd$log_ksat
dd$prediction1<- 10^dd$prediction

ccc = DescTools::CCC(dd$prediction,dd$log_ksat, ci = "z-transform", conf.level = 0.95, na.rm=TRUE)$rho.c
ccc
```

```
##       est    lwr.ci    upr.ci
## 1 0.79605 0.7876551 0.8041494
```

```
hexbinplot(log_ksat1~ prediction1,
           panel = function(x, y, ...){
             panel.hexbinplot(x, y, ...)
             panel.loess(x, y,span = 2/3, col.line = "blue",type="l", lty=2, lwd = 4)
             panel.abline(c(0, 1),lwd = 2)
           },
           data = dd,xlab = "Predicted Ksat [cm/day]", ylab = "Measured Ksat [cm/day]",cex.axis = 4, aspect=
"1", xbins=80, colramp = function(n) {viridis (8,  alpha = 1, begin = 0, end = 1, direction = -1,option = "
C")},xlim=c(0.1,10000), ylim=c(0.1,10000),
           scales=list(
             x = list(log = 10, equispaced.log = FALSE),
             y = list(log = 10, equispaced.log = FALSE)
           ),
           font.lab= 6, cex.labels = 1.2,font.axis = 2,colorcut=c(0,0.01,0.03,0.07,0.15,0.25,0.5,0.75,1) )
```




```
##Fitting final model

rm.ksat <- PTF_temp2[complete.cases(PTF_temp2[,all.vars(fm.ksat)]),]
m.ksat <- ranger(fm.ksat, rm.ksat, num.trees=200, mtry=6, quantreg = TRUE)
m.ksat
```

```
## Ranger result
##
## Call:
##  ranger(fm.ksat, rm.ksat, num.trees = 200, mtry = 6, quantreg = TRUE)
##
## Type:                             Regression
## Number of trees:                  200
## Sample size:                      17979
## Number of independent variables:  28
## Mtry:                             6
## Target node size:                 5
## Variable importance mode:         none
## Splitrule:                        variance
## OOB prediction error (MSE):       0.5286138
## R squared (OOB):                  0.6660434
```

```
##Importance variable
m.ksat <- randomForest(fm.ksat, rm.ksat, num.trees=200, mtry=6, quantreg = TRUE)

varImpPlot(m.ksat, sort=TRUE, n.var=min(12, nrow(m.ksat$importance)))
```


```
## Then Produced the final CoGTF map

#p2 = predict(All_cov,m.ksat, progress='window',type = "response",fun = function(model, ...) predict(model, ...)$predictions)

#writeRaster(p2, "/home/step/data/OpenLandMap/Final_selected_covariates/New_raster_layer/Final_RF_0cm.tif")
```