Pedotransfer functions (PTFs) developed for Ksat using Multivariate polynomial regeression (MPR) and Random forest (RF)

Surya Gupta, Tomislav Hengl, Peter Lehmann, Sara Bonetti, Dani Or

Abstract:

We prepared a comprehensive global compilation of measured Ksat training point data (N= 13,267) called “SoilKsatDB” by importing, quality controlling and standardizing tabular data from existing soil profile databases and legacy reports. The SoilKsatDB was used to develop the pedotransfer functions (PTFs) for temperate climate region and lab-based measured soil samples. These PTFs were applied to tropical climate region and field-based measurements, respectively to evaluate the suitability for other regions. Here, the objective of this report to show the methods used to develop the PTFs with R code and stepwise description.

```
#Loading libraries

library(Metrics)
library(raster)
```

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

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

```
## rgdal: version: 1.4-8, (SVN revision 845)
##  Geospatial Data Abstraction Library extensions to R successfully loaded
##  Loaded GDAL runtime: GDAL 2.2.3, released 2017/11/20
##  Path to GDAL shared files: C:/Users/guptasu.D/Documents/R/R-3.6.3/library/rgdal/gd
al
##  GDAL binary built with GEOS: TRUE
##  Loaded PROJ.4 runtime: Rel. 4.9.3, 15 August 2016, [PJ_VERSION: 493]
##  Path to PROJ.4 shared files: C:/Users/guptasu.D/Documents/R/R-3.6.3/library/rgdal/
proj
##  Linking to sp version: 1.4-1
```

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

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

```
library(ranger)
library(mlr)
```

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

```
##
## Attaching package: 'ParamHelpers'
```

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

```
## '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:raster':
##
##     resample
```

#Temperate climate region Ksat values PTF

Here, we loaded the temperate climate region soil samples. The soil samples were converted into log scale and then selected points with available three soil basic properties (sand, clay and bulk density).

```
rm.hydroprops = read.csv("E:/maps_tests/All_regions_ksat/Temperate_5_04_2020.csv")

rm.hydroprops$log_ksat = signif(log10( rowMeans(rm.hydroprops[,c("ksat_lab","ksat_fiel
d")], na.rm=TRUE) ), 4)

soil_temp_clay <- rm.hydroprops[!is.na(rm.hydroprops$clay_tot_p),]

soil_temp_clay_bd<- soil_temp_clay[!is.na(soil_temp_clay$db_od),]
```

The ksat values were divided into training and testing datasets. We took 80% samples for developing the PTF and 20% for testing the model. Further multivariate polynomial regression (MPR) was fitted.

```
# Randomly split the dataset into training and testing datasets

sample_size = floor(0.80*nrow(soil_temp_clay_bd))

set.seed(777)

picked = sample(seq_len(nrow(soil_temp_clay_bd)),size = sample_size)

development =soil_temp_clay_bd[picked,]

nrow(development)
```

```
## [1] 6666
```

```
holdout =soil_temp_clay_bd[-picked,]

nrow(holdout)
```

```
## [1] 1667
```

```
##Fitting MPR model with degree 2

model_temp_ksat<- lm(log_ksat ~ poly(db_od,clay_tot_p, sand_tot_p, degree=2, raw=TRU
E), data =development )

summary(model_temp_ksat)
```

```
##
## Call:
## lm(formula = log_ksat ~ poly(db_od, clay_tot_p, sand_tot_p, degree = 2,
##     raw = TRUE), data = development)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.0153 -0.3059  0.0518  0.3707  3.1768
##
## Coefficients:
##                                                                   Estimate
## (Intercept)                                                      1.910e+00
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)1.0.0  1.237e+00
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)2.0.0 -8.892e-01
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.1.0  1.011e-02
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)1.1.0 -2.195e-02
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.2.0  1.104e-05
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.0.1  3.424e-03
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)1.0.1  1.054e-03
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.1.1 -2.403e-04
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.0.2  6.097e-05
##                                                                 Std. Error
## (Intercept)                                                      3.925e-01
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)1.0.0  3.816e-01
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)2.0.0  1.282e-01
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.1.0  8.906e-03
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)1.1.0  4.990e-03
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.2.0  6.654e-05
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.0.1  5.151e-03
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)1.0.1  2.826e-03
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.1.1  7.228e-05
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.0.2  2.795e-05
##                                                                 t value
## (Intercept)                                                       4.867
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)1.0.0   3.241
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)2.0.0  -6.935
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.1.0   1.135
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)1.1.0  -4.399
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.2.0   0.166
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.0.1   0.665
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)1.0.1   0.373
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.1.1  -3.325
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.0.2   2.181
##                                                                 Pr(>|t|)
## (Intercept)                                                     1.16e-06 ***
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)1.0.0  0.00120 **
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)2.0.0 4.43e-12 ***
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.1.0  0.25624
```

```
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)1.1.0 1.10e-05 ***
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.2.0  0.86823
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.0.1  0.50619
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)1.0.1  0.70914
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.1.1  0.00089 ***
## poly(db_od, clay_tot_p, sand_tot_p, degree = 2, raw = TRUE)0.0.2  0.02920 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7199 on 6656 degrees of freedom
## Multiple R-squared:  0.477,  Adjusted R-squared:  0.4763
## F-statistic: 674.4 on 9 and 6656 DF,  p-value: < 2.2e-16
```

After fitting the MPR model, Ksat values were predicted for the testing soil samples and compared with the observed Ksat soil samples. Then, the model was evaluated using root mean square error (RMSE), Concordance correlation coefficient (CCC) and Coefficient of determination (R square).

```
##Predction for the testing dataset

holdout$predict<- predict(model_temp_ksat, holdout)

##Linear model between predictions and measurements

Lm_pre_mea<- lm(log_ksat~ predict, data = holdout)

summary(Lm_pre_mea)
```

```
##
## Call:
## lm(formula = log_ksat ~ predict, data = holdout)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3523 -0.2950  0.0544  0.3747  2.6662
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.04647    0.05629   0.826    0.409
## predict      0.97572    0.02567  38.004   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7227 on 1665 degrees of freedom
## Multiple R-squared:  0.4645, Adjusted R-squared:  0.4642
## F-statistic:  1444 on 1 and 1665 DF,  p-value: < 2.2e-16
```

```
##RMSE

rmse(holdout$log_ksat, holdout$predict)
```

```
## [1] 0.7225068
```

```
##CCC

ccc = DescTools::CCC(holdout$log_ksat, holdout$predict, ci = "z-transform", conf.leve
l = 0.95, na.rm=TRUE)$rho.c

ccc
```

```
##         est    lwr.ci    upr.ci
## 1 0.6399059 0.6138452 0.6645735
```

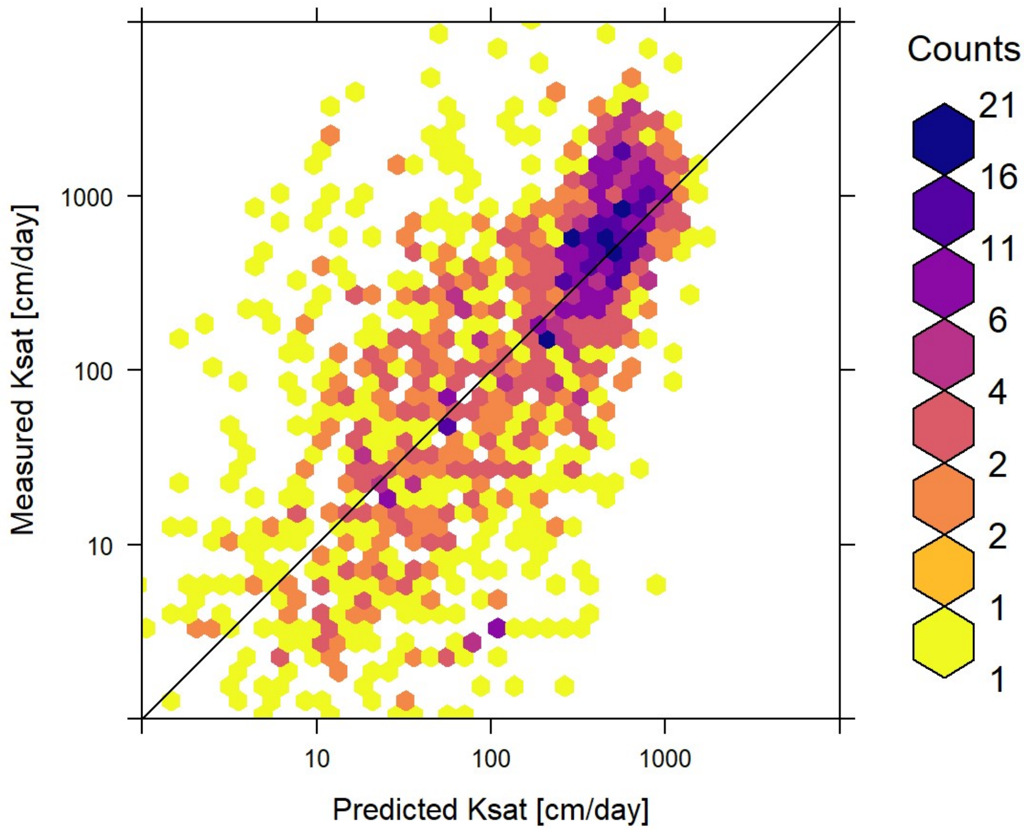
```
##Transform log10 to normal values

holdout$log10ksat1<- 10^holdout$log_ksat

holdout$response1<- 10^holdout$predict

##Hexbin Plot

hexbinplot(log10ksat1~response1,
           panel = function(x, y, ...){
             panel.hexbinplot(x, y, ...)
                     panel.abline(c(0, 1))
           },
           data = holdout,xlab = "Predicted Ksat [cm/day]", ylab = "Measured Ksat [cm/
day]",cex.axis = 4, aspect="1", xbins=40, colramp = function(n) {viridis (8,  alpha =
1, begin = 0, end = 1, direction = -1,option = "C")},xlim=c(1,10000), ylim=c(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(0,0.01,0.03,0.07,0.1
5,0.25,0.5,0.75,1) )
```



#Temperate climate region PTF tested on Tropical dataset

The temperate climate region ksat model was tested on tropical soil samples.

```
##model_tetsed_on_tropical soil samples

Trop = read.csv("E:/maps_tests/All_regions_ksat/Tropical_5_04_2020.csv")

Trop$log_ksat = signif(log10( rowMeans(Trop[,c("ksat_lab","ksat_field")], na.rm=TRUE)
), 4)

soil_temp_clay_trop <- Trop[!is.na(Trop$clay_tot_p),]

soil_temp_clay_bd_trop<- soil_temp_clay_trop[!is.na(soil_temp_clay_trop$db_od),]

##Predction for the testing dataset

soil_temp_clay_bd_trop$ predict<- predict(model_temp_ksat, soil_temp_clay_bd_trop)

##Linear model between predictions and measurements

hh_trop<- lm(log_ksat~ predict, data = soil_temp_clay_bd_trop)

summary(hh_trop)
```

```
##
## Call:
## lm(formula = log_ksat ~ predict, data = soil_temp_clay_bd_trop)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.7416 -0.3273  0.1847  0.4658  1.7606
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.96066    0.05921   16.22   <2e-16 ***
## predict      0.60684    0.02816   21.55   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7498 on 1120 degrees of freedom
## Multiple R-squared:  0.2931, Adjusted R-squared:  0.2924
## F-statistic: 464.3 on 1 and 1120 DF,  p-value: < 2.2e-16
```

```
##RMSE

rmse(soil_temp_clay_bd_trop$log_ksat, soil_temp_clay_bd_trop$predict)
```

```
## [1] 0.8348444
```

```
##CCC

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

ccc
```

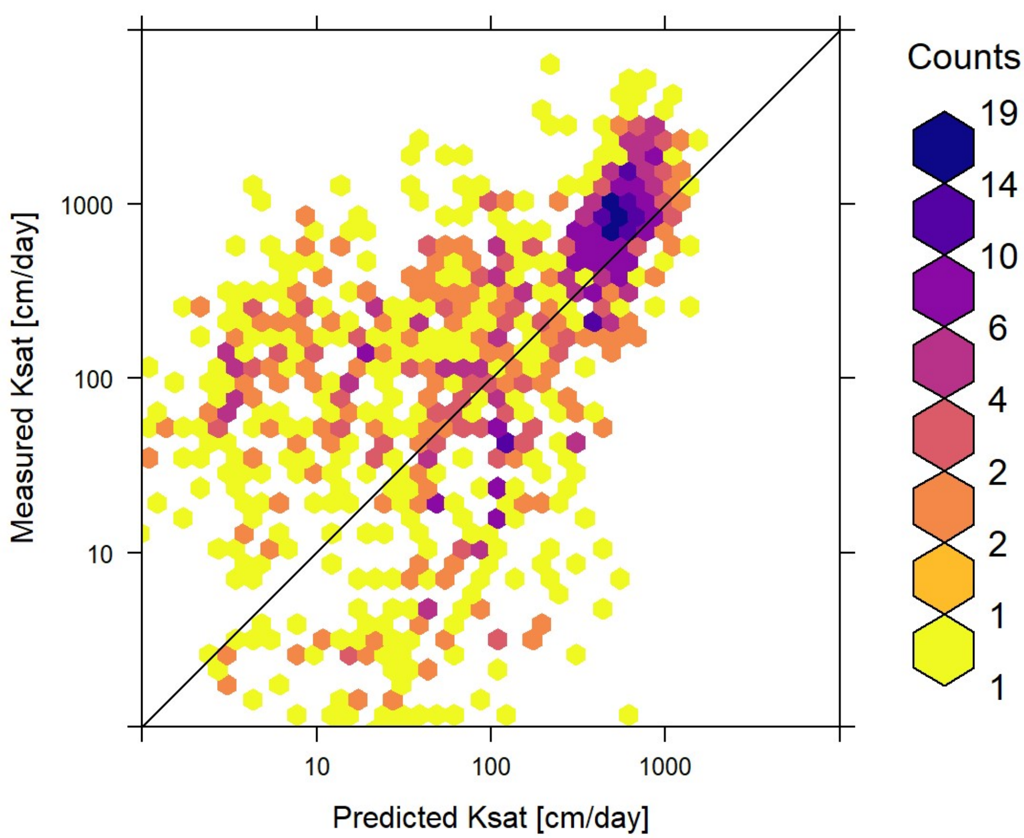
```
##         est    lwr.ci   upr.ci
## 1 0.5238328 0.4814777 0.563747
```

```
##Transform log10 to normal values

soil_temp_clay_bd_trop$log10ksat1<- 10^soil_temp_clay_bd_trop$log_ksat

soil_temp_clay_bd_trop$response1<- 10^soil_temp_clay_bd_trop$predict

hexbinplot(log10ksat1~response1,
           panel = function(x, y, ...){
             panel.hexbinplot(x, y, ...)
                     panel.abline(c(0, 1))
           },
           data = soil_temp_clay_bd_trop,xlab = "Predicted Ksat [cm/day]", ylab = "Mea
sured Ksat [cm/day]",cex.axis = 4, aspect="1", xbins=40, colramp = function(n) {viridi
s (8,  alpha = 1, begin = 0, end = 1, direction = -1,option = "C")},xlim=c(1,10000), y
lim=c(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(0,0.01,0.03,0.07,0.1
5,0.25,0.5,0.75,1) )
```



#Lab measured Ksat values PTF

Similarly, MPR model was fitted for lab measured ksat samples

```
##model_built_on_lab measurements

lab = read.csv("E:/maps_tests/All_regions_ksat/lab_5_04_2020.csv")

lab$log_ksat = signif(log10( rowMeans(lab[,c("ksat_lab","ksat_field")], na.rm=TRUE)
), 4)

soil_temp_clay_lab <- lab[!is.na(lab$clay_tot_psa),]

soil_temp_clay_bd_lab<- soil_temp_clay_lab[!is.na(soil_temp_clay_lab$db_od),]

# Randomly split the dataset into training and testing datasets

sample_size = floor(0.80*nrow(soil_temp_clay_bd_lab))

set.seed(777)

picked = sample(seq_len(nrow(soil_temp_clay_bd_lab)),size = sample_size)

development =soil_temp_clay_bd_lab[picked,]

nrow(development)
```

```
## [1] 6444
```

```
holdout =soil_temp_clay_bd_lab[-picked,]

nrow(holdout)
```

```
## [1] 1612
```

```
##Fitting MPR model with degree 2

model_lab<- lm(log_ksat ~ poly(db_od,clay_tot_psa, sand_tot_psa, degree=2, raw=TRUE),
data =development )

summary(model_lab)
```

```
##
## Call:
## lm(formula = log_ksat ~ poly(db_od, clay_tot_psa, sand_tot_psa,
##     degree = 2, raw = TRUE), data = development)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.8451 -0.3015  0.0516  0.3752  3.0734
##
## Coefficients:
##                                                                  Estimate
## (Intercept)                                                     1.442e+00
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.0.0  2.053e+00
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)2.0.0 -1.256e+00
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.1.0 -5.330e-02
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.1.0 -5.163e-05
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.2.0  5.538e-04
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.0.1  7.985e-03
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.0.1 -8.052e-04
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.1.1  4.311e-05
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.0.2  5.187e-05
##                                                                 Std. Error
## (Intercept)                                                      3.499e-01
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.0.0  3.719e-01
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)2.0.0  1.416e-01
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.1.0  8.242e-03
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.1.0  4.661e-03
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.2.0  5.839e-05
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.0.1  4.595e-03
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.0.1  2.887e-03
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.1.1  6.710e-05
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.0.2  3.042e-05
##                                                                 t value
## (Intercept)                                                       4.122
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.0.0   5.521
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)2.0.0  -8.872
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.1.0  -6.467
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.1.0  -0.011
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.2.0   9.486
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.0.1   1.738
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.0.1  -0.279
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.1.1   0.642
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.0.2   1.705
##                                                                 Pr(>|t|)
## (Intercept)                                                     3.81e-05
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.0.0 3.51e-08
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)2.0.0  < 2e-16
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.1.0 1.07e-10
```

```
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.1.0   0.9912
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.2.0  < 2e-16
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.0.1   0.0823
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.0.1   0.7803
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.1.1   0.5206
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.0.2   0.0882
##
## (Intercept)                                                         ***
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.0.0 ***
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)2.0.0 ***
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.1.0 ***
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.1.0
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.2.0 ***
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.0.1 .
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)1.0.1
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.1.1
## poly(db_od, clay_tot_psa, sand_tot_psa, degree = 2, raw = TRUE)0.0.2 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6835 on 6433 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.5322, Adjusted R-squared:  0.5316
## F-statistic: 813.3 on 9 and 6433 DF,  p-value: < 2.2e-16
```

```
##Predction for the testing dataset

holdout$predict<- predict(model_lab, holdout)

##Linear model between predictions and measurements

LM_pre_mea_Lab<- lm(log_ksat~ predict, data = holdout)

summary(LM_pre_mea_Lab)
```

```
##
## Call:
## lm(formula = log_ksat ~ predict, data = holdout)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3039 -0.2915  0.0530  0.3711  3.0341
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.008271   0.052005  -0.159    0.874
## predict      0.996881   0.023202  42.966   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6739 on 1610 degrees of freedom
## Multiple R-squared:  0.5342, Adjusted R-squared:  0.5339
## F-statistic:  1846 on 1 and 1610 DF,  p-value: < 2.2e-16
```

```
rmse(holdout$log_ksat, holdout$predict)
```

```
## [1] 0.673693
```

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

ccc
```

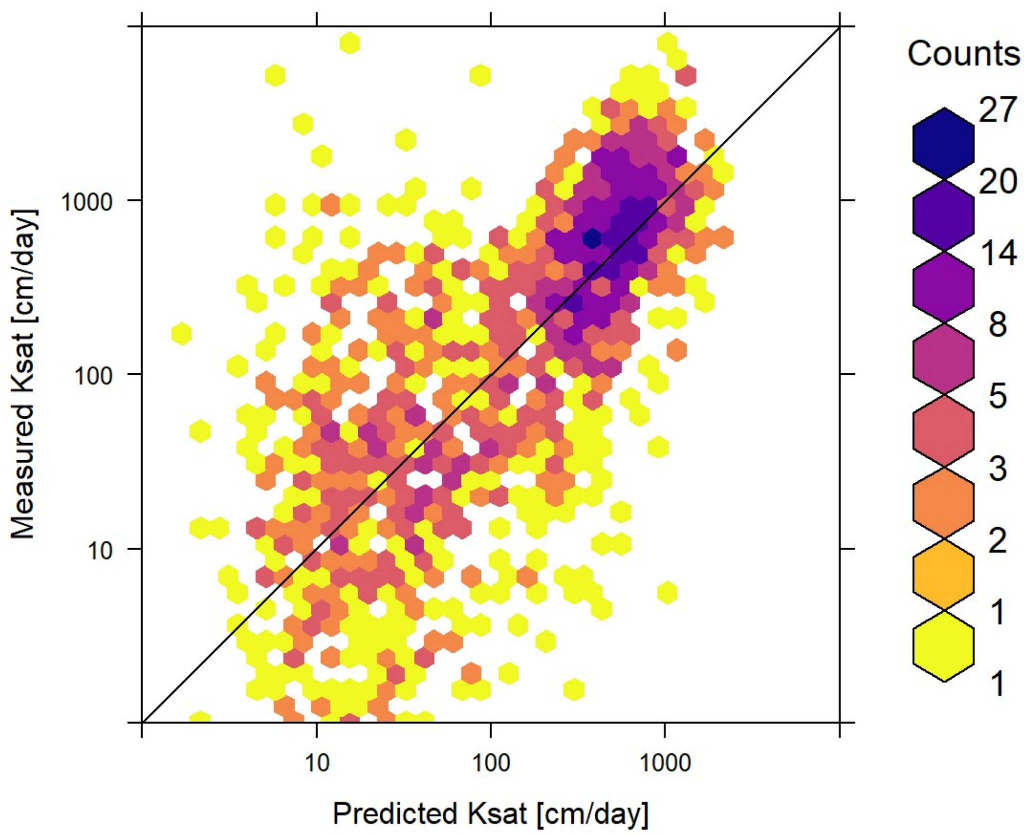
```
##         est    lwr.ci    upr.ci
## 1 0.6969057 0.6733911 0.7190116
```

```
## Transform log 10 to normal values

holdout$log10ksat1<- 10^holdout$log_ksat

holdout$response1<- 10^holdout$predict

hexbinplot(log10ksat1~response1,
           panel = function(x, y, ...){
             panel.hexbinplot(x, y, ...)
                     panel.abline(c(0, 1))
           },
           data = holdout,xlab = "Predicted Ksat [cm/day]", ylab = "Measured Ksat [cm/
day]",cex.axis = 4, aspect="1", xbins=35, colramp = function(n) {viridis (8,  alpha =
1, begin = 0, end = 1, direction = -1,option = "C")},xlim=c(1,10000), ylim=c(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(0,0.01,0.03,0.07,0.1
5,0.25,0.5,0.75,1) )
```



#Lab ksat PTF tested on field dataset

Here, we applied the same method (MPR) for developing the PTF for Lab measured ksat values. We have followed the same steps as described for temperate region PTFs such as divided the dataset into training (80%) and testing soil samples (20%).

```
##model applied on field measurments

field = read.csv("E:/maps_tests/All_regions_ksat/field_5_04_2020.csv")

field$log_ksat = signif(log10( rowMeans(  field[,c("ksat_lab","ksat_field")], na.rm=TR
UE) ), 4)

soil_temp_clay_field<- field[!is.na(field$clay_tot_psa),]

soil_temp_clay_bd_field<- soil_temp_clay_field[!is.na(soil_temp_clay_field$db_od),]

##Predction for the testing dataset

soil_temp_clay_bd_field$ predict<- predict(model_lab, soil_temp_clay_bd_field)

##Linear model between predictions and measurements

LM_pre_mea_field<- lm(log_ksat~ predict, data = soil_temp_clay_bd_field)

summary(LM_pre_mea_field)
```

```
##
## Call:
## lm(formula = log_ksat ~ predict, data = soil_temp_clay_bd_field)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.0266 -0.4092  0.0809  0.5778  3.1793
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.26338    0.06120   20.64   <2e-16 ***
## predict      0.38553    0.03668   10.51   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9824 on 2394 degrees of freedom
## Multiple R-squared:  0.04411,    Adjusted R-squared:  0.04371
## F-statistic: 110.5 on 1 and 2394 DF,  p-value: < 2.2e-16
```

```
##RMSE

rmse(soil_temp_clay_bd_field$log_ksat, soil_temp_clay_bd_field$predict)
```

```
## [1] 1.079013
```

```
##CCC

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

ccc
```

```
##         est    lwr.ci    upr.ci
## 1 0.1654739 0.1347498 0.1958801
```

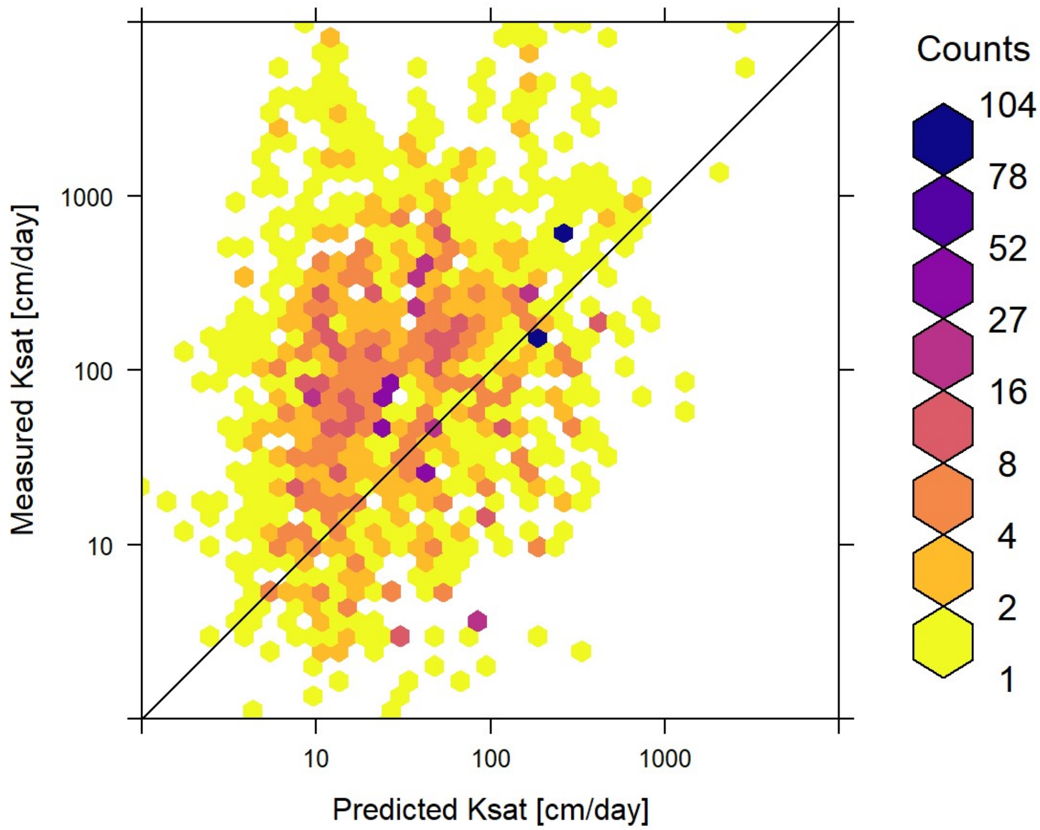
```
## Transform log 10 to normal values

soil_temp_clay_bd_field$log10ksat1<- 10^soil_temp_clay_bd_field$log_ksat

soil_temp_clay_bd_field$response1<- 10^soil_temp_clay_bd_field$predict

## hexabin plots

hexbinplot(log10ksat1~response1,
           panel = function(x, y, ...){
             panel.hexbinplot(x, y, ...)
                     panel.abline(c(0, 1))
           },
           data = soil_temp_clay_bd_field,xlab = "Predicted Ksat [cm/day]", ylab = "Me
asured Ksat [cm/day]",cex.axis = 4, aspect="1", xbins=35, colramp = function(n) {virid
is (8,  alpha = 1, begin = 0, end = 1, direction = -1,option = "C")},xlim=c(1,10000),
ylim=c(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(0,0.01,0.03,0.07,0.1
5,0.25,0.5,0.75,1) )
```



Ksat PTF developed for temperate soil samples using RF

Here, we tried to fit the PTF for temperate soil samples using the random forest algorithm.

```
PTF_temp<-read.csv("E:/maps_tests/All_regions_ksat/Temperate_5_04_2020.csv")

PTF_temp$log_ksat = signif(log10( rowMeans(PTF_temp[,c("ksat_lab","ksat_field")], na.r
m=TRUE)), 4)

soil_ksat3 <- PTF_temp[!is.na(PTF_temp$clay_tot_p),]

soil_ksat4 <- soil_ksat3[!is.na(soil_ksat3$db_od),]

## Selecting the list of indepenedent covariates used for developing a model

I.vars = make.names(unique(unlist(sapply(c( "clay_","db_od", "sand_"), function(i){nam
es(soil_ksat4)[grep(i, names(soil_ksat4))]}))))

## Dependent/Target variable

t.vars = c("log_ksat")

sel.n <- c(t.vars,I.vars)

sel.r <- complete.cases(soil_ksat4[,sel.n])

PTF_temp2 <- soil_ksat4[sel.r,sel.n]

## dividing the dataset based on testing and trainning datasets

test.set = seq(3, nrow(PTF_temp2), by = 5)

str(test.set)
```

```
##  num [1:1667] 3 8 13 18 23 28 33 38 43 48 ...
```

```
training.set = which(!1:nrow(PTF_temp2) %in% test.set)

str(training.set)
```

```
##  int [1:6666] 1 2 4 5 6 7 9 10 11 12 ...
```

```
##Fitting the Random forest algorithm: Ranger

learner = mlr::makeLearner("regr.ranger", mtry =3, num.trees=85)

tsk <- mlr::makeRegrTask(data = PTF_temp2, target = t.vars, blocking = NULL)

m <- mlr::train(learner, tsk,subset = training.set)

##Predction for the testing dataset

p = predict(m, newdata = PTF_temp2[test.set,] )

comparison <- p$data

##Linear model between predictions and measurements

gg<- lm(comparison$truth~ comparison$response)

summary(gg)
```

```
##
## Call:
## lm(formula = comparison$truth ~ comparison$response)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.8981 -0.2560  0.0269  0.3252  2.8672
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          0.18942    0.04634   4.088 4.57e-05 ***
## comparison$response  0.91330    0.02081  43.878  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6659 on 1665 degrees of freedom
## Multiple R-squared:  0.5362, Adjusted R-squared:  0.536
## F-statistic:  1925 on 1 and 1665 DF,  p-value: < 2.2e-16
```

```
##CCC

ccc = DescTools::CCC(comparison$truth, comparison$response, ci = "z-transform", conf.l
evel = 0.95, na.rm=TRUE)$rho.c

ccc
```

```
##         est    lwr.ci    upr.ci
## 1 0.7147411 0.6916738 0.7363516
```

```
##RMSE

rmse(comparison$truth, comparison$response)
```

```
## [1] 0.6689804
```

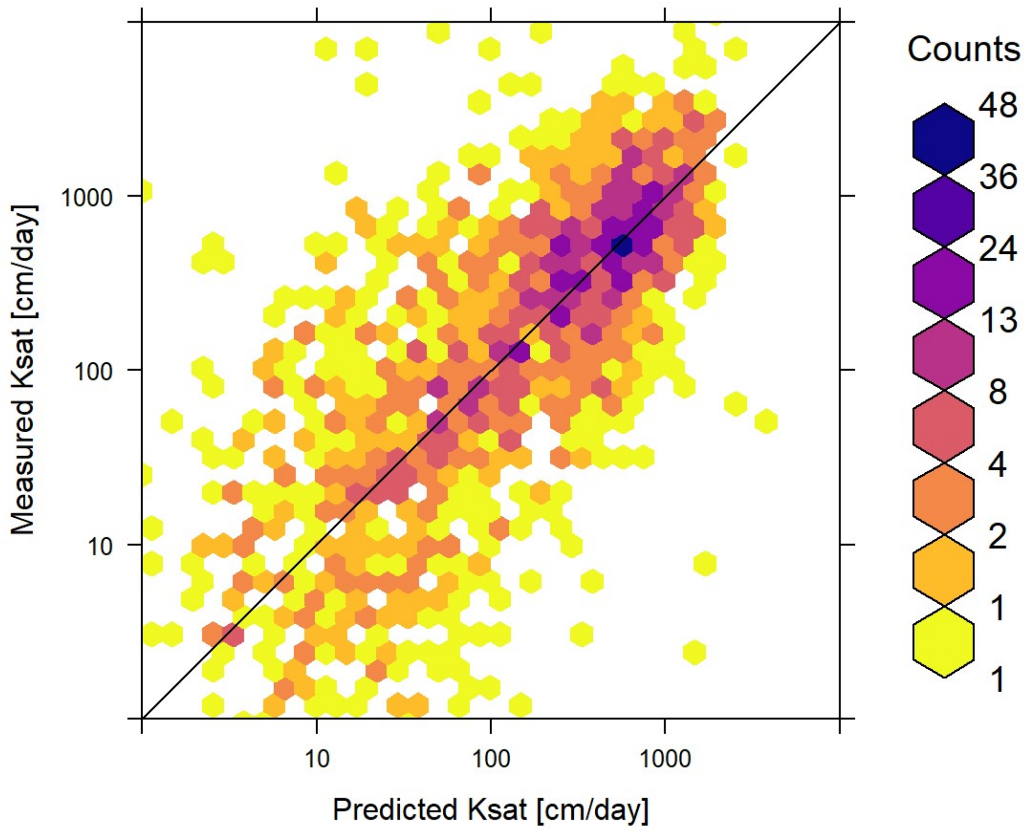
```
##Transform log10 to normal values

comparison$log10ksat1<- 10^comparison$truth

comparison$response1<- 10^comparison$response

##Hexbin Plot

hexbinplot(log10ksat1~response1,
           panel = function(x, y, ...){
             panel.hexbinplot(x, y, ...)
                     panel.abline(c(0, 1))
           },
           data =comparison ,xlab = "Predicted Ksat [cm/day]", ylab = "Measured Ksat
[cm/day]",cex.axis = 4, aspect="1", xbins=40, colramp = function(n) {viridis (8,  alph
a = 1, begin = 0, end = 1, direction = -1,option = "C")},xlim=c(1,10000), ylim=c(1,100
00),
           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(0,0.01,0.03,0.07,0.1
5,0.25,0.5,0.75,1) )
```



Temperate Ksat PTF tested on tropical dataset

PTF developed for the temperate region using RF used to predict the ksat values for tropical soil samples.

```
Trop = read.csv("E:/maps_tests/All_regions_ksat/Tropical_5_04_2020.csv")

Trop$log10ksat = signif(log10( rowMeans(Trop[,c("ksat_lab","ksat_field")], na.rm=TRU
E) ), 4)

soil_temp_clay_trop <- Trop[!is.na(Trop$clay_tot_p),]

soil_temp_clay_bd_trop<- soil_temp_clay_trop[!is.na(soil_temp_clay_trop$db_od),]


##Predction for testing dataset

p = predict(m, newdata = soil_temp_clay_bd_trop)

comparison <- p$data

y<- cbind(comparison, soil_temp_clay_bd_trop)

##CCC

ccc = DescTools::CCC(y$log10ksat, y$response, ci = "z-transform", conf.level = 0.95, n
a.rm=TRUE)$rho.c

ccc
```

```
##         est    lwr.ci    upr.ci
## 1 0.5076688 0.4640259 0.5488532
```

```
##RMSE

rmse(y$log10ksat, y$response)
```

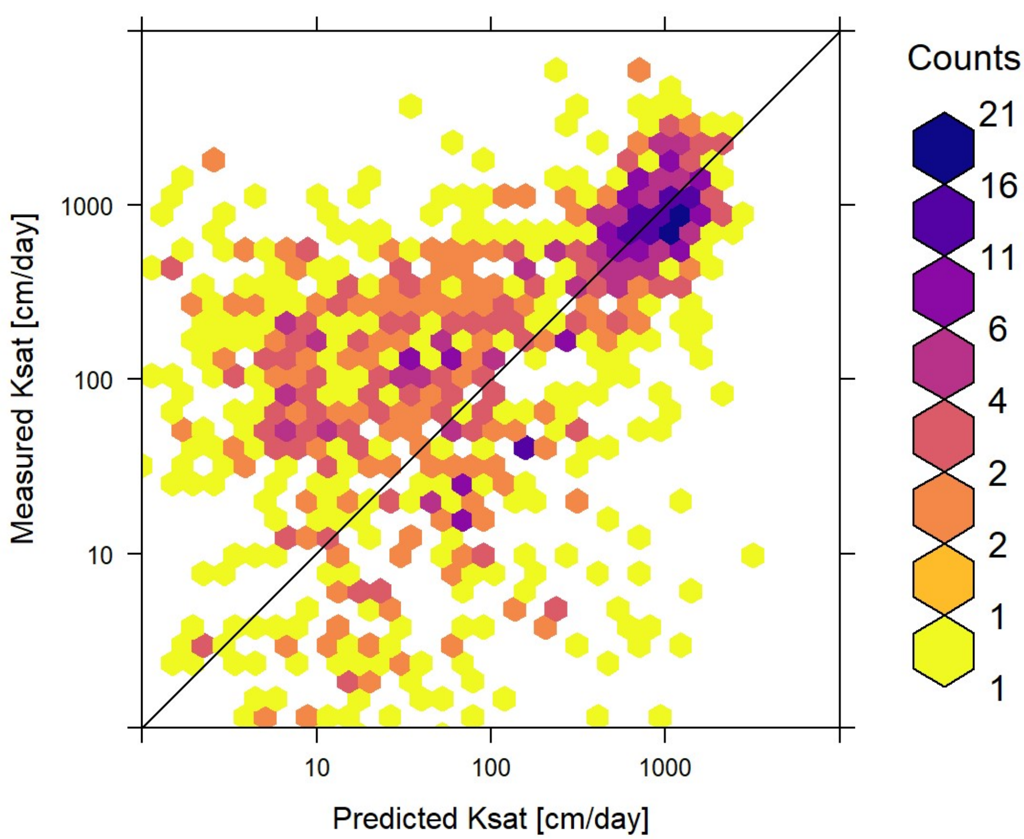
```
## [1] 0.9138471
```

```
##Transform Log10 to normal values

y$log10ksat1<- 10^y$log10ksat

y$response1<- 10^y$response

hexbinplot(log10ksat1~response1,
           panel = function(x, y, ...){
             panel.hexbinplot(x, y, ...)
                     panel.abline(c(0, 1))
           },
           data =y ,xlab = "Predicted Ksat [cm/day]", ylab = "Measured Ksat [cm/day]",
cex.axis = 4, aspect="1", xbins=40, colramp = function(n) {viridis (8,  alpha = 1, beg
in = 0, end = 1, direction = -1,option = "C")},xlim=c(1,10000), ylim=c(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(0,0.01,0.03,0.07,0.1
5,0.25,0.5,0.75,1) )
```



Ksat PTF developed for Lab measurements soil samples using RF

```
PTF_lab<-read.csv("E:/maps_tests/All_regions_ksat/lab_5_04_2020.csv")

PTF_lab$log_ksat = signif(log10( rowMeans(PTF_lab[,c("ksat_lab","ksat_field")], na.rm=
TRUE)), 4)

soil_ksat3 <- PTF_lab[!is.na(PTF_lab$clay_tot_psa),]

soil_ksat4 <- soil_ksat3[!is.na(soil_ksat3$db_od),]

## Selecting the list of independent covariates used for developing a model

I.vars = make.names(unique(unlist(sapply(c( "clay_","db_od", "sand_"), function(i){nam
es(soil_ksat4)[grep(i, names(soil_ksat4))]}))))

## Dependent/Target variable

t.vars = c("log_ksat")

sel.n <- c(t.vars,I.vars)

sel.r <- complete.cases(soil_ksat4[,sel.n])

PTF_lab2 <- soil_ksat4[sel.r,sel.n]

## dividing the dataset based on testing and training datasets

test.set = seq(3, nrow(PTF_lab2), by = 5)

str(test.set)
```

```
##  num [1:1611] 3 8 13 18 23 28 33 38 43 48 ...
```

```
training.set = which(!1:nrow(PTF_lab2) %in% test.set)

str(training.set)
```

```
##  int [1:6444] 1 2 4 5 6 7 9 10 11 12 ...
```

```
##Fitting the Random forest algorithm: Ranger

learner = mlr::makeLearner("regr.ranger", mtry =3, num.trees=85)

tsk <- mlr::makeRegrTask(data = PTF_lab2, target = t.vars, blocking = NULL)# weights
= case.weights)

m <- mlr::train(learner, tsk,subset = training.set)

##Predction for testing dataset

p = predict(m, newdata = PTF_lab2[test.set,] )

performance(p)
```

```
##       mse
## 0.4450646
```

```
comparison <- p$data

##Linear model between predictions and measurements

gg<- lm(comparison$truth~ comparison$response)

summary(gg)
```

```
##
## Call:
## lm(formula = comparison$truth ~ comparison$response)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3989 -0.2698  0.0459  0.3811  3.0153
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          0.16739    0.04597   3.642  0.00028 ***
## comparison$response  0.92434    0.02063  44.807  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6647 on 1609 degrees of freedom
## Multiple R-squared:  0.5551, Adjusted R-squared:  0.5548
## F-statistic:  2008 on 1 and 1609 DF,  p-value: < 2.2e-16
```

```
##CCC

ccc = DescTools::CCC(comparison$truth, comparison$response, ci = "z-transform", conf.l
evel = 0.95, na.rm=TRUE)$rho.c

ccc
```

```
##         est    lwr.ci    upr.ci
## 1 0.7280216 0.7054402 0.7491265
```

```
##RMSE

rmse(comparison$truth, comparison$response)
```

```
## [1] 0.6671316
```

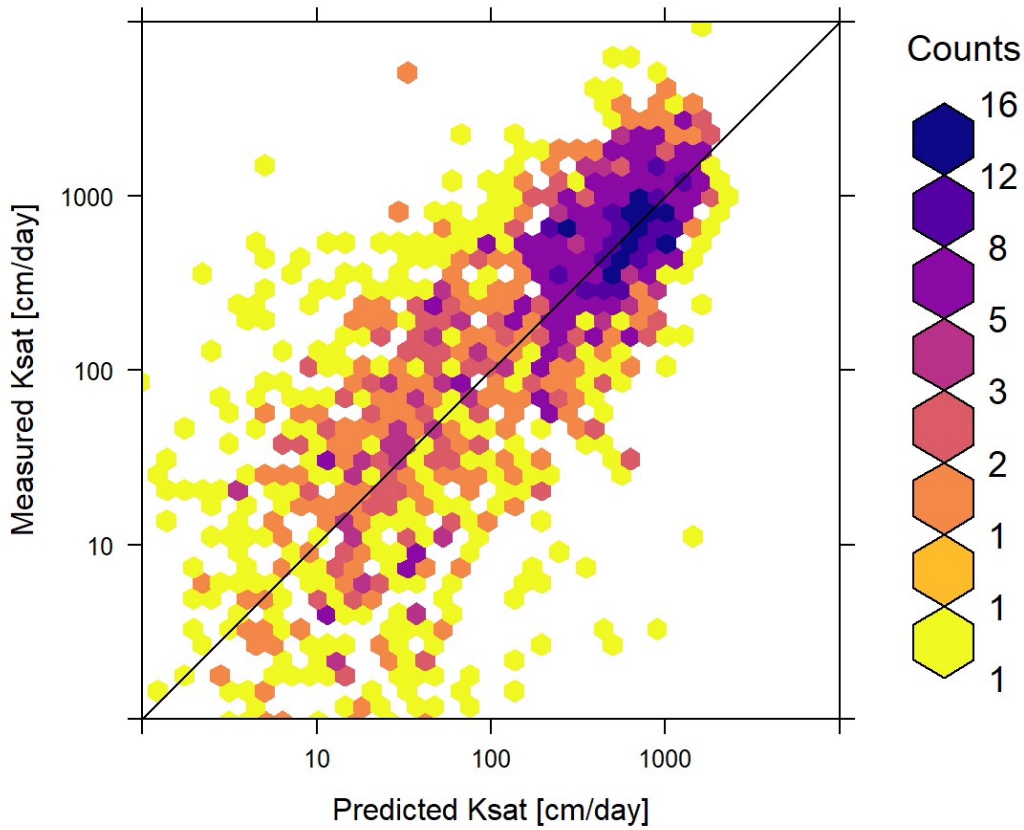
```
##Transform log10 to normal values

comparison$log10ksat1<- 10^comparison$truth

comparison$response1<- 10^comparison$response

##Hexbin Plot

hexbinplot(log10ksat1~response1,
           panel = function(x, y, ...){
             panel.hexbinplot(x, y, ...)
                     panel.abline(c(0, 1))
           },
           data =comparison ,xlab = "Predicted Ksat [cm/day]", ylab = "Measured Ksat
[cm/day]",cex.axis = 4, aspect="1", xbins=40, colramp = function(n) {viridis (8,  alph
a = 1, begin = 0, end = 1, direction = -1,option = "C")},xlim=c(1,10000), ylim=c(1,100
00),
           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(0,0.01,0.03,0.07,0.1
5,0.25,0.5,0.75,1) )
```



Lab PTF tested on field dataset

```
PTF_lab<-read.csv("E:/maps_tests/All_regions_ksat/field_5_04_2020.csv")

PTF_lab$log_ksat = signif(log10( rowMeans(PTF_lab[,c("ksat_lab","ksat_field")], na.rm=
TRUE)), 4)

soil_ksat3 <- PTF_lab[!is.na(PTF_lab$clay_tot_psa),]

soil_ksat4 <- soil_ksat3[!is.na(soil_ksat3$db_od),]
##Prediction for testing dataset

p1 = predict(m, newdata = soil_ksat4)

comparison <- p1$data

##Linear model between predictions and measurements

gg<- lm(comparison$truth~ comparison$response)

summary(gg)
```

```
##
## Call:
## lm(formula = comparison$truth ~ comparison$response)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.0974 -0.4399  0.1435  0.5993  3.0630
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          1.47165    0.05627   26.16  < 2e-16 ***
## comparison$response  0.23983    0.03152    7.61 3.92e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9929 on 2394 degrees of freedom
## Multiple R-squared:  0.02362,    Adjusted R-squared:  0.02321
## F-statistic: 57.91 on 1 and 2394 DF,  p-value: 3.923e-14
```

```
##CCC

ccc = DescTools::CCC(comparison$truth, comparison$response, ci = "z-transform", conf.l
evel = 0.95, na.rm=TRUE)$rho.c

ccc
```

```
##         est    lwr.ci    upr.ci
## 1 0.1355882 0.1007948 0.1700505
```

```
##RMSE

rmse(comparison$truth, comparison$response)
```

```
## [1] 1.125453
```

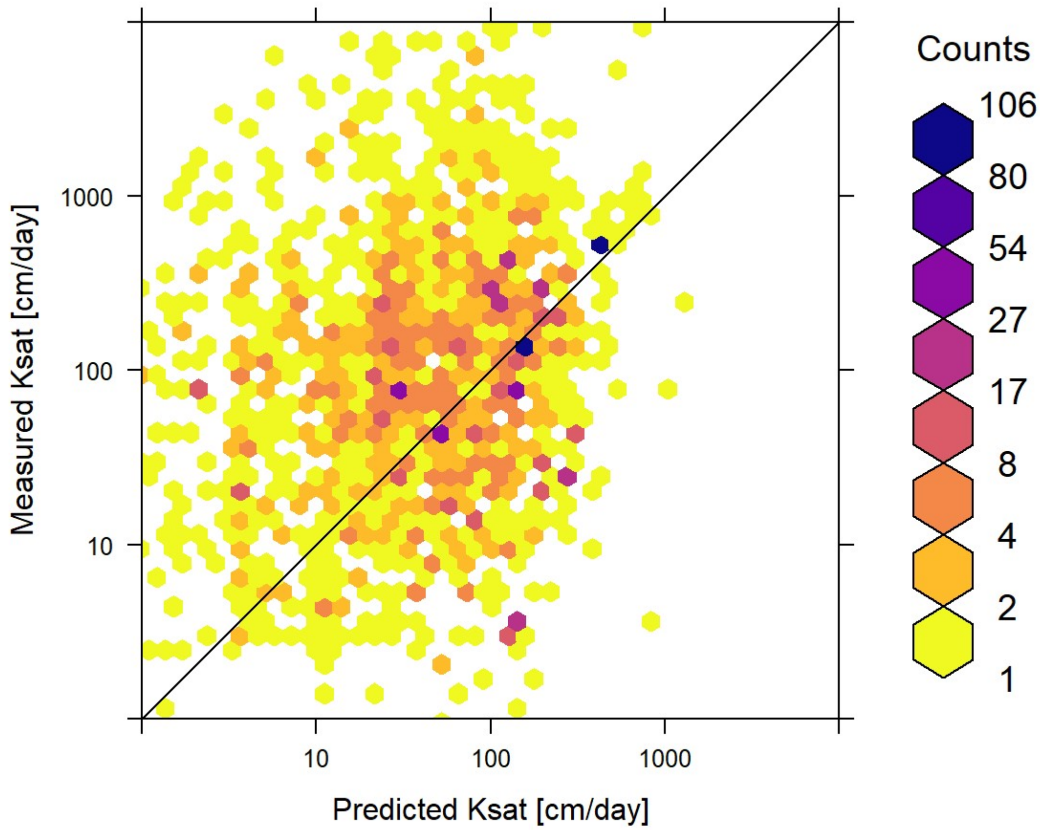
```r
##Transform log10 to normal values

comparison$log10ksat1<- 10^comparison$truth

comparison$response1<- 10^comparison$response

##Hexbin Plot

hexbinplot(log10ksat1~response1,
           panel = function(x, y, ...){
             panel.hexbinplot(x, y, ...)
                     panel.abline(c(0, 1))
           },
           data =comparison ,xlab = "Predicted Ksat [cm/day]", ylab = "Measured Ksat
[cm/day]",cex.axis = 4, aspect="1", xbins=40, colramp = function(n) {viridis (8,  alph
a = 1, begin = 0, end = 1, direction = -1,option = "C")},xlim=c(1,10000), ylim=c(1,100
00),
           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(0,0.01,0.03,0.07,0.1
5,0.25,0.5,0.75,1))
```



Measured Ksat [cm/day]
Predicted Ksat [cm/day]
10
100
1000
Counts
106
80
54
27
17
8
4
2
1