Available Water Capacity Random Forest Model from Cornell Soil Health Program

Joseph P. Amsili, Harold M. van Es, and Robert R. Schindelbeck

7/13/2020

Contents

Set up 1
Random Forest Model 1
Random Forest Explainer 2
Visualizing Random Forest Model 7
Compare Predicted and Observed AWC 13

Set up

1) Load Packages required for Random Forest Analysis
2) Load SH database ("2020-7_Cornell_SH_dataset_for_AWC_model_development.csv")
3) Load Constructed dataset to visualize partial dependence of variables in model. This can also be achieved more routinely with the pdp package in R.

```
library(tidyverse)
library(Metrics) # for rmse function
library(randomForest)
library(randomForestExplainer)

## A function to present values as absolute values and commas to mark 1,000s. For AWC vs soil texture p
absolutenumbers <- function(x, ...) {format(abs(x), big.mark = ",", trim = TRUE,
                            scientific = FALSE, ...)}


SHdata <-read.csv('2020-7_Cornell_SH_dataset_for_AWC_model_development.csv')
Madeup.SHdata.forAWCmodel<-read.csv('Madeup.data to look at predictions from AWC model.csv')
```

Random Forest Model

Set Seed for R's random number generator. This basically allows for someone else to exactly replicate this random forest analysis.

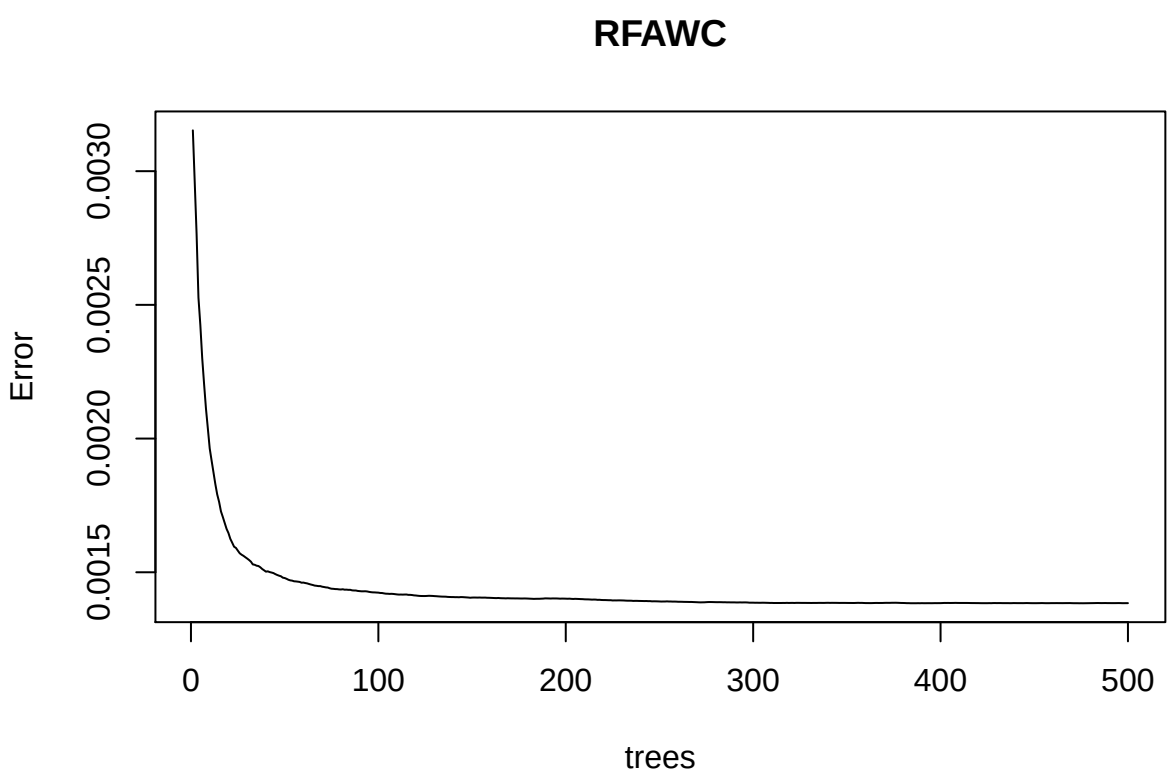
```
set.seed(500)
```

Run Random randomForest function to predict AWC using sand, silt, clay, SOM, WAS (wet aggregate stability), ACE, AC (POXC), Resp (soil respiration), K, Mg, Fe, and Mn.

```
RFAWC=randomForest(AWC~sand+silt+clay+SOM+WAS+AC+Resp+K+Mg+Fe+Mn, data=SHdata,
                   ntree=500, proximity=TRUE, localImp=TRUE)
save(RFAWC, file = "RFAWC.rda")
```

Since running this Random Forest Analysis takes a few minutes, you can save time by loading it from the working directory for subsequent analyses.

```
load("RFAWC.rda")
plot(RFAWC)
```




```
RFAWC
```

```
##
## Call:
##  randomForest(formula = AWC ~ sand + silt + clay + SOM + WAS +      AC + Resp + K + Mg + Fe + Mn, da
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 0.001384552
##                     % Var explained: 71.82
```

Random Forest Explainer

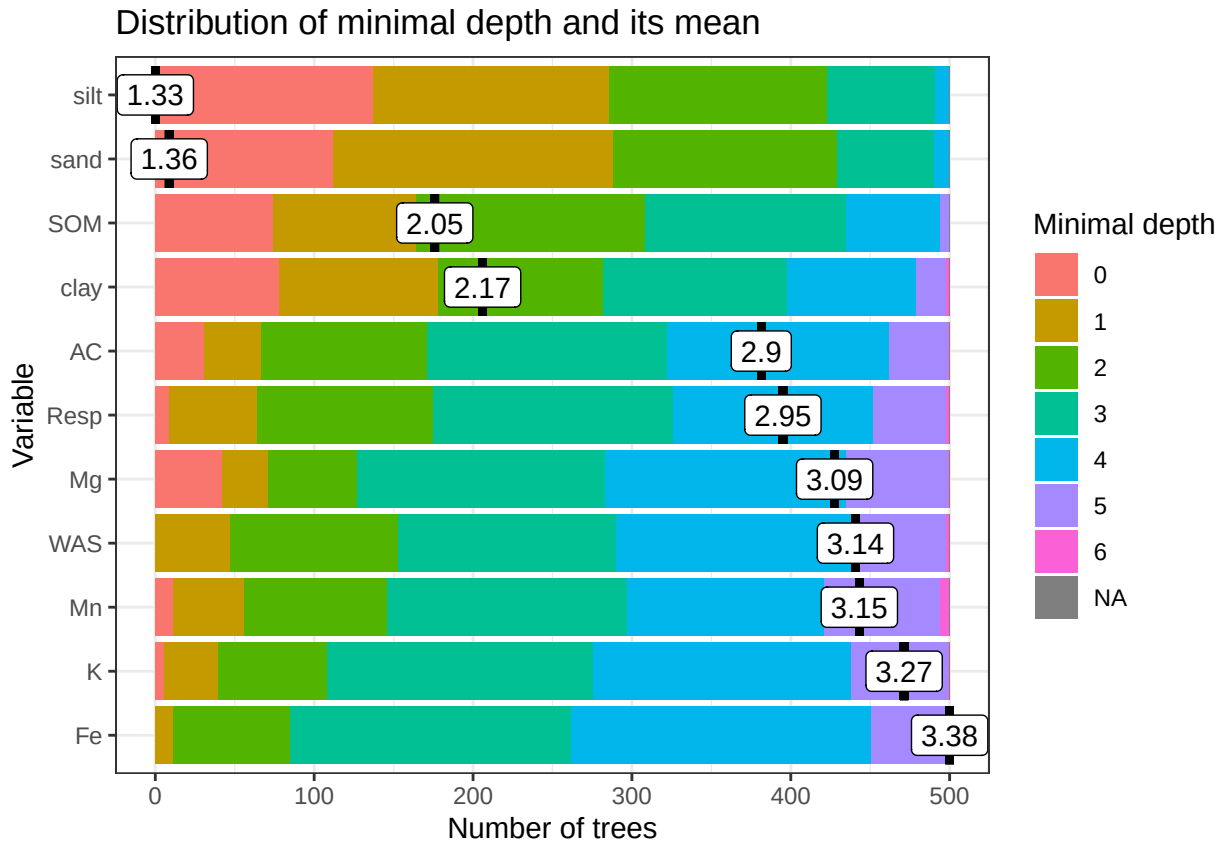
Use the Random Forest Explainer Package to understand Random Forest Analysis (These code lines take a long time to run and are therefore commented out:

The first function that we will run from the Random Forest Explainer Package is the min_depth_distribution function. This function helps visualize what variables are more important in the model (higher up in the decision trees).

```
# min_depth_frame <- min_depth_distribution(RFAWC)
# save(min_depth_frame, file = "min_depth_frame.rda")

load("min_depth_frame.rda")
head(min_depth_frame, n = 10)
```

```
##    tree variable minimal_depth
## 1     1       AC             2
## 2     1     clay             1
## 3     1       Fe             3
## 4     1        K             2
## 5     1       Mg             4
## 6     1       Mn             2
## 7     1     Resp             5
## 8     1     sand             1
## 9     1     silt             2
## 10    1      SOM             0
```

```
#png("RF.AWC.min depth distribution.png")
plot_min_depth_distribution(min_depth_frame, mean_sample="all_trees", k=13)
```




```
#dev.off() #place after plot code
```

This is one way to save your figures as a png file in your working directory (the lines before and after the plot function).

The second function that we will run from the Random Forest Explainer Package is the measure_importance

function.

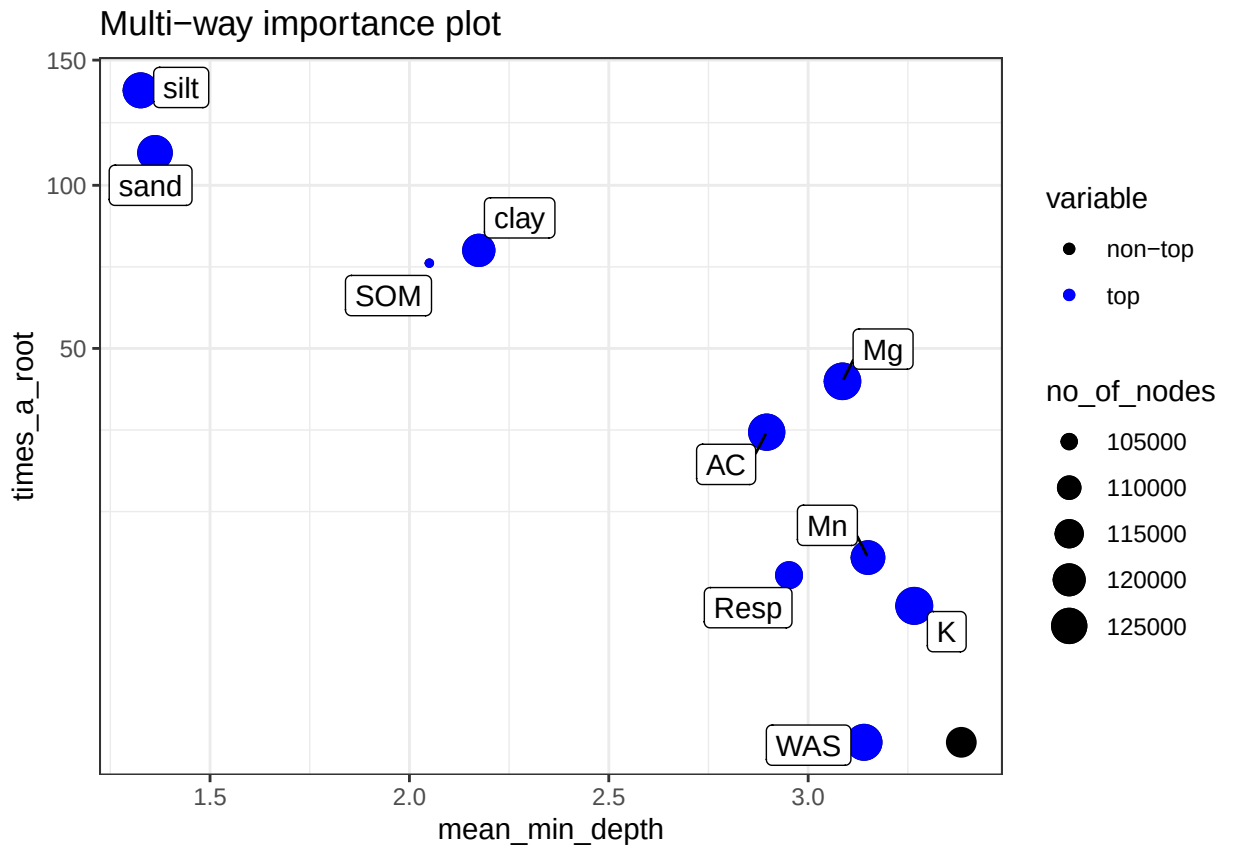
```
# importance_frame <- measure_importance(RFAWC)
# save(importance_frame, file = "importance_frame.rda")
load("importance_frame.rda")
importance_frame
```

```
##    variable mean_min_depth no_of_nodes mse_increase node_purity_increase
## 1        AC          2.896      126502 0.0003734612             2.029289
## 2      clay          2.174      120337 0.0006649878             3.531909
## 3        Fe          3.384      116850 0.0002609809             1.542823
## 4         K          3.266      127659 0.0002981864             1.634729
## 5        Mg          3.086      127107 0.0003740738             1.985424
## 6        Mn          3.150      122386 0.0002225542             1.614815
## 7      Resp          2.952      113474 0.0003135509             1.659570
## 8      sand          1.362      123797 0.0023201928             9.050404
## 9      silt          1.326      124495 0.0028537873             9.619378
## 10      SOM          2.050      103112 0.0006497874             3.400550
## 11      WAS          3.140      126019 0.0004983374             1.800952
##    no_of_trees times_a_root      p_value
## 1          500           31 7.671044e-60
## 2          500           78 9.862305e-01
## 3          500            0 1.000000e+00
## 4          500            6 8.816860e-87
## 5          500           42 2.672390e-73
## 6          500           11 3.623914e-05
## 7          500            9 1.000000e+00
## 8          500          112 1.201420e-16
## 9          500          137 3.981438e-25
## 10         500           74 1.000000e+00
## 11         500            0 4.327743e-50
```

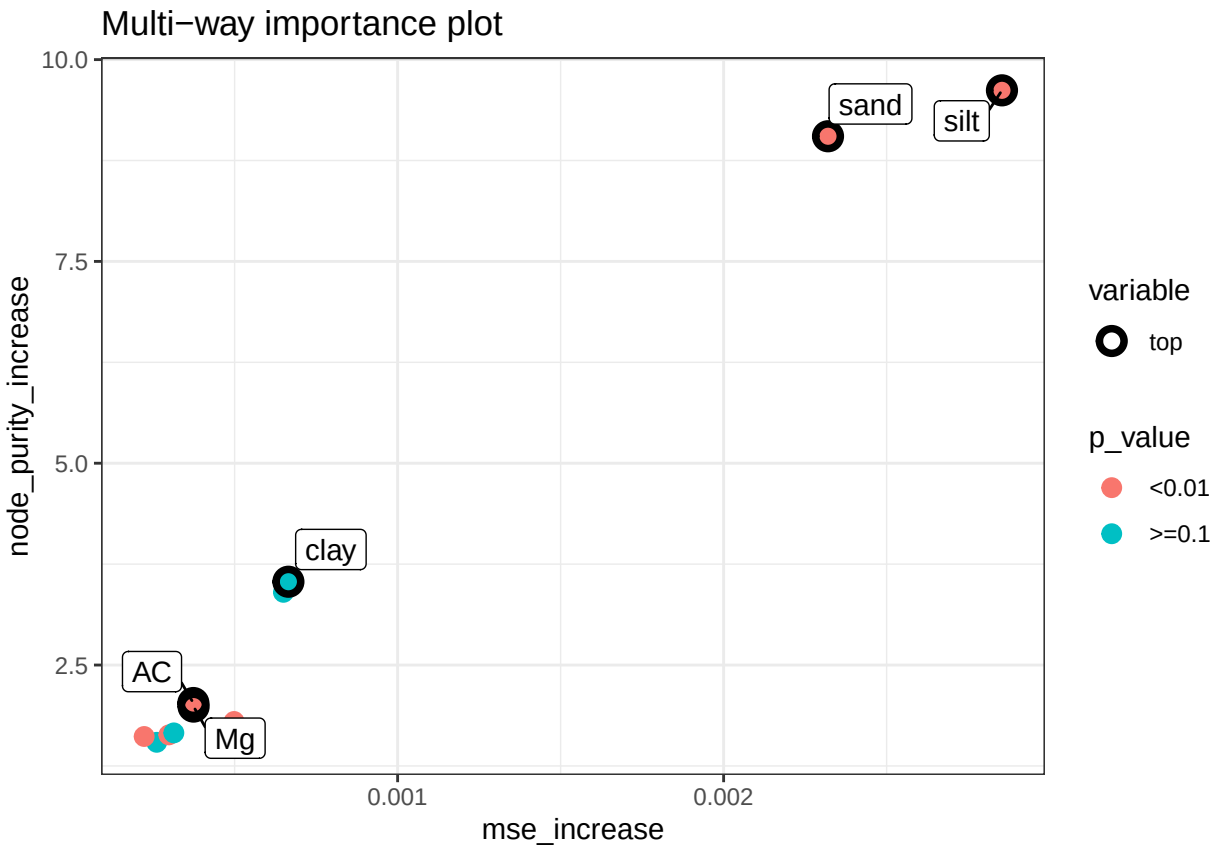
This function allows us to plot various attributes of the Random Forest Model against one another. Here we will plot:

1) mean_min_depth vs. times a root
2) mse_increase vs. node_purity_increase
3) Relationships between all measures (pairs plot), including, mean_min_depth, mse_increase, node_purity_increase, no_of_nodes, times_a_root

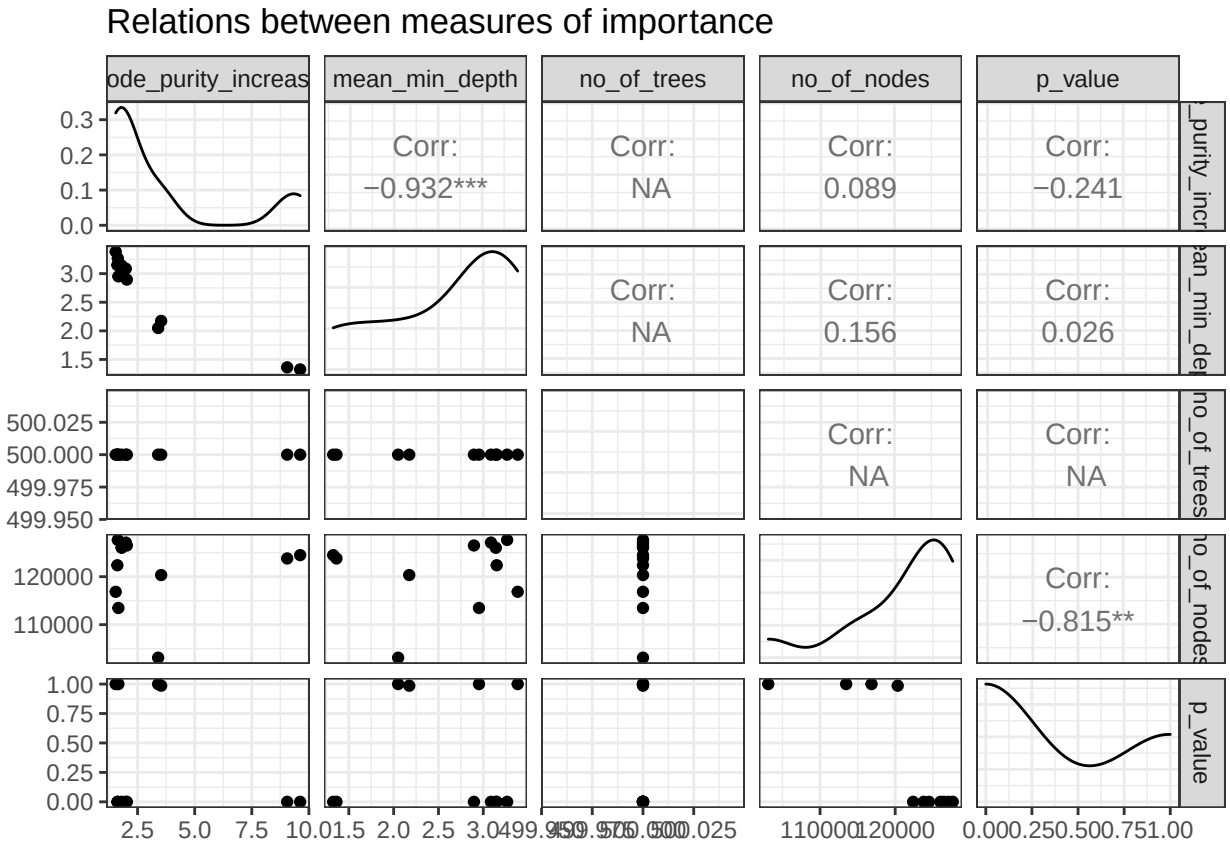
```
plot_multi_way_importance(importance_frame, size_measure = "no_of_nodes")
```




```
plot_multi_way_importance(importance_frame, x_measure = "mse_increase",
                          y_measure = "node_purity_increase",
                          size_measure = "p_value", no_of_labels = 5)
```




```
plot_importance_ggpairs(importance_frame)
```



Visualizing Random Forest Model

While powerful, random forest models can be difficult to visualize due to the number of decision trees that were aggregated and the number of variables. We can construct a dataset where only one variable changes at a time to understand the relationship between each predictor variable and AWC. For this document, I only included the effects of % sand, % silt, % clay, and OM on AWC.

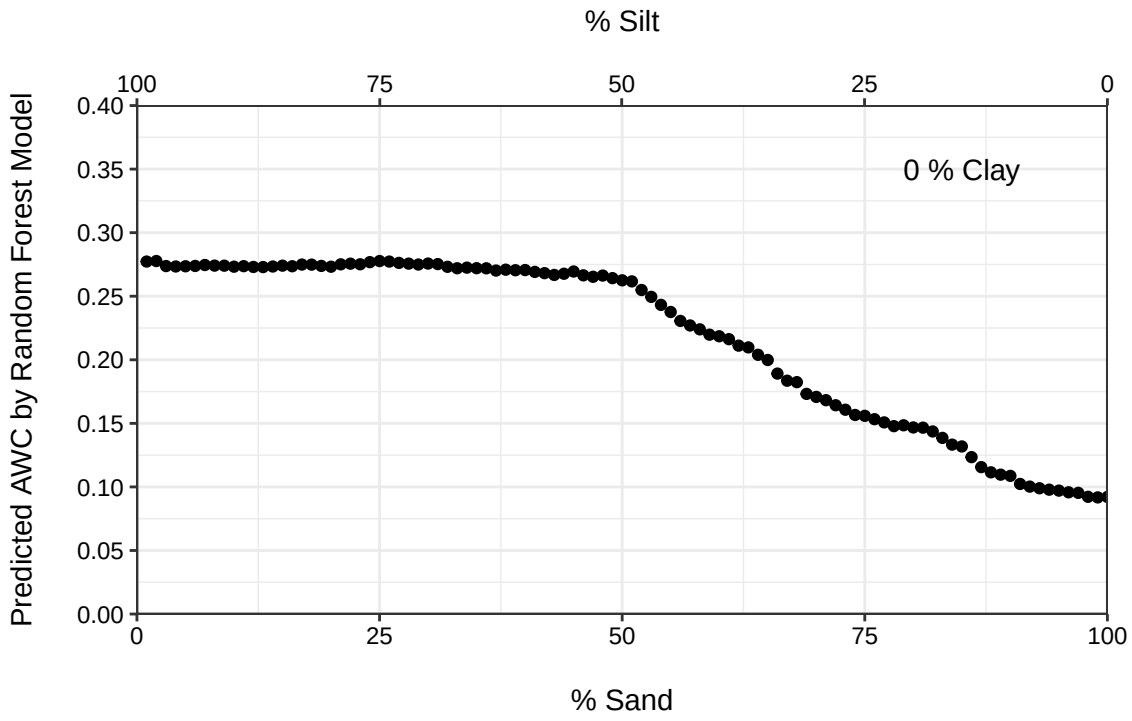
First we must use our Random Forest Model to predict AWC for our made up dataset.

```
Madeup.SHdata.forAWCmodel$AWC.predict = predict(RFAWC, Madeup.SHdata.forAWCmodel)
```

Plot predicted AWC vs. sand and silt content with 0% clay:

```
sand.silt.Madeup.SHdata.forAWCmodel<-filter(Madeup.SHdata.forAWCmodel,
                                            treatment=="Sand.Silt")

AWC.predictedbyRFsand<-ggplot(sand.silt.Madeup.SHdata.forAWCmodel, aes(x=sand, y=AWC.predict)) +
  geom_point()+
  labs(y="Predicted AWC by Random Forest Model\n")+
  scale_x_continuous("\n% Sand",expand = c(0,0),limits=c(0,100),
                     sec.axis = sec_axis(~ .-100, labels=absolutenumbers, name = "% Silt\n"))+
  scale_y_continuous(expand = c(0,0), breaks=c(0,.05,.1,.15,.2,.25,.3,.35,.4), limits=c(0,.4))+
  theme_bw()+
  theme(plot.title = element_text(hjust = 0.5))+
  theme(plot.title=element_text(face="bold"))+
  annotate("text", x=85, y=.35, label="0 % Clay")+
  theme(legend.title = element_text(face="bold"))+
```

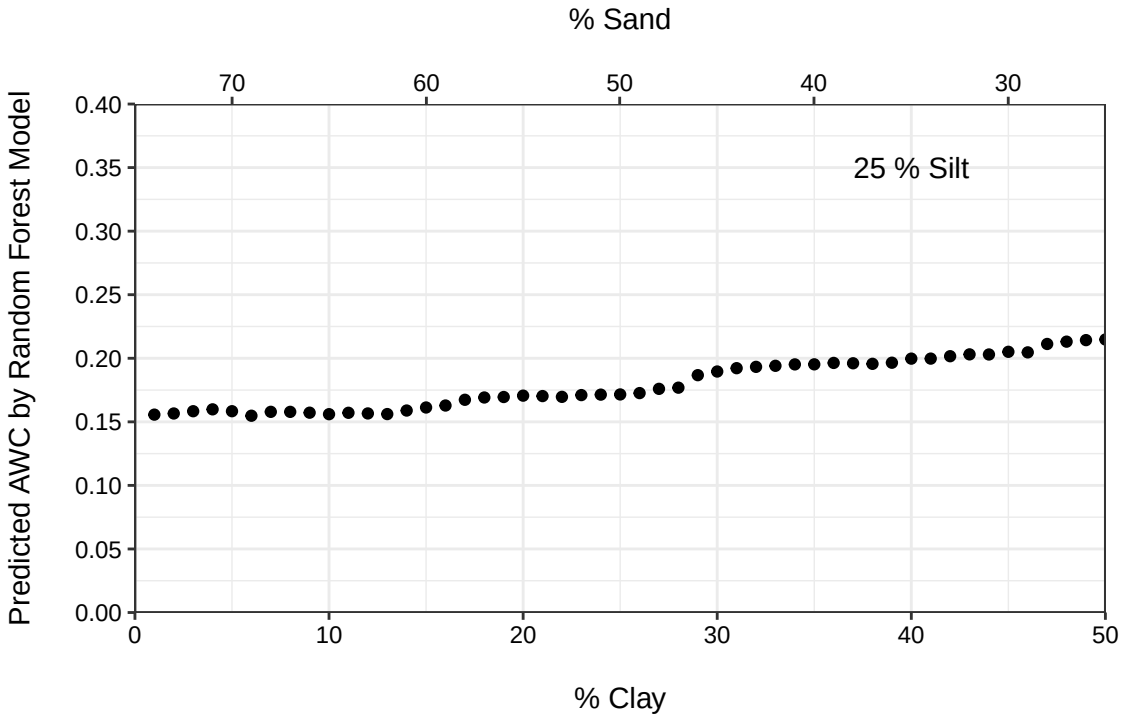
```
  theme(legend.key.width = unit(3, "line"))+
  theme(axis.text.x = element_text( color="black"))+
  theme(axis.text.y=element_text(color="black"))+
  theme(plot.margin = unit(c(1,1,1,1), "cm"))+
  theme(panel.background=element_rect(fill="transparent", color=NA),
        plot.background=element_rect(fill="transparent", color=NA))
AWC.predictedbyRFsand
```



Plot predicted AWC vs. sand and clay content with 25% silt:

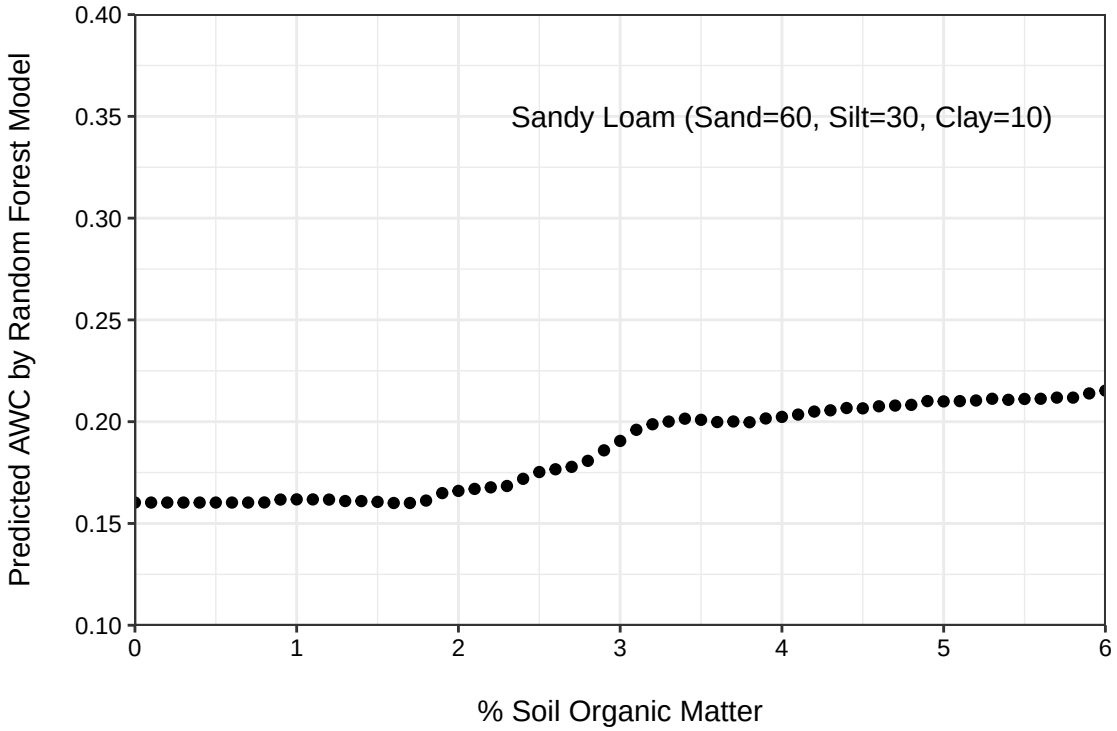
```
sand.clay.25silt.Madeup.SHdata.forAWCmodel<-filter(Madeup.SHdata.forAWCmodel, treatment=="Sand.Clay.25S

AWC.predictedbyRFclay<-ggplot(sand.clay.25silt.Madeup.SHdata.forAWCmodel, aes(x=clay, y=AWC.predict)) +
  geom_point()+
  labs(y="Predicted AWC by Random Forest Model\n")+
  scale_x_continuous("\n% Clay",expand = c(0,0),limits=c(0,50), sec.axis = sec_axis(~ .-75,
        labels=absolutenumbers, name = "% Sand\n"))+
  scale_y_continuous(expand = c(0,0), breaks=c(0,.05,.1,.15,.2,.25,.3,.35,.4), limits=c(0,.4))+
  theme_bw()+
  theme(plot.title = element_text(hjust = 0.5))+
  annotate("text", x=40, y=.35, label="25 % Silt")+
  theme(legend.title = element_text(face="bold"))+
  theme(legend.key.width = unit(3, "line"))+
  theme(axis.text.x = element_text(color="black"))+
  theme(axis.text.y=element_text(color="black"))+
  theme(plot.margin = unit(c(1,1,1,1), "cm"))+
```

```
  theme(panel.background=element_rect(fill="transparent", color=NA),
        plot.background=element_rect(fill="transparent", color=NA))
AWC.predictedbyRFclay
```

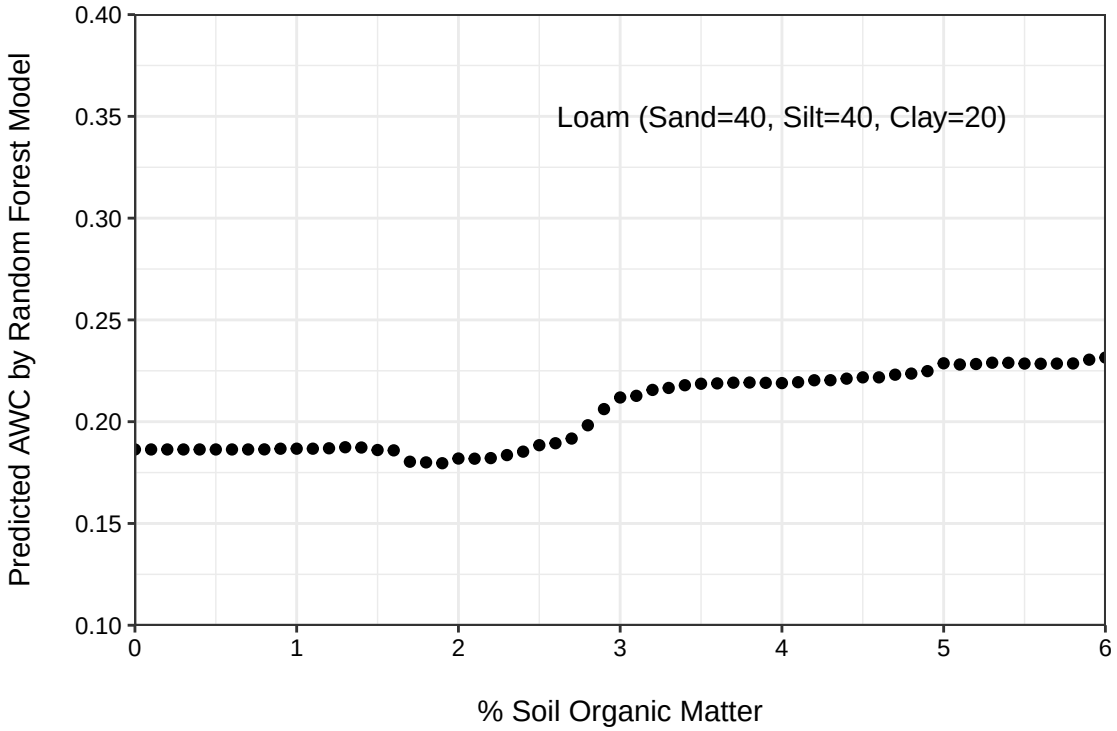


Plot predicted AWC vs. OM in a sandy loam soil:

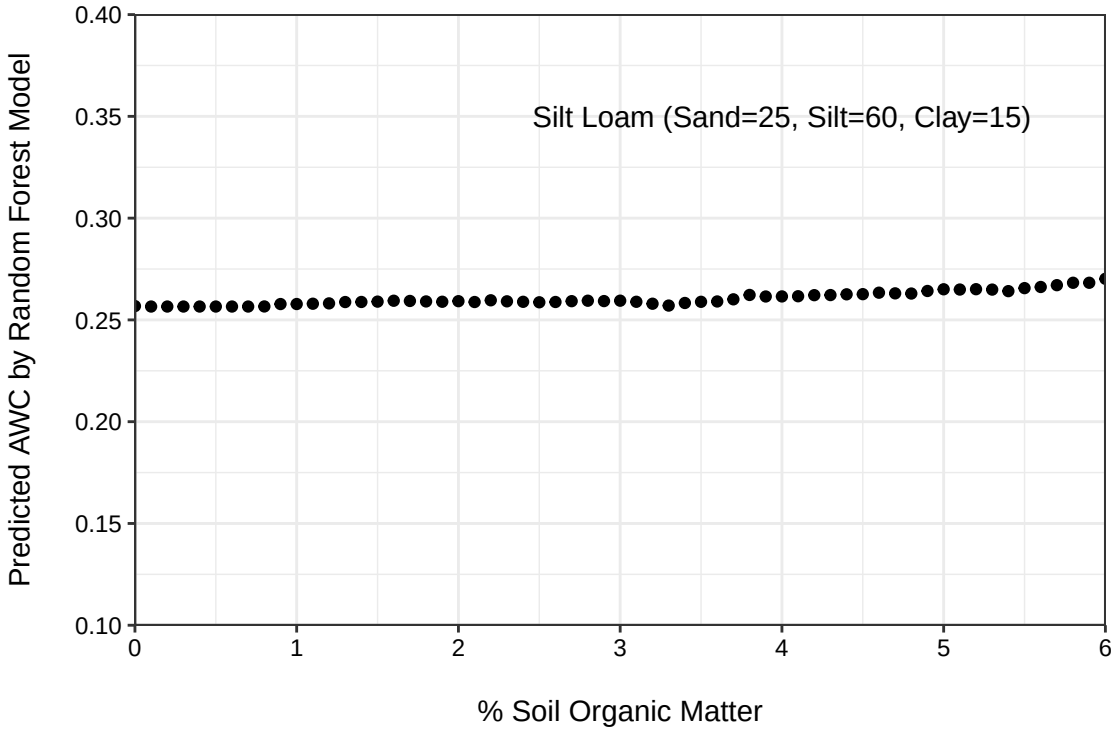
```
sandyloam.OM.Madeup.SHdata.forAWCmodel<-filter(Madeup.SHdata.forAWCmodel, treatment=="Sandy.loam.OM")

AWC.predictedbyRFOMsandyloam<-ggplot(sandyloam.OM.Madeup.SHdata.forAWCmodel, aes(x=SOM, y=AWC.predict))
  geom_point()+
  labs(y="Predicted AWC by Random Forest Model\n")+
  scale_x_continuous("\n% Soil Organic Matter",expand = c(0,0),limits=c(0,6))+
  scale_y_continuous(expand = c(0,0), breaks=c(.1,.15,.2,.25,.3,.35,.4), limits=c(.1,.4))+
  theme_bw()+
  theme(plot.title = element_text(hjust = 0.5))+
  theme(plot.title=element_text(face="bold"))+
  annotate("text", x=4, y=.35, label="Sandy Loam (Sand=60, Silt=30, Clay=10)")+
  theme(legend.title = element_text(face="bold"))+
  theme(legend.key.width = unit(3, "line"))+
  theme(axis.text.x = element_text(color="black"))+
  theme(axis.text.y=element_text(color="black"))+
  theme(plot.margin = unit(c(1,1,1,1), "cm"))+
  theme(panel.background=element_rect(fill="transparent", color=NA),
        plot.background=element_rect(fill="transparent", color=NA))
AWC.predictedbyRFOMsandyloam
```



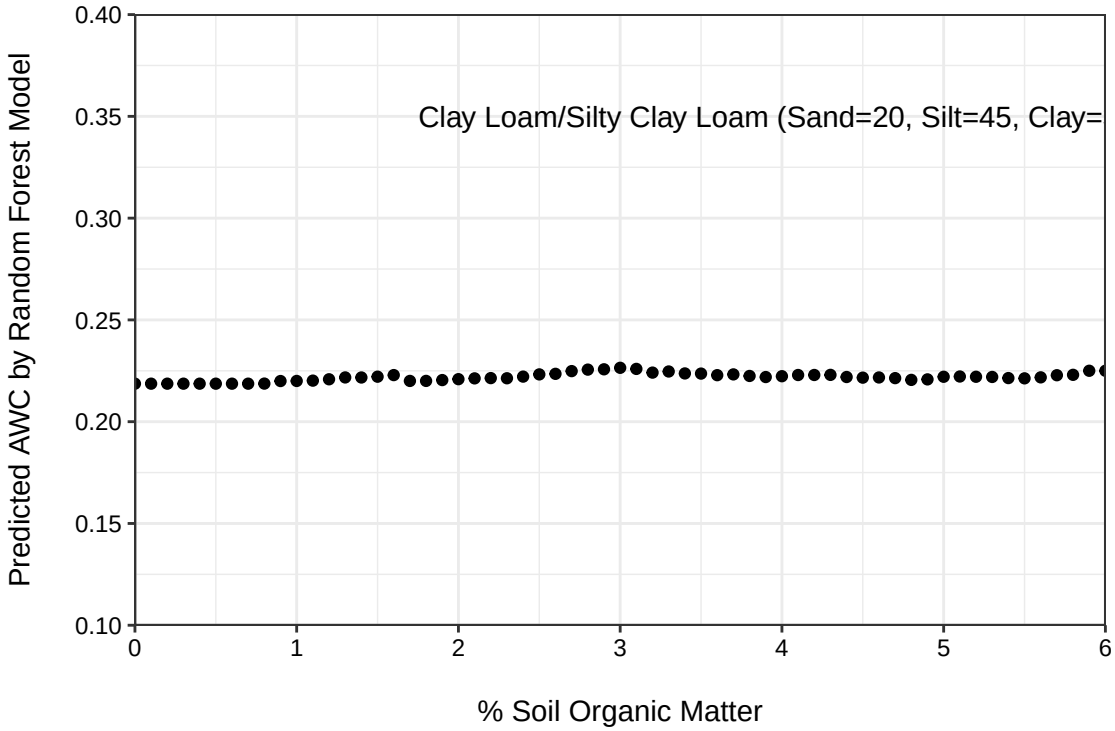
Plot predicted AWC vs. OM in a loam soil:



Plot predicted AWC vs. OM in a silt loam soil:

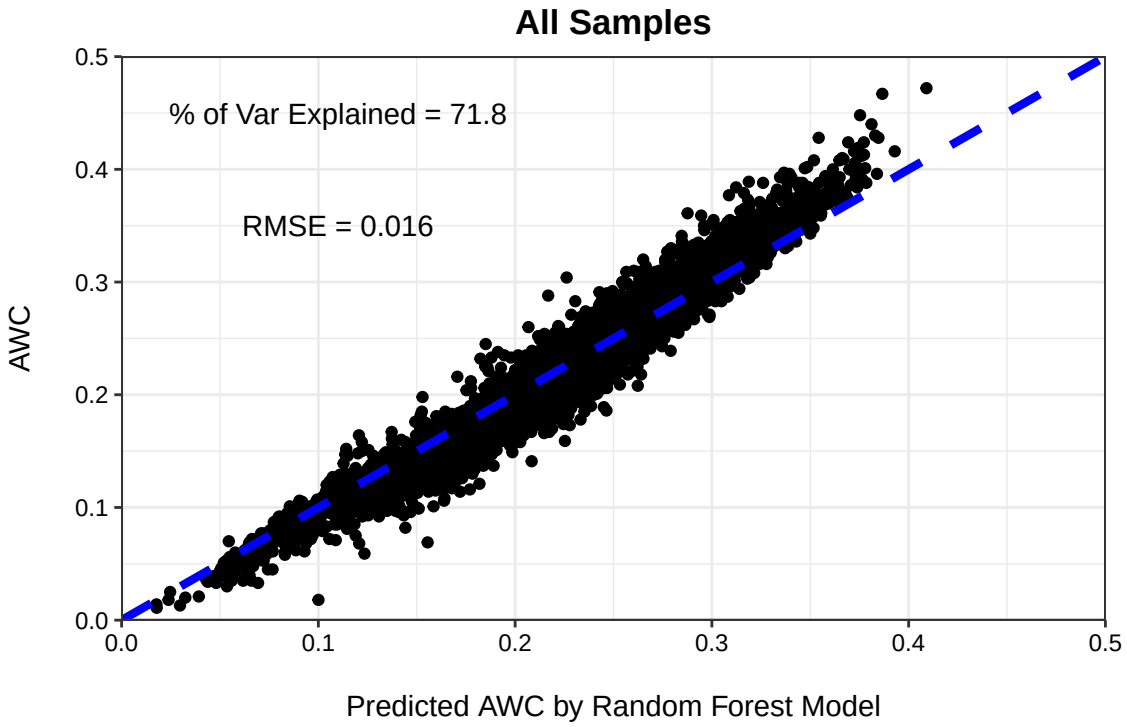


Plot predicted AWC vs. OM in a soil on the edge of clay loam and silty clay loam:



Compare Predicted and Observed AWC

1) All Samples:

```
SHdata$AWC.predict = predict(RFAWC, SHdata)

maxAWC<-max(SHdata$AWC)
minAWC<-min(SHdata$AWC)
rangeAWC<-maxAWC-minAWC
RMSE.AWC<-rmse(SHdata$AWC, SHdata$AWC.predict)
perRMSE.AWC<-100*(RMSE.AWC/rangeAWC)

AWC.predictedbyRF<-ggplot(SHdata, aes(x=AWC.predict, y=AWC)) +
  geom_point()+
  geom_abline(intercept=0, slope=1, color="blue", linetype="dashed", size=1.5)+
  ggtitle("All Samples")+
  labs(x="\nPredicted AWC by Random Forest Model")+
  labs(y="AWC\n")+
  scale_y_continuous(expand = c(0,0),breaks=c(0,.1,.2,.3,.4,.5), limits=c(0,.5))+
  scale_x_continuous(expand = c(0,0), breaks=c(0,.1,.2,.3,.4,.5), limits=c(0,.5))+
  theme_bw()+
  theme(plot.title = element_text(hjust = 0.5))+
  theme(plot.title=element_text(face="bold"))+
  annotate("text", x=.11, y=.35, label="RMSE = 0.016")+
  annotate("text",  x=.11, y=.45, label="% of Var Explained = 71.8")+
  theme(legend.title = element_text(face="bold"))+
```

```
  theme(legend.key.width = unit(3, "line"))+
  theme(axis.text.x = element_text(color="black"))+
  theme(axis.text.y=element_text(color="black"))+
  theme(plot.margin = unit(c(1,1,1,1), "cm"))+
  theme(panel.background=element_rect(fill="transparent", color=NA), plot.background=element_rect(fill=
AWC.predictedbyRF
```