This report covers the code for the comparison of benchmarks section from the AIM Annual report. For the purpose, methods and example reporting see the “Annual report code purpose and methods” report located on the Github. A brief summary of methods will be provided in this report. In addition, comments on any changes needed or important details will be located throughout the code.

Method 1 – Benchmarks are Known

This is using TerrADat Query Results. Save your csv file in your Documents with then name “Allyears_query_U”

If packages are not yet installed use function install.packages(“insert_package_name_here”

```
benchtool<-read.csv("~/Allyears_query_U.csv")
View(benchtool)
library(tidyverse)
library(dplyr)
library(formattable)
library(gridExtra)
library(grid)

benchtool$Tree<-
benchtool$Noxious.Tree.Cover.Pct.Any.Hit+benchtool$NonNoxious.Tree.Cover.Pct.Any.Hit
bench<- benchtool %>%
  select(Bare.Soil.Pct,Soil.Stability.All, Total.Litter.Cover.Pct.First.Hit,
         NonNoxious.Plant.Cover.Pct.Any.Hit, Grass.Cover.Pct.Any.Hit,
Forb.Cover.Pct.Any.Hit,

Shrub.Cover.Pct.Any.Hit,Tree,Sagebrush.Cover.Pct.Any.Hit,NonSagebrush.Shrub.Cover.Pct.Any.H
it,
         Average.Grass.Height.cm,Average.Forb.Height.cm,Noxious.Cover.Pct.Any.Hit,
         Number.Preferred.Forb.Species,Actual.Eco.Site)
```

The select function above displays all of the TerrADat Query Results which were used for calculations.

```
#set the benchmarks. "Tags" are the meeting or not meeting labels. For example,
##0-10% (technically 9.9 but not coded this way) is not meeting, 10-40% is meeting, and 40-
100% is not meeting
###tags for this example will be tags<-c("[NO)","[YES)","[NO)"). SADE is my ecological site
or strata.
###breaks are the ecological benchmarks. max and min will be used for graphing max being
the upper benchmark limit
###and min being the lower benchmark limit

SADE<-subset(bench,Actual.Eco.Site=="SD")
SADEBAREGROUNDbreaks<-c(0,10,40,100)
SADEBAREGROUNDtags<-c("[NO)","[YES)","[NO)")
```

```
SADEBAREGROUNDmax<-10
SADEBAREGROUNDmin<-40

SADESOILSTABILITYbreaks<-c(0,3,6)
SADESOILSTABILITYtags<-c("[NO)","[YES)")
SADESOILSTABILITYmax<-6
SADESOILSTABILITYmin<-3

SADELITTERbreaks<-c(0,5,20,100)
SADELITTERtags<-c("[NO)","[YES)","[NO)")
SADELITTERmax<-20
SADELITTERmin<-5

SADENATIVEbreaks<-c(0,20,40)
SADENATIVEtags<-c("[NO)","[YES)")
SADENATIVEmax<-40
SADENATIVEmin<-20

SADEGRASSbreaks<-c(0,5,19,100)
SADEGRASStags<-c("[NO)","[YES)","[NO)")
SADEGRASSmax<-19
SADEGRASSmin<-5

SADEFORBbreaks<-c(0,5,14,100)
SADEFORBtags<-c("[NO)","[YES)","[NO)")
SADEFORBmax<-14
SADEFORBmin<-5

SADESHRUBbreaks<-c(0,10,39,100)
SADESHRUBtags<-c("[NO)","[YES)","[NO)")
SADESHRUBmax<-39
SADESHRUBmin<-10

SADENONNATIVEbreaks<-c(0,19,100)
SADENONNATIVEtags<-c("[YES)","[NO)")
SADENONNATIVEmax<-19
SADENONNATIVEmin<-0

SADENUMBEROFFORBSbreaks<-c(0,5,100)
SADENUMBEROFFORBStags<-c("[NO)","[YES)")
SADENUMBEROFFORBSmax<-100
SADENUMBEROFFORBSmin<-5
```

```
#compiling the data for the table. no output will be seen until the  table section!
#bareground
group_tags<-cut(na.omit(SADE$Bare.Soil.Pct),
breaks=SADEBAREGROUNDbreaks,include.lowest=TRUE, right=FALSE,labels=SADEBAREGROUNDtags)
group_tags<-data.frame(group_tags)
SADEPBGY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SADEPBG<-SADEPBGY[!duplicated(SADEPBGY$sumY),]


#soil stability
group_tags<-cut(na.omit(SADE$Soil.Stability.All),
breaks=SADESOILSTABILITYbreaks,include.lowest=TRUE,
right=FALSE,labels=SADESOILSTABILITYtags)
group_tags<-data.frame(group_tags)
SADESASY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SADESAS<-SADESASY[!duplicated(SADESASY$sumY),]


#Litter
group_tags<-cut(na.omit(SADE$Total.Litter.Cover.Pct.First.Hit),
breaks=SADELITTERbreaks,include.lowest=TRUE, right=FALSE,labels=SADELITTERtags)
group_tags<-data.frame(group_tags)
SADELY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SADEL<-SADELY[!duplicated(SADELY$sumY),]


#native
group_tags<-cut(na.omit(SADE$NonNoxious.Plant.Cover.Pct.Any.Hit),
breaks=SADENATIVEbreaks,include.lowest=TRUE, right=FALSE,labels=SADENATIVEtags)
group_tags<-data.frame(group_tags)
SADENY<-group_tags %>%
```

%>% indicates we are continuing the line of code. Mutate() will create a new row in our data frame. Filter() allows us to select only the values we are interested in (in this case values that are meeting the benchmarks). ! means is not and is used in conjunction with the [row,column] format in R to remove duplicated values. At this point we are only interested in one value (sumY which = the sum of plots meeting benchmark/the sum of all plots) and so we can remove all rows except for one to call on the value determined later.

```
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SADEN<-SADENY[!duplicated(SADENY$sumY),]

#percent cover grass
group_tags<-cut(na.omit(SADE$Grass.Cover.Pct.Any.Hit),
breaks=SADEGRASSbreaks,include.lowest=TRUE, right=FALSE,labels=SADEGRASStags)
group_tags<-data.frame(group_tags)
SADEGY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SADEG<-SADEGY[!duplicated(SADEGY$sumY),]

#percent cover forb
group_tags<-cut(na.omit(SADE$Forb.Cover.Pct.Any.Hit),
breaks=SADEFORBbreaks,include.lowest=TRUE, right=FALSE,labels=SADEFORBtags)
group_tags<-data.frame(group_tags)
SADEFY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SADEF<-SADEFY[!duplicated(SADEFY$sumY),]

#percent cover shrub
group_tags<-cut(na.omit(SADE$Shrub.Cover.Pct.Any.Hit),
breaks=SADESHRUBbreaks,include.lowest=TRUE, right=FALSE,labels=SADESHRUBtags)
group_tags<-data.frame(group_tags)
SADESY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SADES<-SADESY[!duplicated(SADESY$sumY),]


#nonnative
group_tags<-cut(na.omit(SADE$Noxious.Cover.Pct.Any.Hit),
breaks=SADENONNATIVEbreaks,include.lowest=TRUE, right=FALSE,labels=SADENONNATIVEtags)
```

```
group_tags<-data.frame(group_tags)
SADENNY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SADENN<-SADENNY[!duplicated(SADENNY$sumY),]

#number of forbs
group_tags<-cut(na.omit(SADE$Number.Preferred.Forb.Species),
breaks=SADENUMBEROFFORBSbreaks,include.lowest=TRUE,
right=FALSE,labels=SADENUMBEROFFORBStags)
group_tags<-data.frame(group_tags)
SADENFY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SADENF<-SADENFY[!duplicated(SADENFY$sumY),]
```

If you have more benchmark indicators or different indicators just copy this step and the initial step of setting the benchmark values and change to your benchmark indicator name and values. Be sure to change/add the name to the Category vector in the #table section below. If you have less, remove those sections of code and remove the name from the Category vector. Example of these modifications can be seen in the sage step (SAS) stratum

```
#table
##note stratablanks<-rep("",8) is based on having 9 benchmark categories, therefore if you
have
##5 categories then change 8 to 4
stratablanks<-rep("",8)
Strata<-c("SD",stratablanks)
Category<- c("Percent Bare Ground", "Soil Aggregate Stability","Litter","Nonnoxious
Species","Percent Grass Cover",
             "Percent Forb Cover","Percent Shrub Cover", "Percent Noxious Species","Number
of Forbs")
Percent.Meeting<-
c(round(SADEPBG$sumY,digits=2),round(SADESAS$sumY,digits=2),round(SADEL$sumY,digits=2),roun
d(SADEN$sumY,digits=2),round(SADEG$sumY,digits=2),round(SADEF$sumY,digits=2),

round(SADES$sumY,digits=2),round(SADENN$sumY,digits=2),round(SADENF$sumY,digits=2))
SADEdataframe<-data.frame(Strata,Category,Percent.Meeting)
formattable(SADEdataframe)

#to add another strata. SAST=sage steppe
SAST<-subset(bench,Actual.Eco.Site=="SS")
SASTBAREGROUNDbreaks<-c(0,10,35,100)
```

```
SASTBAREGROUNDtags<-c("[NO)","[YES)","[NO)")
SASTBAREGROUNDmax<-35
SASTBAREGROUNDmin<-10

SASTSOILSTABILITYbreaks<-c(0,4,6)
SASTSOILSTABILITYtags<-c("[NO)","[YES)")
SASTSOILSTABILITYmax<-6
SASTSOILSTABILITYmin<-4

SASTLITTERbreaks<-c(0,10,40,100)
SASTLITTERtags<-c("[NO)","[YES)","[NO)")
SASTLITTERmax<-40
SASTLITTERmin<-10

#SASTNATIVEbreaks=NO BENCHMARK SET, THEREFORE OMITTING
#SASTNATIVEtags=NO BENCHMARK SET, THEREFORE OMITTING

SASTGRASSbreaks<-c(0,10,100)
SASTGRASStags<-c("[NO)","[YES)")
SASTGRASSmax<-100
SASTGRASSmin<-10

SASTFORBbreaks<-c(0,5,40,100)
SASTFORBtags<-c("[NO)","[YES)","[NO)")
SASTFORBmax<-40
SASTFORBmin<-5

#SASTSHRUBbreaks=NO BENCHMARK SET, THEREFORE OMITTING
#SASTSHRUBtags=NO BENCHMARK SET, THEREFORE OMITTING

SASTNONNATIVEbreaks<-c(0,10,100)
SASTNONNATIVEtags<-c("[YES)","[NO)")
SASTNONNATIVEmax<-10
SASTNONNATIVEmin<-0

SASTNUMBEROFFORBSbreaks<-c(0,5,100)
SASTNUMBEROFFORBStags<-c("[NO)","[YES)")
SASTNUMBEROFFORBmax<-100
SASTNUMBEROFFORBmin<-5

#bareground
group_tags<-cut(na.omit(SAST$Bare.Soil.Pct),
breaks=SASTBAREGROUNDbreaks,include.lowest=TRUE, right=FALSE,labels=SASTBAREGROUNDtags)
```

```
group_tags<-data.frame(group_tags)
SASTPBGY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SASTPBG<-SASTPBGY[!duplicated(SASTPBGY$sumY),]


#soil stability
group_tags<-cut(na.omit(SAST$Soil.Stability.All),
breaks=SASTSOILSTABILITYbreaks,include.lowest=TRUE,
right=FALSE,labels=SASTSOILSTABILITYtags)
group_tags<-data.frame(group_tags)
SASTSASY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SASTSAS<-SASTSASY[!duplicated(SASTSASY$sumY),]


#Litter
group_tags<-cut(na.omit(SAST$Total.Litter.Cover.Pct.First.Hit),
breaks=SASTLITTERbreaks,include.lowest=TRUE, right=FALSE,labels=SASTLITTERtags)
group_tags<-data.frame(group_tags)
SASTLY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SASTL<-SASTLY[!duplicated(SASTLY$sumY),]


#native
##NONE SET FOR THIS STRATA

#percent cover grass
group_tags<-cut(na.omit(SAST$Grass.Cover.Pct.Any.Hit),
breaks=SASTGRASSbreaks,include.lowest=TRUE, right=FALSE,labels=SASTGRASStags)
group_tags<-data.frame(group_tags)
SASTGY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
```

```
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SASTG<-SASTGY[!duplicated(SASTGY$sumY),]

#percent cover forb
group_tags<-cut(na.omit(SAST$Forb.Cover.Pct.Any.Hit),
breaks=SASTFORBbreaks,include.lowest=TRUE, right=FALSE,labels=SASTFORBtags)
group_tags<-data.frame(group_tags)
SASTFY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SASTF<-SASTFY[!duplicated(SASTFY$sumY),]

#percent cover shrub
##NONE SET FOR THIS STRATA


#nonnative
group_tags<-cut(na.omit(SAST$Noxious.Cover.Pct.Any.Hit),
breaks=SASTNONNATIVEbreaks,include.lowest=TRUE, right=FALSE,labels=SASTNONNATIVEtags)
group_tags<-data.frame(group_tags)
SASTNNY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SASTNN<-SASTNNY[!duplicated(SASTNNY$sumY),]

#number of forbs
group_tags<-cut(na.omit(SAST$Number.Preferred.Forb.Species),
breaks=SASTNUMBEROFFORBSbreaks,include.lowest=TRUE,
right=FALSE,labels=SASTNUMBEROFFORBStags)
group_tags<-data.frame(group_tags)
SASTNFY<-group_tags %>%
  mutate(sumT=sum(length(group_tags))) %>%
  filter(group_tags=="[YES)") %>%
  mutate(sum=sum(length(group_tags))) %>%
  mutate(sumY=(sum/sumT)*100)
SASTNF<-SASTNFY[!duplicated(SASTNFY$sumY),]
```

```
#table, note since a native benchmark was omitted from this strata the only thing that is
changed is Percent.Meeting
##SASTN$...is removed and replaced with "N/A", SAME WITH SHRUBCOVER
stratablanks<-rep("",8)
Strata<-c("","SS",stratablanks)
Category<- c("","Percent Bare Ground", "Soil Aggregate Stability","Litter","Nonnoxious
Species","Percent Grass Cover",
             "Percent Forb Cover","Percent Shrub Cover", "Percent Noxious Species","Number
of Forbs")
Percent.Meeting<-
c("",round(SASTPBG$sumY,digits=2),round(SASTSAS$sumY,digits=2),round(SASTL$sumY,digits=2),"
N/A",round(SASTG$sumY,digits=2),round(SASTF$sumY,digits=2),
                  "N/A",round(SASTNN$sumY,digits=2),round(SASTNF$sumY,digits=2))
SASTdataframe<-data.frame(Strata,Category,Percent.Meeting)
SSdataframe<-rbind.data.frame(SADEdataframe,SASTdataframe)
formattable(SSdataframe)

#to graph
##SADE (salt desert)
B<-ggplot(SADE,aes(x=1,y=Bare.Soil.Pct))
B2<-B+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SADEBAREGROUNDmin,ymax=SADEBAREGROUNDmax),
            fill="palegreen1",alpha=0.03)
B3<-B2+ggtitle("Average Percent Bare Soil ")+labs(x="SD Plots",y="Average % Bare
Soil")+theme(axis.text.x=element_blank())


L<-ggplot(SADE,aes(x=1,y=Total.Litter.Cover.Pct.First.Hit))
L2<-L+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SADELITTERmin,ymax=SADELITTERmax),
            fill="palegreen1",alpha=0.03)
L3<-L2+ggtitle("Average Percent Litter Cover ")+labs(x="SD Plots",y="Average % Litter
Cover")+theme(axis.text.x=element_blank())


NN<-ggplot(SADE,aes(x=1,y=NonNoxious.Plant.Cover.Pct.Any.Hit))
NN2<-NN+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SADENATIVEmin,ymax=SADENATIVEmax),
            fill="palegreen1",alpha=0.03)
NN3<-NN2+ggtitle("Average Percent Nonnoxious Species ")+labs(x="SD Plots",y="Average %
Nonnoxious Species")+theme(axis.text.x=element_blank())
```

We will be assigning graphs a name ex. B3 in order to use the grid_arrange() function to compile the graphs into one plot. If you want to view one specific plot just type the name into the console.

```
G<-ggplot(SADE,aes(x=1,y=Grass.Cover.Pct.Any.Hit))
G2<-G+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SADEGRASSmin,ymax=SADEGRASSmax),
            fill="palegreen1",alpha=0.03)
G3<-G2+ggtitle("Average Percent Grass Cover ")+labs(x="SD Plots",y="Average % Grass
Cover")+theme(axis.text.x=element_blank())

F<-ggplot(SADE,aes(x=1,y=Forb.Cover.Pct.Any.Hit))
F2<-F+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SADEFORBmin,ymax=SADEFORBmax),
            fill="palegreen1",alpha=0.03)
F3<-F2+ggtitle("Average Percent Forb Cover ")+labs(x="SD Plots",y="Average % Forb
Cover")+theme(axis.text.x=element_blank())


S<-ggplot(SADE,aes(x=1,y=Shrub.Cover.Pct.Any.Hit))
S2<-S+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SADESHRUBmin,ymax=SADESHRUBmax),
            fill="palegreen1",alpha=0.03)
S3<-S2+ggtitle("Average Percent Shrub Cover ")+labs(x="SD Plots",y="Average % Shrub
Cover")+theme(axis.text.x=element_blank())


N<-ggplot(SADE,aes(x=1,y=Noxious.Cover.Pct.Any.Hit))
N2<-N+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SADENONNATIVEmin,ymax=SADENONNATIVEmax),
            fill="palegreen1",alpha=0.03)
N3<-N2+ggtitle("Average Percent Noxious Species ")+labs(x="SD Plots",y="Average % Noxious
Species")+theme(axis.text.x=element_blank())

NF<-ggplot(SADE,aes(x=1,y=Number.Preferred.Forb.Species))
NF2<-NF+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SADENUMBEROFFORBSmin,ymax=SADENUMBEROFFORBSmax),
            fill="palegreen1",alpha=0.03)
NF3<-NF2+ggtitle("Number of Preferred Forbs ")+labs(x="SD Plots",y="Number of Preferred
Forbs")+theme(axis.text.x=element_blank())

SAS<-ggplot(SADE,aes(x=1,y=Soil.Stability.All))
SAS2<-SAS+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SADESOILSTABILITYmin,ymax=SADESOILSTABILITYmax),
            fill="palegreen1",alpha=0.03)
SAS3<-SAS2+ggtitle("Number of Preferred Forbs ")+labs(x="SD Plots",y="Number of Preferred
Forbs")+theme(axis.text.x=element_blank())
```

```
grid.arrange(B3,SAS3,L3,NN3,G3,F3,S3,N3,NF3,nrow=2,
             top = textGrob("Display of Plots within Benchmark by Benchmark
Category",gp=gpar(fontsize=20,font=3)))

#to graph
##SAST (sage steppe)
B<-ggplot(SAST,aes(x=1,y=Bare.Soil.Pct))
B2<-B+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SASTBAREGROUNDmin,ymax=SASTBAREGROUNDmax),
            fill="palegreen1",alpha=0.03)
B3<-B2+ggtitle("Average Percent Bare Soil ")+labs(x="SS Plots",y="Average % Bare
Soil")+theme(axis.text.x=element_blank())


L<-ggplot(SAST,aes(x=1,y=Total.Litter.Cover.Pct.First.Hit))
L2<-L+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SASTLITTERmin,ymax=SASTLITTERmax),
            fill="palegreen1",alpha=0.03)
L3<-L2+ggtitle("Average Percent Litter Cover ")+labs(x="SS Plots",y="Average % Litter
Cover")+theme(axis.text.x=element_blank())


#N<- OMITTING NO NATIVE BENCHMARK SET

G<-ggplot(SAST,aes(x=1,y=Grass.Cover.Pct.Any.Hit))
G2<-G+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SASTGRASSmin,ymax=SASTGRASSmax),
            fill="palegreen1",alpha=0.03)
G3<-G2+ggtitle("Average Percent Grass Cover ")+labs(x="SS Plots",y="Average % Grass
Cover")+theme(axis.text.x=element_blank())

F<-ggplot(SAST,aes(x=1,y=Forb.Cover.Pct.Any.Hit))
F2<-F+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SASTFORBmin,ymax=SASTFORBmax),
            fill="palegreen1",alpha=0.03)
F3<-F2+ggtitle("Average Percent Forb Cover ")+labs(x="SS Plots",y="Average % Forb
Cover")+theme(axis.text.x=element_blank())


#s<-OMITTING NO SHRUB BENCHMARK SET

N<-ggplot(SAST,aes(x=1,y=Noxious.Cover.Pct.Any.Hit))
```

```
N2<-N+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SASTNONNATIVEmin,ymax=SASTNONNATIVEmax),
            fill="palegreen1",alpha=0.03)
N3<-N2+ggtitle("Average Percent Noxious Species ")+labs(x="SS Plots",y="Average % Noxious
Species")+theme(axis.text.x=element_blank())

NF<-ggplot(SAST,aes(x=1,y=Number.Preferred.Forb.Species))
NF2<-NF+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SASTNUMBEROFFORBmin,ymax=SASTNUMBEROFFORBmax),
            fill="palegreen1",alpha=0.03)
NF3<-NF2+ggtitle("Number of Preferred Forbs ")+labs(x="SS Plots",y="Number of Preferred
Forbs")+theme(axis.text.x=element_blank())

SAS<-ggplot(SAST,aes(x=1,y=Soil.Stability.All))
SAS2<-SAS+geom_jitter()+
  geom_rect(aes(xmin=-Inf,xmax=Inf,ymin=SASTSOILSTABILITYmin,ymax=SASTSOILSTABILITYmax),
            fill="palegreen1",alpha=0.03)
SAS3<-SAS2+ggtitle("Soil Stability ")+labs(x="SS Plots",y="Average Soil
Stability")+theme(axis.text.x=element_blank())

grid.arrange(B3,SAS3,L3,G3,F3,N3,NF3,nrow=2,
             top = textGrob("Display of Plots within Benchmark by Benchmark
Category",gp=gpar(fontsize=20,font=3)))
```

Method 2- Benchmark's not yet assigned

```
#this code creates benchmarks for all indicators (indicators= baresoil, rock, tree, shrub,
soil stability, forb number, forb cover,
#grass cover, forb height, grass height, succulent, litter, noxious species, nonnoxious
species and succulent cover) then produces
#tables for indicators met by each plot, percent of plots meeting each indicator by stratum
and percent of plots being met by allotment

library(tidyverse)
library(dplyr)
library(ggrepel)
library(formattable)

benchtool<-read.csv("~/Allyears_query.csv")
plots<-read.csv("~/Allyears_plots.csv")
benchtool$Actual.Eco.Site<-
plots$Actual.Eco.Site[match(benchtool$Primary.Key,plots$PrimaryKey)]
```

Again, ensure your files are saved to Documents with the same file name as in the code "Allyears_query_U"… Benchtool is the Query Results and plots is plots from TerrADat. Match() is used to assign my Acutal.Eco.Site to other csvs by "matching" primary keys.

```
benchtool$plotID<-plots$PlotID[match(benchtool$Primary.Key,plots$PrimaryKey)]
benchtool$Allotment<-plots$Allotment.Name[match(benchtool$Primary.Key,plots$PrimaryKey)]
View(benchtool)


bench<-benchtool[-c(2:148)]
bench$baresoil<-benchtool$Bare.Soil.Pct
bench$soilstability<-benchtool$Soil.Stability.All
bench$litter<-benchtool$Total.Litter.Cover.Pct.First.Hit
bench$native<-benchtool$NonNoxious.Plant.Cover.Pct.Any.Hit
bench$grasscover<-benchtool$Grass.Cover.Pct.Any.Hit
bench$forbcover<-benchtool$Forb.Cover.Pct.Any.Hit
bench$shrubcover<-benchtool$Shrub.Cover.Pct.Any.Hit
bench$treecover<-
benchtool$Noxious.Tree.Cover.Pct.Any.Hit+benchtool$NonNoxious.Tree.Cover.Pct.Any.Hit
bench$sagecover<-benchtool$Sagebrush.Cover.Pct.Any.Hit
bench$nonsagecover<-benchtool$NonSagebrush.Shrub.Cover.Pct.Any.Hit
bench$grassheight<-benchtool$Average.Grass.Height.cm
bench$sageheight<-benchtool$Average.Sagebrush.Height.cm
bench$forb.height<-benchtool$Average.Forb.Height.cm
bench$nonnative<-benchtool$Noxious.Cover.Pct.Any.Hit
bench$forbnumber<-benchtool$Number.Preferred.Forb.Species
bench$Actual.Eco.Site<-benchtool$Actual.Eco.Site
bench$rock<-benchtool$Rock.Cover.Pct.First.Hit
bench$succulent<-
benchtool$NonNoxious.Succulent.Cover.Pct.Any.Hit+benchtool$Noxious.Succulent.Cover.Pct.Any.
Hit
bench$gaps<-benchtool$Gap.Cover.25.Plus.Pct
bench$plotID<-benchtool$plotID
bench$allotment<-benchtool$Allotment

#BH
BH<-subset(bench,Actual.Eco.Site=="BH")
BHmean<-mean(na.omit(BH$baresoil))
BHsd<-sd(na.omit(BH$baresoil))
breaks<-c(-Inf,unique(c(BHmean-(2*BHsd),BHmean+(2*BHsd))),Inf)
tags<-c("0","1","0")
BH$BSgroup_tags<-cut(BH$baresoil, breaks=breaks,include.lowest=TRUE,
right=FALSE,labels=tags)
summary(BH$BSgroup_tags)
BH$BSgroup_tags<-as.numeric(as.character(BH$BSgroup_tags))

BH<-BH %>%
```

Here I am reassigning TerrADat column names to make coding easier. This section can also be reviewed to understand which TerrADat Query Result columns are used in the calculations.

Subset() is used to select the stratum of interest. Calculations are assigned names using <- . Breaks assigns our benchmarks values (which are within +/- 2 sds of the mean). Tags assigns meeting (1) or not meeting (0). If the break values are not unique (this happens if the mean value is 0) you will need to change tags. See the nonnative section for an example of this. In that case tags would be c("1","0"). If none of the mean values are zero, no changes need to be made to the code.

```
   mutate(baresoil.bin=sum(na.omit(BSgroup_tags))/length(na.omit(BSgroup_tags)))

BHSSmean<-mean(na.omit(BH$soilstability))
BHSSsd<-sd(na.omit(BH$soilstability))
breaks<-c(-Inf,unique(c(BHSSmean-(2*BHSSsd),BHSSmean+(2*BHSSsd))),Inf)
breaks
tags<-c("0","1","0")
BH$SSgroup_tags<-cut(BH$soilstability, breaks=breaks,include.lowest=TRUE,
right=FALSE,labels=tags)
summary(BH$SSgroup_tags)
BH$SSgroup_tags<-as.numeric(as.character(BH$SSgroup_tags))

BH<-BH %>%
  mutate(soilstabilitybin=sum(na.omit(SSgroup_tags))/length(na.omit(SSgroup_tags)))

BHNmean<-mean(na.omit(BH$native))
BHNsd<-sd(na.omit(BH$native))
breaks<-c(-Inf,unique(c(BHNmean-(2*BHNsd),BHNmean+(2*BHNsd))),Inf)
tags<-c("0","1","0")
BH$Ngroup_tags<-cut(BH$native, breaks=breaks,include.lowest=TRUE, right=FALSE,labels=tags)
summary(BH$Ngroup_tags)
BH$Ngroup_tags<-as.numeric(as.character(BH$Ngroup_tags))

BH<-BH %>%
  mutate(nativebin=sum(na.omit(Ngroup_tags))/length(na.omit(Ngroup_tags)))


BHGmean<-mean(na.omit(BH$grasscover))
BHGsd<-sd(na.omit(BH$grasscover))
breaks<-c(-Inf,unique(c(BHGmean-(2*BHGsd),BHGmean+(2*BHGsd))),Inf)
tags<-c("0","1","0")
BH$Ggroup_tags<-cut(BH$grasscover, breaks=breaks,include.lowest=TRUE,
right=FALSE,labels=tags)
summary(BH$Ggroup_tags)
BH$Ggroup_tags<-as.numeric(as.character(BH$Ggroup_tags))

BH<-BH %>%
  mutate(grasscoverbin=sum(na.omit(Ggroup_tags))/length(na.omit(Ggroup_tags)))


BHFmean<-mean(na.omit(BH$forbcover))
BHFsd<-sd(na.omit(BH$forbcover))
breaks<-c(-Inf,unique(c(BHFmean-(2*BHFsd),BHFmean+(2*BHFsd))),Inf)
```

Mutate() step is adding a column with the sum of plots meeting benchmark/the total number of plots

```
tags<-c("0","1","0")
BH$Fgroup_tags<-cut(BH$forbcover, breaks=breaks,include.lowest=TRUE,
right=FALSE,labels=tags)
summary(BH$Fgroup_tags)
BH$Fgroup_tags<-as.numeric(as.character(BH$Fgroup_tags))

BH<-BH %>%
  mutate(forbcoverbin=sum(na.omit(Fgroup_tags))/length(na.omit(Fgroup_tags)))

BHSmean<-mean(na.omit(BH$shrubcover))
BHSsd<-sd(na.omit(BH$shrubcover))
breaks<-c(-Inf,unique(c(BHSmean-(2*BHSsd),BHSmean+(2*BHSsd))),Inf)
tags<-c("0","1","0")
BH$Sgroup_tags<-cut(BH$shrubcover, breaks=breaks,include.lowest=TRUE,
right=FALSE,labels=tags)
summary(BH$Sgroup_tags)
BH$Sgroup_tags<-as.numeric(as.character(BH$Sgroup_tags))

BH<-BH %>%
  mutate(shrubcoverbin=sum(na.omit(Sgroup_tags))/length(na.omit(Sgroup_tags)))

BHTmean<-mean(na.omit(BH$treecover))
BHTsd<-sd(na.omit(BH$treecover))
breaks<-c(-Inf,unique(c(BHTmean-(2*BHTsd),BHTmean+(2*BHTsd))),Inf)
tags<-c("0","1","0")
BH$Tgroup_tags<-cut(BH$treecover, breaks=breaks,include.lowest=TRUE,
right=FALSE,labels=tags)
summary(BH$Tgroup_tags)
BH$Tgroup_tags<-as.numeric(as.character(BH$Tgroup_tags))

BH<-BH %>%
  mutate(treecoverbin=sum(na.omit(Tgroup_tags))/length(na.omit(Tgroup_tags)))

BHGHmean<-mean(na.omit(BH$grassheight))
BHGHsd<-sd(na.omit(BH$grassheight))
breaks<-c(-Inf,BHGHmean-BHGHsd,BHGHmean+BHGHsd,Inf)
tags<-c("0","1","0")
BH$GHgroup_tags<-cut(BH$grassheight, breaks=breaks,include.lowest=TRUE,
right=FALSE,labels=tags)
summary(BH$GHgroup_tags)
BH$GHgroup_tags<-as.numeric(as.character(BH$GHgroup_tags))

BH<-BH %>%
```

```
  mutate(grassheightbin=sum(na.omit(GHgroup_tags))/length(na.omit(GHgroup_tags)))

BHFHmean<-mean(na.omit(BH$forb.height))
BHFHsd<-sd(na.omit(BH$forb.height))
breaks<-c(-Inf,unique(c(BHFHmean-(2*BHFHsd),BHFHmean+(2*BHFHsd))),Inf)
tags<-c("0","1","0")
BH$FHgroup_tags<-cut(BH$forb.height, breaks=breaks,include.lowest=TRUE,
right=FALSE,labels=tags)
summary(BH$FHgroup_tags)
BH$FHgroup_tags<-as.numeric(as.character(BH$FHgroup_tags))

BH<-BH %>%
  mutate(forb.heightbin=sum(na.omit(FHgroup_tags))/length(na.omit(FHgroup_tags)))

#since only 3 bins, had to change "tags" to two categories
BHNNmean<-mean(na.omit(BH$nonnative))
BHNNsd<-sd(na.omit(BH$nonnative))
breaks<-c(-Inf,unique(BHNNmean-BHNNsd,BHNNmean+BHNNsd),Inf)
breaks
tags<-c("0","1")
BH$NNgroup_tags<-cut(BH$nonnative, breaks=breaks,include.lowest=TRUE,
right=FALSE,labels=tags)
summary(BH$NNgroup_tags)
BH$NNgroup_tags<-as.numeric(as.character(BH$NNgroup_tags))

BH<-BH %>%
  mutate(nonnativebin=sum(na.omit(NNgroup_tags))/length(na.omit(NNgroup_tags)))

BHFNmean<-mean(na.omit(BH$forbnumber))
BHFNsd<-sd(na.omit(BH$forbnumber))
breaks<-c(-Inf,unique(c(BHFNmean-(2*BHFNsd),BHFNmean+(2*BHFNsd))),Inf)
tags<-c("0","1","0")
BH$FNgroup_tags<-cut(BH$forbnumber, breaks=breaks,include.lowest=TRUE,
right=FALSE,labels=tags)
summary(BH$FNgroup_tags)
BH$FNgroup_tags<-as.numeric(as.character(BH$FNgroup_tags))
BH<-BH %>%
  mutate(forbnumberbin=sum(na.omit(FNgroup_tags))/length(na.omit(FNgroup_tags)))

BHRmean<-mean(na.omit(BH$rock))
BHRsd<-sd(na.omit(BH$rock))
breaks<-c(-Inf,unique(c(BHRmean-(2*BHRsd),BHRmean+(2*BHRsd))),Inf)
tags<-c("0","1","0")
```

```
BH$Rgroup_tags<-cut(BH$rock, breaks=breaks,include.lowest=TRUE, right=FALSE,labels=tags)
summary(BH$Rgroup_tags)
BH$Rgroup_tags<-as.numeric(as.character(BH$Rgroup_tags))

BH<-BH %>%
  mutate(rockbin=sum(na.omit(Rgroup_tags))/length(na.omit(Rgroup_tags)))

BHLmean<-mean(na.omit(BH$litter))
BHLsd<-sd(na.omit(BH$litter))
breaks<-c(-Inf,unique(c(BHLmean-(2*BHLsd),BHLmean+(2*BHLsd))),Inf)
tags<-c("0","1","0")
BH$Lgroup_tags<-cut(BH$litter, breaks=breaks,include.lowest=TRUE, right=FALSE,labels=tags)
summary(BH$Lgroup_tags)
BH$Lgroup_tags<-as.numeric(as.character(BH$Lgroup_tags))

BH<-BH %>%
  mutate(litterbin=sum(na.omit(Lgroup_tags))/length(na.omit(Lgroup_tags)))


BHSUmean<-mean(na.omit(BH$succulent))
BHSUsd<-sd(na.omit(BH$succulent))
breaks<-c(-Inf,unique(c(BHSUmean-(2*BHSUsd),BHSUmean+(2*BHSUsd))),Inf)
tags<-c("0","1","0")
BH$SUgroup_tags<-cut(BH$succulent, breaks=breaks,include.lowest=TRUE,
right=FALSE,labels=tags)
summary(BH$SUgroup_tags)
BH$SUgroup_tags<-as.numeric(as.character(BH$SUgroup_tags))

BH<-BH %>%
  mutate(succulentbin=sum(na.omit(SUgroup_tags))/length(na.omit(SUgroup_tags)))

BHGAmean<-mean(na.omit(BH$gaps))
BHGAsd<-sd(na.omit(BH$gaps))
breaks<-c(-Inf,unique(c(BHGAmean-(2*BHGAsd),BHGAmean+(2*BHGAsd))),Inf)
tags<-c("0","1","0")
BH$GAgroup_tags<-cut(BH$gaps, breaks=breaks,include.lowest=TRUE, right=FALSE,labels=tags)
summary(BH$GAgroup_tags)
BH$GAgroup_tags<-as.numeric(as.character(BH$GAgroup_tags))

BH<-BH %>%
  mutate(gapsbin=sum(na.omit(GAgroup_tags))/length(na.omit(GAgroup_tags)))

##
```

```
BHL<-BH %>%
  select(plotID,BSgroup_tags,SSgroup_tags,Ggroup_tags,Fgroup_tags,Ngroup_tags,
           NNgroup_tags,Tgroup_tags,Sgroup_tags,GHgroup_tags,FHgroup_tags,FNgroup_tags,
           Rgroup_tags,SUgroup_tags,Lgroup_tags,GAgroup_tags)
BHL<-gather(BHL,indicator,value,-plotID,na.rm=TRUE)
BHL<- BHL %>%
  group_by(plotID) %>%
  mutate(sum=n())
BHGraph<-BH
BHGraph$sum<-BHL$sum[match(BHGraph$plotID,BHL$plotID)]
BHGraph[is.na(BHGraph)] <- 0


BHGraph<-BHGraph %>%
  mutate(graph=((BSgroup_tags+SSgroup_tags+Ggroup_tags+Fgroup_tags+Ngroup_tags+

NNgroup_tags+Tgroup_tags+Sgroup_tags+GHgroup_tags+FHgroup_tags+FNgroup_tags+
                  Rgroup_tags+SUgroup_tags+Lgroup_tags+GAgroup_tags)/sum)*100)
BHGraph$list<-(1:length(BHGraph$graph))

BHB<-
ggplot(BHGraph,aes(x=list,y=graph,color=graph,label=plotID))+geom_point()+geom_text_repel(a
es(label=plotID),hjust=0, vjust=0)
B3<-BHB+ggtitle("Plots within expected range BH")+labs(x="Plot ID",y="Percent of indicators
within expected range")+theme(axis.text.x=element_blank())
B3+scale_colour_gradient(low = "red", high = "green3", na.value = NA)+
theme(legend.position = "none")


#table of benchmarks met for appendix
BH2<-BH
BH2$graph<-as.numeric(BHGraph$graph)
BH2$graph<-round(BH2$graph,digits=2)
BH3<-BH2 %>%

select(plotID,allotment,BSgroup_tags,SSgroup_tags,Fgroup_tags,Ggroup_tags,Sgroup_tags,Tgrou
p_tags,FHgroup_tags,

Ngroup_tags,NNgroup_tags,GAgroup_tags,Lgroup_tags,Rgroup_tags,SUgroup_tags,FNgroup_tags,GHg
roup_tags,graph)
BH3<-BH3%>%
  arrange(graph) %>%
  filter(graph < 100)
```

Select() allows us to select only the columns we're interested in. In this we selected our group_tags which is the column giving meeting (1) or not meeting (0). Gather() takes a long data frame and makes it wide. Group_by() allows us to make calculations based on the group plot ID. Then we can get the sum of indicator benchmarks for each plot. We add this calculation back into the initial data frame and then calculate the sum of indicators meeting benchmark divided by the sum of indicators. This is done in order to remove NAs.

Ggplot() creates base plot, geom_text_repel insures plot names are not overlapping, ggtitle() inserts the title, theme() allows for modification of text and background of the plot and scale_color_gradient() allows us to change the color gradient

This produces the table of plots and whether or not each indicator benchmark was met

```
names(BH3)<-
c("Plot.ID","Allotment","Bare.Soil","Soil.Stability","Forb","Grass","Shrub","Tree","Forb.He
ight","Nonnoxious","Noxious","Gaps","Litter","Rock","Succulent","Forb.Number","Grass.Height
","Percent.Meeting")
formattable(BH3)
write.csv(BH3,"BHIndicatorsmet.csv")
#to view allotments and percent of plot meeting benchmark
BHA1<-BH
BH2$graph<-as.numeric(BHGraph$graph)
BHA1$graph<-round(BH2$graph,digits=2)
BHA<-BHA1 %>%
  select(plotID,allotment,graph)
BHA<-BHA%>%
  arrange(allotment)
names(BHA)<-c("Plot.ID","Allotment.Name","Percent.Meeting")
formattable(BHA)

write.csv(BHA,"BHallotmentspercentmeeting.csv")

#REMOVE DUPLICATES
BH$baresoil.bin<-BH$baresoil.bin*100
BH$soilstabilitybin<-BH$soilstabilitybin*100
BH$treecoverbin<-BH$treecoverbin*100
BH$shrubcoverbin<-BH$shrubcoverbin*100
BH$grasscoverbin<-BH$grasscoverbin*100
BH$grassheightbin<-BH$grassheightbin*100
BH$forb.heightbin<-BH$forb.heightbin*100
BH$forbcoverbin<-BH$forbcoverbin*100
BH$gapsbin<-BH$gapsbin*100
BH$nonnativebin<-BH$nonnativebin*100
BH$nativebin<-BH$nativebin*100
BH$succulentbin<-BH$succulentbin*100
BH$litterbin<-BH$litterbin*100
BH$rockbin<-BH$rockbin*100
BH$forbnumberbin<-BH$forbnumberbin*100
BH$graph<-BHGraph$graph[match(BH$plotID,BHGraph$plotID)]
breaks<-c(0,100,Inf)
tags<-c("0","1")
BH$SUMgroup_tags<-cut(BH$graph, breaks=breaks,include.lowest=TRUE, right=FALSE,labels=tags)
summary(BH$SUMgroup_tags)
BH$SUMgroup_tags<-as.numeric(as.character(BH$SUMgroup_tags))
```

<-converting proportions to percent

Graph is the calculation of percent of indicators meeting benchmark per plot. We are now creating bins for meeting and not meeting based on if 100% of the indicator benchmarks were met by the plot.

Mutate() is used to create a column with the calculation of percent of plots meeting all indicators. The calculation is sum of plots that met 100% of the indicator benchmarks/ total number of plots.

```
BH<-BH %>%
  mutate(SUMbin=(sum(na.omit(SUMgroup_tags))/length(na.omit(SUMgroup_tags))*100))
BHR<-BH
BH<-BH[!duplicated(BH$nativebin),]
stratablanks<-rep("",15)
Strata<-c("BH",stratablanks)
Indicator<- c("Percent Bare Ground", "Soil Aggregate Stability","Litter","Nonnoxious
Species","Percent Grass Cover",
              "Percent Forb Cover","Percent Shrub Cover", "Percent Noxious Species","Number
of Forbs", "Percent Rock Cover",
              "Percent Tree Cover","Percent Succulent Cover","Percent Gap Cover","Forb
Height","Grass Height","Total Percent of Plots Meeting Benchmark")
Percent.Meeting<-
c(round(BH$baresoil.bin,digits=2),round(BH$soilstabilitybin,digits=2),round(BH$litterbin,di
gits=2),round(BH$nativebin,digits=2),round(BH$grasscoverbin,digits=2),round(BH$forbcoverbin
,digits=2),

round(BH$shrubcoverbin,digits=2),round(BH$nonnativebin,digits=2),round(BH$forbnumberbin,dig
its=2),round(BH$rockbin,digits=2),

round(BH$treecoverbin,digits=2),round(BH$succulentbin,digits=2),round(BH$gapsbin,digits=2),
round(BH$forb.heightbin,digits=2),round(BH$grassheightbin,digits=2),round(BH$SUMbin,digits=
2))
BHdataframe<-data.frame(Strata,Indicator,Percent.Meeting)
formattable(BHdataframe)
```

For both methods of benchmark analysis, TerrADat Query Results and Plots were used for calculations. New columns were added to the Query Results in both methods of meeting benchmarks in order to set values of meeting (1) or not meeting (0). Calculations of percent meeting were done by simply taking the sum divided by the total number. NAs were assigned a value of zero and removed from total number of plot observations. This was done in order to completely omit NA values because NAs could have been non-sampled data.