SPP\_Figures

Jesse Wood

July 17, 2018

# using model avg package

library(AICcmodavg)  
library(unmarked)

## Loading required package: reshape

## Loading required package: lattice

## Loading required package: parallel

## Loading required package: Rcpp

setwd("C:/Users/woodj/Documents/GRAD SCHOOL - CLEMSON/Project-Specific/R work/USDA-songbirds/USDA-songbirds")

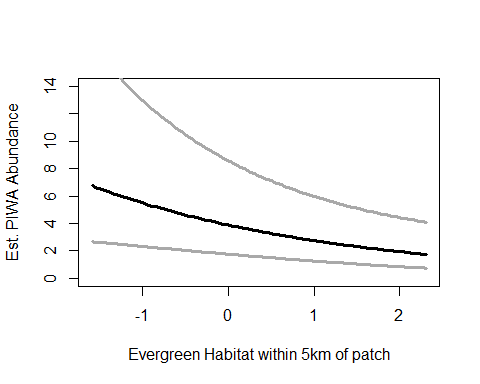
# predict with unmarked # code from ESA 2014

#remember, treat differently for those that are factors vs. those that are #s

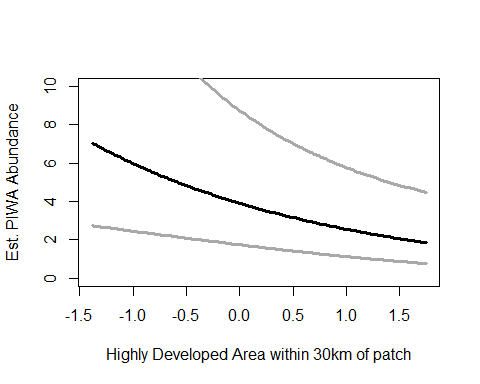
# general structure:

fitList of top models (d<2) for SP write data frame including all important variables - # concat factor (levels) if necessary # set interest variable to max/min # but set every other variable besides one of interest to 0 define any factors with categorical names predict function on the new matrix, type = state plot matrix & 95% CI lines #barplot with error bars code if need be - see bottom

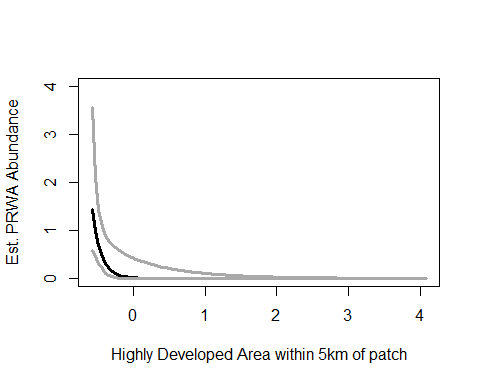
#PIWA  
piwa.abund<- csvToUMF("piwa\_abund.csv", long = FALSE, type = "unmarkedFramePCount")  
obsCovs(piwa.abund)= scale (obsCovs(piwa.abund))  
sc <- siteCovs(piwa.abund)  
sc[,c(5:74)] <- scale(sc[, c(5:74)])  
siteCovs(piwa.abund) <- sc  
  
landscape5.piwa <- pcount(~ Jdate + Time   
 ~ Evergreen5km + HighDev5km  
 , piwa.abund, mixture="P", K=40)  
  
#evergreen5km from landscape 5km model  
NewData.Evergreen5km <- data.frame(Evergreen5km= seq(min(sc$Evergreen5km),max(sc$Evergreen5km), length=100), HighDev5km = 0)   
piwa.est.evergreen5km <- predict(landscape5.piwa, type="state", newdata=NewData.Evergreen5km,appendData=TRUE)  
  
plot(Predicted~ Evergreen5km, data=piwa.est.evergreen5km, ylim=c(0,14), type="l", lwd=3, xlab="Evergreen Habitat within 5km of patch", ylab="Est. PIWA Abundance")  
##95% confidence intervals  
lines(lower~ Evergreen5km, data=piwa.est.evergreen5km, type="l", lwd=3, col="darkgray")  
lines(upper~ Evergreen5km, data=piwa.est.evergreen5km, type="l", lwd=3, col="darkgray")



#PIWA  
piwa.abund<- csvToUMF("piwa\_abund.csv", long = FALSE, type = "unmarkedFramePCount")  
obsCovs(piwa.abund)= scale (obsCovs(piwa.abund))  
sc <- siteCovs(piwa.abund)  
sc[,c(5:74)] <- scale(sc[, c(5:74)])  
siteCovs(piwa.abund) <- sc  
  
landscape30.piwa <- pcount(~ Jdate + Time   
 ~ Evergreen30km + HighDev30km + Protected30km  
 , piwa.abund, mixture="P", K=40)  
  
#High-Dev 30km  
NewData.HighDev30km <- data.frame(Evergreen30km=0,  
 HighDev30km=seq(min(sc$HighDev30km),max(sc$HighDev30km), length=100),  
 Protected30km=0)  
   
piwa.est.highdev30km <- predict(landscape30.piwa, type="state", newdata=NewData.HighDev30km,appendData=TRUE)  
  
plot(Predicted~ HighDev30km, data=piwa.est.highdev30km, ylim=c(0,10), type="l", lwd=3, xlab="Highly Developed Area within 30km of patch", ylab="Est. PIWA Abundance")  
##95% confidence intervals  
lines(lower~ HighDev30km, data=piwa.est.highdev30km, type="l", lwd=3, col="darkgray")  
lines(upper~ HighDev30km, data=piwa.est.highdev30km, type="l", lwd=3, col="darkgray")

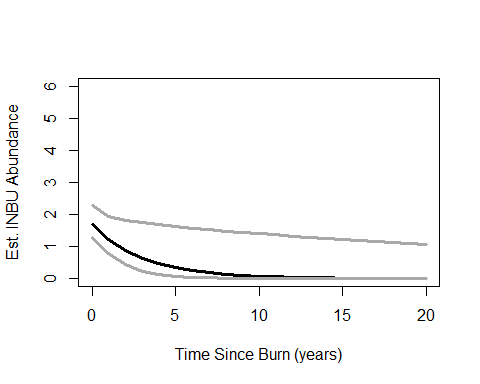


#PRWA - landscape 5km model, high development variable  
prwa.abund<- csvToUMF("prwa\_abund.csv", long = FALSE, type = "unmarkedFramePCount")  
obsCovs(prwa.abund)= scale (obsCovs(prwa.abund))  
sc <- siteCovs(prwa.abund)  
sc[,c(5:74)] <- scale(sc[, c(5:74)])  
siteCovs(prwa.abund) <- sc  
landscape5.prwa <- pcount(~ Jdate  
 ~ Evergreen5km + Grass5km + HighDev5km + Schrubs5km  
 , prwa.abund, mixture="NB", K=60)  
  
NewData.HighDev5km <- data.frame(Evergreen5km=0, Grass5km=0, HighDev5km=seq(min(sc$HighDev5km),max(sc$HighDev5km),length=100), Schrubs5km=0)  
prwa.est.highdev5 <- predict(landscape5.prwa, type="state",  
 newdata=NewData.HighDev5km,appendData=TRUE)  
  
plot(Predicted~ HighDev5km, data=prwa.est.highdev5, ylim=c(0,4), type="l", lwd=3,  
 xlab="Highly Developed Area within 5km of patch", ylab="Est. PRWA Abundance")  
##95% confidence intervals  
lines(lower~ HighDev5km, data=prwa.est.highdev5, type="l", lwd=3, col="darkgray")  
lines(upper~ HighDev5km, data=prwa.est.highdev5, type="l", lwd=3, col="darkgray")

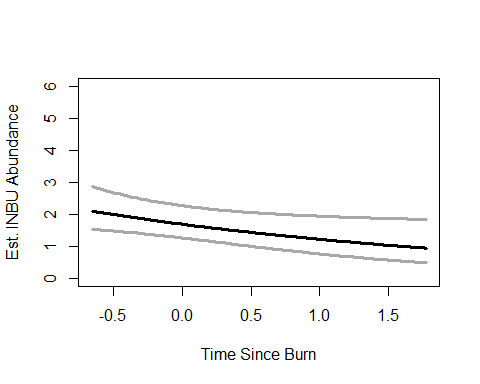


# YBCH - landscape 1km model best, 5km model second best - but no sigs!

#INBU  
inbu.abund<- csvToUMF("inbu\_abund.csv", long = FALSE, type = "unmarkedFramePCount")  
obsCovs(inbu.abund)= scale (obsCovs(inbu.abund))  
sc <- siteCovs(inbu.abund)  
sc[,c(5:74)] <- scale(sc[, c(5:74)])  
siteCovs(inbu.abund) <- sc  
disturbance.inbu <- pcount(~ Jdate  
 ~ TimeSinceB + TimeSinceT  
 , inbu.abund, mixture="P", K=40)  
  
#INBU - disturbance model, Time Since Burn variable  
  
#experimentation#  
newData.inbu<-data.frame(TimeSinceT=0, TimeSinceB=0:20)  
newdata.inbu.est.timeburn<-predict(disturbance.inbu, type="state", newdata=newData.inbu, appendData=TRUE, 2)  
  
plot(Predicted~ TimeSinceB, data=newdata.inbu.est.timeburn, ylim=c(0,6), type="l", lwd=3,  
 xlab="Time Since Burn (years)", ylab="Est. INBU Abundance")  
##95% confidence intervals  
lines(lower~ TimeSinceB, data=newdata.inbu.est.timeburn, type="l", lwd=3, col="darkgray")  
lines(upper~ TimeSinceB, data=newdata.inbu.est.timeburn, type="l", lwd=3, col="darkgray")

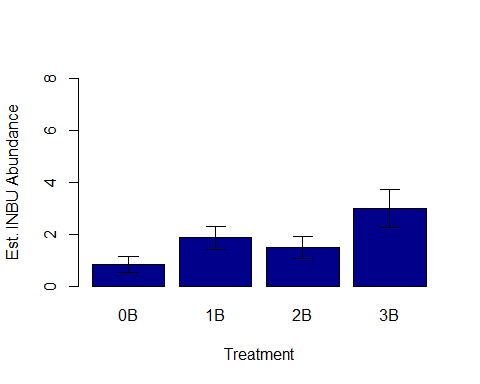


## keep below  
NewData.TsinceB <- data.frame(TimeSinceT= 0, TimeSinceB=seq(min(sc$TimeSinceB),max(sc$TimeSinceB), length=100))  
inbu.est.timeburn <- predict(disturbance.inbu, type="state",  
 newdata=NewData.TsinceB,appendData=TRUE)  
  
plot(Predicted~ TimeSinceB, data=inbu.est.timeburn, ylim=c(0,6), type="l", lwd=3,  
 xlab="Time Since Burn", ylab="Est. INBU Abundance")  
##95% confidence intervals  
lines(lower~ TimeSinceB, data=inbu.est.timeburn, type="l", lwd=3, col="darkgray")  
lines(upper~ TimeSinceB, data=inbu.est.timeburn, type="l", lwd=3, col="darkgray")

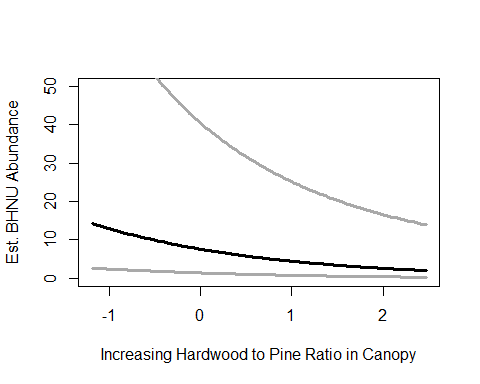


#INBU - nothing significant for life history model

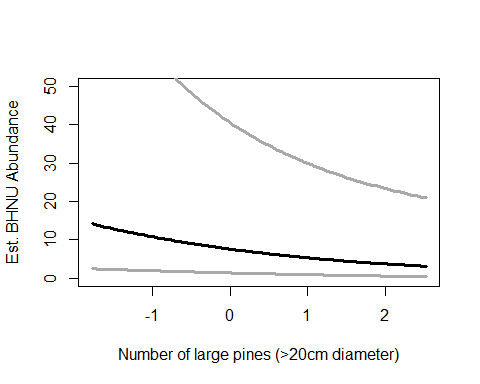
#INBU - treatment model, 1B & 3B significant variable  
  
treatment.inbu <- pcount(~ Jdate  
 ~ Treatment + Nthins  
 , inbu.abund, mixture ="P", K=40)  
  
##  
error.bar <- function(x, y, upper, lower=upper, length=0.1,...){  
 if(length(x) != length(y) | length(y) !=length(lower) | length(lower) != length(upper))  
 stop("vectors must be same length")  
 arrows(x,y+upper, x, y-lower, angle=90, code=3, length=length, ...)  
}  
##  
  
NewData.matrix <- data.frame(Treatment=factor(c("0B","1B","2B","3B")),  
 Nthins=0)  
inbu.est.matrix <- predict(treatment.inbu, type="state",  
 newdata=NewData.matrix,appendData=TRUE)  
  
data.mean<-inbu.est.matrix$Predicted  
data.sd<-inbu.est.matrix$SE  
  
bar.p <-barplot(data.mean,  
 names.arg=c("0B","1B","2B","3B"),  
 ylim = c(0, 8), ylab="Est. INBU Abundance", xlab="Treatment",  
 #cex.names = 1.5, cex.axis=1.5, cex.lab=1.5,   
 col="darkblue")  
  
error.bar(bar.p,data.mean,data.sd) #sd



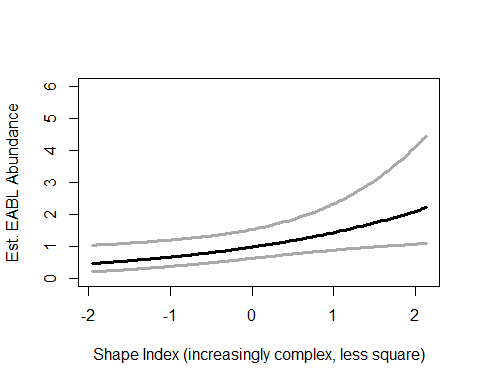
#BHNU  
bhnu.abund<- csvToUMF("bhnu\_abund.csv", long = FALSE, type = "unmarkedFramePCount")  
obsCovs(bhnu.abund)= scale (obsCovs(bhnu.abund))  
sc <- siteCovs(bhnu.abund)  
sc[,c(5:74)] <- scale(sc[, c(5:74)]) #from 26 to 74 +landscape+soils  
siteCovs(bhnu.abund) <- sc  
lh.bhnu <- pcount(~ Time  
 ~ Ccover + Age + Nsnags + TreeHt + NP\_over\_20cm + Rel\_HW2P\_canopy  
 , bhnu.abund, mixture="P", K=40)  
  
#BHNU - life history model, relative hardwood to pine canopy cover variable  
  
NewData.RelHW2Pcanopy <- data.frame(Ccover=0,Age=0,Nsnags=0,TreeHt=0,NP\_over\_20cm=0,  
 Rel\_HW2P\_canopy=seq(min(sc$Rel\_HW2P\_canopy),max(sc$Rel\_HW2P\_canopy), length=100))  
bhnu.est.relhw2p <- predict(lh.bhnu, type="state",  
 newdata=NewData.RelHW2Pcanopy,appendData=TRUE)  
  
plot(Predicted~ Rel\_HW2P\_canopy, data=bhnu.est.relhw2p, ylim=c(0,50), type="l", lwd=3,  
 xlab="Increasing Hardwood to Pine Ratio in Canopy", ylab="Est. BHNU Abundance")  
##95% confidence intervals  
lines(lower~ Rel\_HW2P\_canopy, data=bhnu.est.relhw2p, type="l", lwd=3, col="darkgray")  
lines(upper~ Rel\_HW2P\_canopy, data=bhnu.est.relhw2p, type="l", lwd=3, col="darkgray")



#BHNU - life history model, large pines variable  
  
NewData.NPover20cm <-data.frame(Ccover=0,Age=0,Nsnags=0,TreeHt=0,  
 NP\_over\_20cm=seq(min(sc$NP\_over\_20cm),max(sc$NP\_over\_20cm), length=100),  
 Rel\_HW2P\_canopy=0)  
bhnu.est.NPover20cm <- predict(lh.bhnu, type="state",  
 newdata=NewData.NPover20cm,appendData=TRUE)  
  
plot(Predicted~ NP\_over\_20cm, data=bhnu.est.NPover20cm, ylim=c(0,50), type="l", lwd=3,  
 xlab="Number of large pines (>20cm diameter)", ylab="Est. BHNU Abundance")  
##95% confidence intervals  
lines(lower~ NP\_over\_20cm, data=bhnu.est.NPover20cm, type="l", lwd=3, col="darkgray")  
lines(upper~ NP\_over\_20cm, data=bhnu.est.NPover20cm, type="l", lwd=3, col="darkgray")



# EABL  
eabl.abund<- csvToUMF("eabl\_abund.csv", long = FALSE, type = "unmarkedFramePCount")  
obsCovs(eabl.abund)= scale (obsCovs(eabl.abund))  
sc <- siteCovs(eabl.abund)  
sc[,c(5:74)] <- scale(sc[, c(5:74)])  
siteCovs(eabl.abund) <- sc  
landmetrics.eabl <- pcount (~ Jdate  
 ~ Parea + ShapeIndex  
 , eabl.abund, mixture="P",K=80)  
  
#EABL - land metrics model, shape index variable  
  
NewData.shapeindex <-data.frame(Parea=0,ShapeIndex=seq(min(sc$ShapeIndex),max(sc$ShapeIndex), length=100))  
eabl.est.shapeindex <- predict(landmetrics.eabl, type="state",  
 newdata=NewData.shapeindex,appendData=TRUE)  
  
plot(Predicted~ ShapeIndex, data=eabl.est.shapeindex, ylim=c(0,6), type="l", lwd=3,  
 xlab="Shape Index (increasingly complex, less square)", ylab="Est. EABL Abundance")  
##95% confidence intervals  
lines(lower~ ShapeIndex, data=eabl.est.shapeindex, type="l", lwd=3, col="darkgray")  
lines(upper~ ShapeIndex, data=eabl.est.shapeindex, type="l", lwd=3, col="darkgray")



# NOBO - a ton

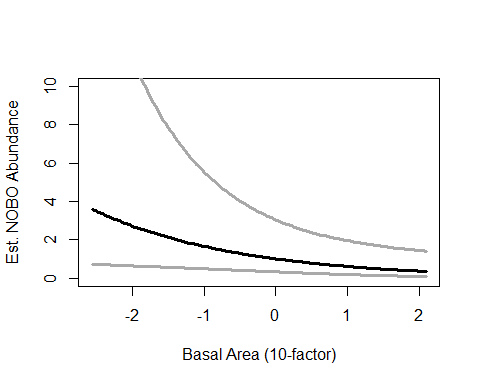
#NOBO - life history model, basal area variable  
  
nobo.abund<- csvToUMF("nobo\_abund.csv", long = FALSE, type = "unmarkedFramePCount")  
obsCovs(nobo.abund)= scale (obsCovs(nobo.abund))  
sc <- siteCovs(nobo.abund)  
sc[,c(5:74)] <- scale(sc[, c(5:74)])  
siteCovs(nobo.abund) <- sc  
lh.nobo <- pcount(~ Jdate + Wind + Sky + Noise +Time  
 ~ BA + Ldepth + HW\_dens\_1050 + FG\_shrub + Age  
 , nobo.abund, mixture="NB", K=80)  
lh.nobo

##   
## Call:  
## pcount(formula = ~Jdate + Wind + Sky + Noise + Time ~ BA + Ldepth +   
## HW\_dens\_1050 + FG\_shrub + Age, data = nobo.abund, K = 80,   
## mixture = "NB")  
##   
## Abundance:  
## Estimate SE z P(>|z|)  
## (Intercept) 0.0173 0.559 0.031 0.9753  
## BA -0.4922 0.216 -2.281 0.0226  
## Ldepth -0.1166 0.252 -0.462 0.6439  
## HW\_dens\_1050 0.3123 0.222 1.407 0.1593  
## FG\_shrub 0.5488 0.212 2.586 0.0097  
## Age 0.0773 0.192 0.404 0.6865  
##   
## Detection:  
## Estimate SE z P(>|z|)  
## (Intercept) -2.2880 0.760 -3.0105 2.61e-03  
## Jdate 1.2972 0.331 3.9196 8.87e-05  
## Wind 0.4707 0.262 1.7995 7.19e-02  
## Sky -0.6892 0.334 -2.0642 3.90e-02  
## Noise -0.0248 0.264 -0.0942 9.25e-01  
## Time -0.3358 0.319 -1.0518 2.93e-01  
##   
## Dispersion:  
## Estimate SE z P(>|z|)  
## 10.5 107 0.0983 0.922  
##   
## AIC: 167.5485

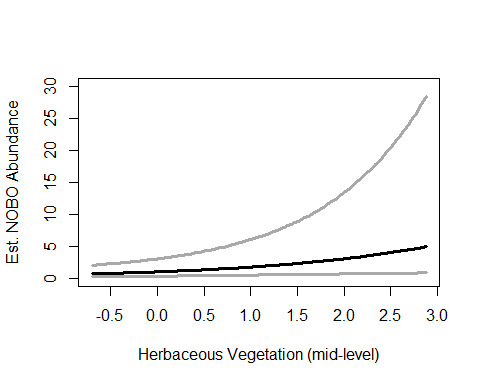
confint(lh.nobo, type="state",method="normal")

## 0.025 0.975  
## lam(Int) -1.0792112 1.11387443  
## lam(BA) -0.9152859 -0.06920571  
## lam(Ldepth) -0.6112370 0.37795119  
## lam(HW\_dens\_1050) -0.1226556 0.74725458  
## lam(FG\_shrub) 0.1329231 0.96466267  
## lam(Age) -0.2982293 0.45289993

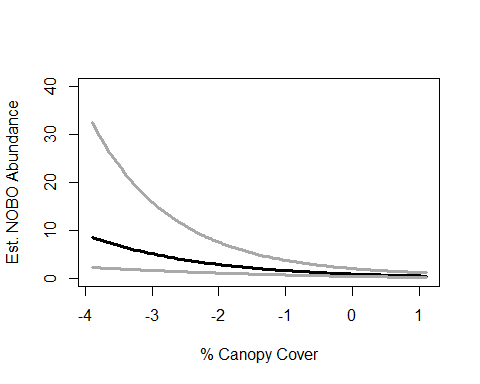
NewData.BA <-data.frame(BA=seq(min(sc$BA),max(sc$BA),length=100),  
 Age=0,FG\_shrub=0,Ldepth=0,HW\_dens\_1050=0)  
nobo.est.ba <- predict(lh.nobo, type="state",  
 newdata=NewData.BA,appendData=TRUE)  
  
plot(Predicted~ BA, data=nobo.est.ba, ylim=c(0,10), type="l", lwd=3,  
 xlab="Basal Area (10-factor)", ylab="Est. NOBO Abundance")  
##95% confidence intervals  
lines(lower~ BA, data=nobo.est.ba, type="l", lwd=3, col="darkgray")  
lines(upper~ BA, data=nobo.est.ba, type="l", lwd=3, col="darkgray")



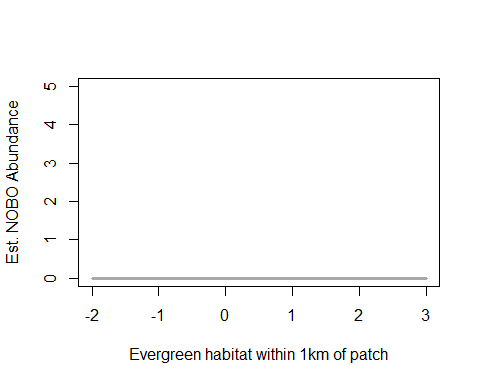
#NOBO - life history model, forbes&grasses\_shrub ht variable  
  
NewData.FGshrub <-data.frame(BA=0,Age=0,FG\_shrub=seq(min(sc$FG\_shrub),max(sc$FG\_shrub),length=100),  
 Ldepth=0,HW\_dens\_1050=0)  
nobo.est.fgshrub <- predict(lh.nobo, type="state",  
 newdata=NewData.FGshrub,appendData=TRUE)  
  
plot(Predicted~ FG\_shrub, data=nobo.est.fgshrub, ylim=c(0,30), type="l", lwd=3,  
 xlab="Herbaceous Vegetation (mid-level)", ylab="Est. NOBO Abundance")  
##95% confidence intervals  
lines(lower~ FG\_shrub, data=nobo.est.fgshrub, type="l", lwd=3, col="darkgray")  
lines(upper~ FG\_shrub, data=nobo.est.fgshrub, type="l", lwd=3, col="darkgray")



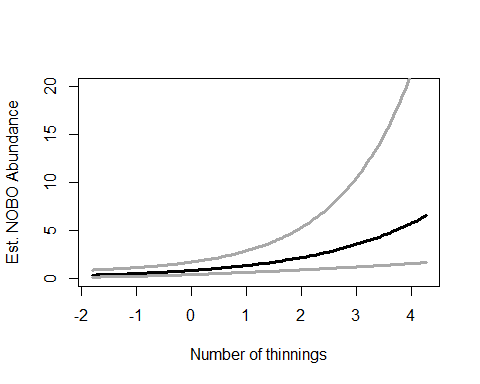
#NOBO - local model, canopy cover variable  
  
local.nobo <- pcount(~ Jdate + Wind + Sky + Noise +Time  
 ~ Ccover + TreeHt + Ldepth  
 , nobo.abund, mixture="NB", K=80)  
  
NewData.ccover <-data.frame(Ccover=seq(min(sc$Ccover),max(sc$Ccover),length=100),TreeHt=0,Ldepth=0)  
nobo.est.ccover <- predict(local.nobo, type="state",  
 newdata=NewData.ccover,appendData=TRUE)  
  
plot(Predicted~ Ccover, data=nobo.est.ccover, ylim=c(0,40), type="l", lwd=3,  
 xlab="% Canopy Cover", ylab="Est. NOBO Abundance")  
##95% confidence intervals  
lines(lower~ Ccover, data=nobo.est.ccover, type="l", lwd=3, col="darkgray")  
lines(upper~ Ccover, data=nobo.est.ccover, type="l", lwd=3, col="darkgray")



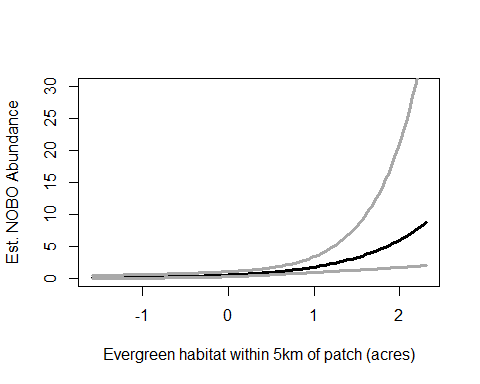
#NOBO - landscape 1km model, evergreen variable  
landscape1.nobo <- pcount(~ Jdate + Wind + Sky + Noise +Time  
 ~ Grass1km + HighDev1km + Schrubs1km + Evergreen1km + Ag1km  
 , nobo.abund, mixture="NB", K=80)  
  
NewData.Evergreen1km <- data.frame(Evergreen1km=seq(-2,3,length=100), Grass1km=0,HighDev1km=0,Schrubs1km=0,Ag1km=0)  
nobo.est.evergreen1 <- predict(landscape1.nobo, type="state",  
 newdata=NewData.Evergreen1km,appendData=TRUE)  
  
plot(Predicted~ Evergreen1km, data=nobo.est.evergreen1, ylim=c(0,5), type="l", lwd=3,  
 xlab="Evergreen habitat within 1km of patch", ylab="Est. NOBO Abundance")  
##95% confidence intervals  
lines(lower~ Evergreen1km, data=nobo.est.evergreen1, type="l", lwd=3, col="darkgray")  
lines(upper~ Evergreen1km, data=nobo.est.evergreen1, type="l", lwd=3, col="darkgray")



#NOBO - treatment model, #Thins variable  
treatment.nobo <- pcount(~ Jdate + Wind + Sky + Noise +Time  
 ~ Nburns + Nthins  
 , nobo.abund, mixture ="NB", K=80) #this one I made Nburns instead of treatment for ones with many many burns  
  
NewData.Nthins <- data.frame(Nburns= 0, Nthins=seq(min(sc$Nthins),max(sc$Nthins), length=100))  
nobo.est.nthins <- predict(treatment.nobo, type="state",  
 newdata=NewData.Nthins,appendData=TRUE)  
  
plot(Predicted~ Nthins, data=nobo.est.nthins, ylim=c(0,20), type="l", lwd=3,  
 xlab="Number of thinnings", ylab="Est. NOBO Abundance")  
##95% confidence intervals  
lines(lower~ Nthins, data=nobo.est.nthins, type="l", lwd=3, col="darkgray")  
lines(upper~ Nthins, data=nobo.est.nthins, type="l", lwd=3, col="darkgray")

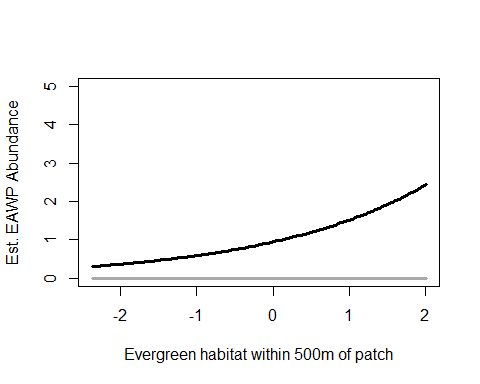


#NOBO - landscape 5km model, evergreen variable  
landscape5.nobo <- pcount(~ Jdate + Wind + Sky + Noise +Time  
 ~ Grass5km + HighDev5km + Schrubs5km + Evergreen5km  
 , nobo.abund, mixture="NB", K=80)  
  
NewData.Evergreen5km <- data.frame(Evergreen5km=seq(min(sc$Evergreen5km),max(sc$Evergreen5km), length=100),Grass5km=0,HighDev5km=0,Schrubs5km=0)  
nobo.est.evergreen5 <- predict(landscape5.nobo, type="state",  
 newdata=NewData.Evergreen5km,appendData=TRUE)  
  
plot(Predicted~ Evergreen5km, data=nobo.est.evergreen5, ylim=c(0,30), type="l", lwd=3,  
 xlab="Evergreen habitat within 5km of patch (acres)", ylab="Est. NOBO Abundance")  
##95% confidence intervals  
lines(lower~ Evergreen5km, data=nobo.est.evergreen5, type="l", lwd=3, col="darkgray")  
lines(upper~ Evergreen5km, data=nobo.est.evergreen5, type="l", lwd=3, col="darkgray")

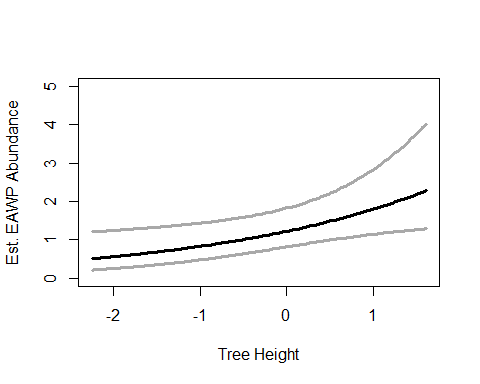


# EAWP - a few

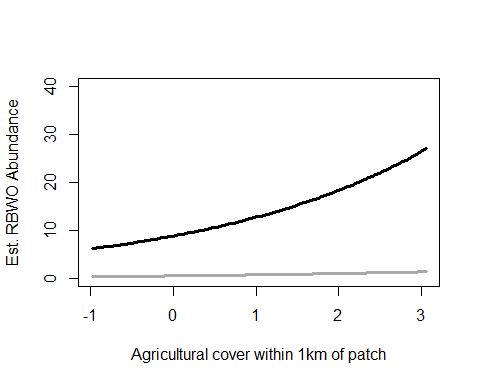
#EAWP - landscape 500m model, evergreen variable  
eawp.abund<- csvToUMF("eawp\_abund.csv", long = FALSE, type = "unmarkedFramePCount")  
obsCovs(eawp.abund)= scale (obsCovs(eawp.abund))  
sc <- siteCovs(eawp.abund)  
sc[,c(5:74)] <- scale(sc[, c(5:74)])  
siteCovs(eawp.abund) <- sc  
landscape500.eawp <- pcount(~ Wind + Sky  
 ~ Evergreen500m + Ag500m + HighDev500m  
 , eawp.abund, mixture="P", K=40)  
  
NewData.Evergreen500m <- data.frame(Evergreen500m=seq(min(sc$Evergreen500m),max(sc$Evergreen500m), length=100),Ag500m=0,HighDev500m=0)  
eawp.est.evergreen500 <- predict(landscape500.eawp, type="state",  
 newdata=NewData.Evergreen500m,appendData=TRUE)  
  
plot(Predicted~ Evergreen500m, data=eawp.est.evergreen500, ylim=c(0,5), type="l", lwd=3,  
 xlab="Evergreen habitat within 500m of patch", ylab="Est. EAWP Abundance")  
##95% confidence intervals  
lines(lower~ Evergreen500m, data=eawp.est.evergreen500, type="l", lwd=3, col="darkgray")  
lines(upper~ Evergreen500m, data=eawp.est.evergreen500, type="l", lwd=3, col="darkgray")



#EAWP - local model, tree height variable  
local.eawp <- pcount(~ Wind + Sky  
 ~ Ccover + TreeHt + Ldepth  
 , eawp.abund, mixture="P", K=40)  
  
NewData.TreeHt <- data.frame(TreeHt=seq(min(sc$TreeHt),max(sc$TreeHt), length=100),  
 Ccover=0,Ldepth=0)  
eawp.est.treeht <- predict(local.eawp, type="state",  
 newdata=NewData.TreeHt,appendData=TRUE)  
  
plot(Predicted~ TreeHt, data=eawp.est.treeht, ylim=c(0,5), type="l", lwd=3,  
 xlab="Tree Height", ylab="Est. EAWP Abundance")  
##95% confidence intervals  
lines(lower~ TreeHt, data=eawp.est.treeht, type="l", lwd=3, col="darkgray")  
lines(upper~ TreeHt, data=eawp.est.treeht, type="l", lwd=3, col="darkgray")



#RBWO - landscape 1km model, ag variable  
rbwo.abund<- csvToUMF("rbwo\_abund.csv", long = FALSE, type = "unmarkedFramePCount")  
obsCovs(rbwo.abund)= scale (obsCovs(rbwo.abund))  
sc <- siteCovs(rbwo.abund)  
sc[,c(5:74)] <- scale(sc[, c(5:74)]) #from 26 to 74 +landscape+soils  
siteCovs(rbwo.abund) <- sc  
landscape1.rbwo <- pcount(~ Jdate + Wind + Sky + Noise + Time  
 ~ Evergreen1km + HighDev1km + Schrubs1km + Ag1km  
 , rbwo.abund, mixture="P", K=120)  
  
NewData.Ag1km <- data.frame(Evergreen1km=0, Ag1km=seq(min(sc$Ag1km),max(sc$Ag1km), length=100),Schrubs1km=0,HighDev1km=0)  
rbwo.est.ag1 <- predict(landscape1.rbwo, type="state",  
 newdata=NewData.Ag1km,appendData=TRUE)  
  
plot(Predicted~ Ag1km, data=rbwo.est.ag1, ylim=c(0,40), type="l", lwd=3,  
 xlab="Agricultural cover within 1km of patch", ylab="Est. RBWO Abundance")  
##95% confidence intervals  
lines(lower~ Ag1km, data=rbwo.est.ag1, type="l", lwd=3, col="darkgray")  
lines(upper~ Ag1km, data=rbwo.est.ag1, type="l", lwd=3, col="darkgray")



# RBWO - upstate model but no variables significant

# CAWR - upstate model but no variables significant

# Barplot with error bars if need be

### A function - just copy and paste between the next 6 lines

error.bar <- function(x, y, upper, lower=upper, length=0.1,…){ if(length(x) != length(y) | length(y) !=length(lower) | length(lower) != length(upper)) stop(“vectors must be same length”) arrows(x,y+upper, x, y-lower, angle=90, code=3, length=length, …) }

#### Copy and paste above. Do not edit

# code below for categorical bar plot data - not yet edited for my data

data.mean<-bhnu.est.matrixSE

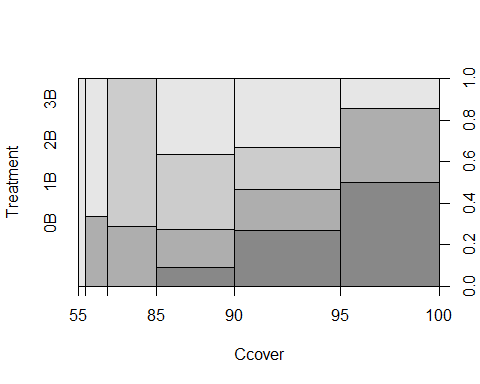
bar.p <-barplot(data.mean, names.arg=c(“Semi-urban”,“Recreation”,“Protected”,“Agriculture”), ylim = c(0, 10), ylab=“Est. BHNU Abundance”, xlab=“System Type”, #cex.names = 1.5, cex.axis=1.5, cex.lab=1.5, col=“darkblue”)

error.bar(bar.p,data.mean,data.sd) #sd #end

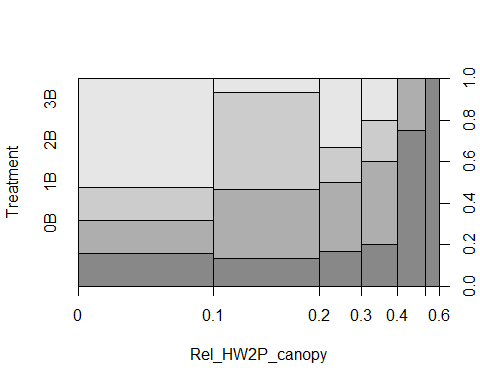
VEG DATA

vdata <-read.csv("template\_pcount\_csv.csv")

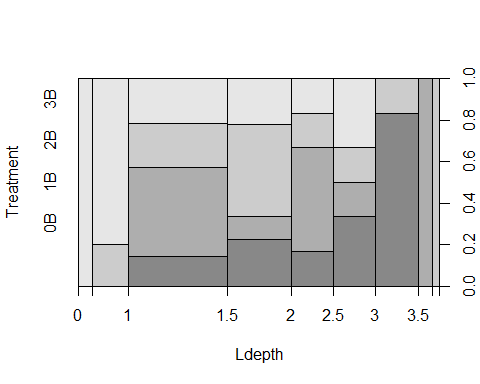
plot(Treatment ~ Ccover, data=vdata)



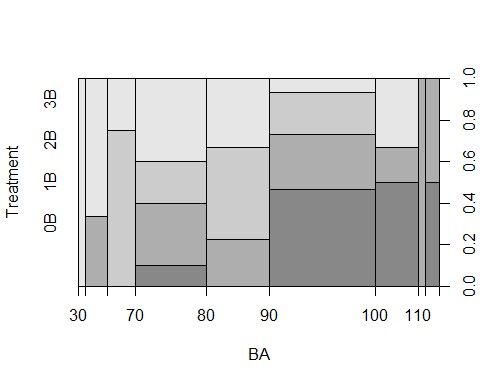
plot(Treatment ~ Rel\_HW2P\_canopy, data=vdata)



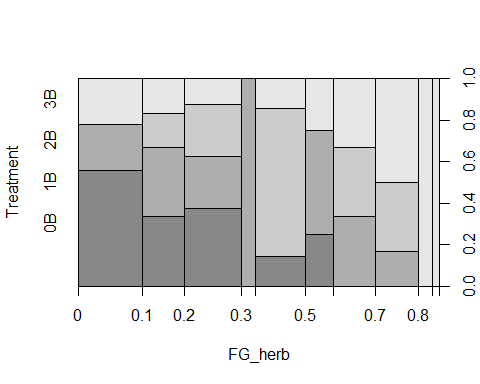
plot(Treatment ~ Ldepth, data=vdata)



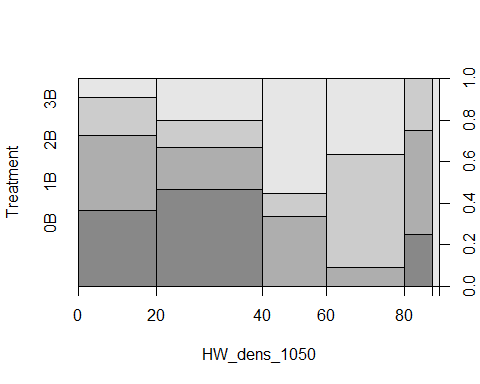
plot(Treatment ~ BA, data=vdata)



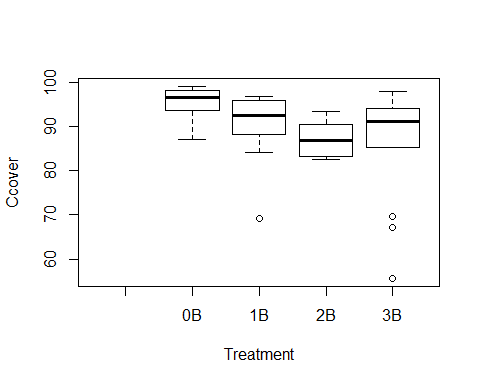
plot(Treatment ~ FG\_herb, data=vdata)



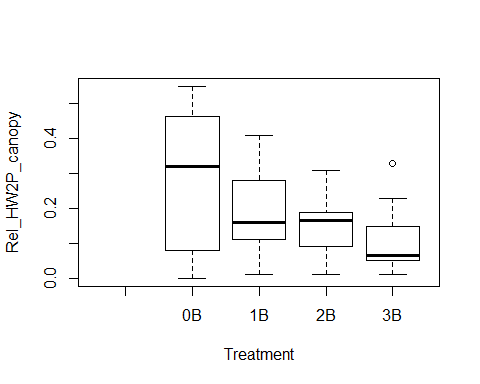
plot(Treatment ~ HW\_dens\_1050, data=vdata)



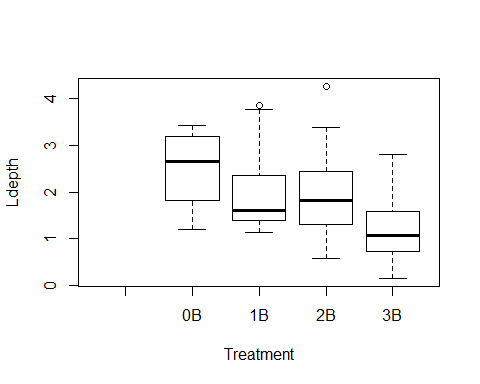
plot(Ccover ~ Treatment, data=vdata)



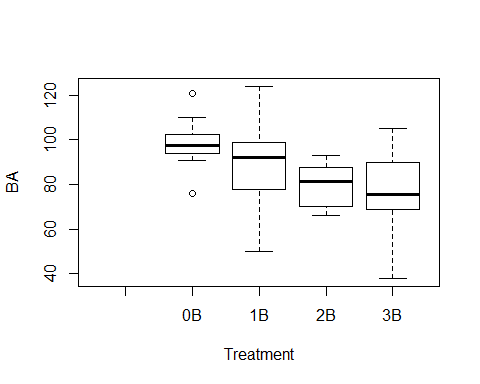
plot(Rel\_HW2P\_canopy ~ Treatment, data=vdata)



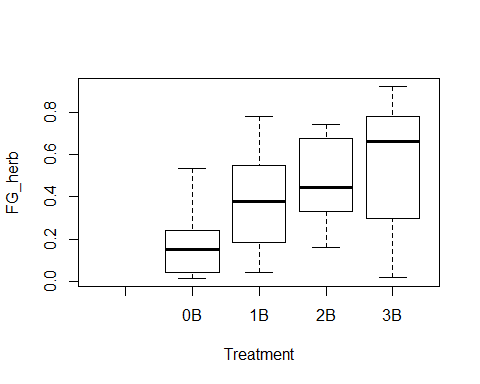
plot(Ldepth ~ Treatment, data=vdata)



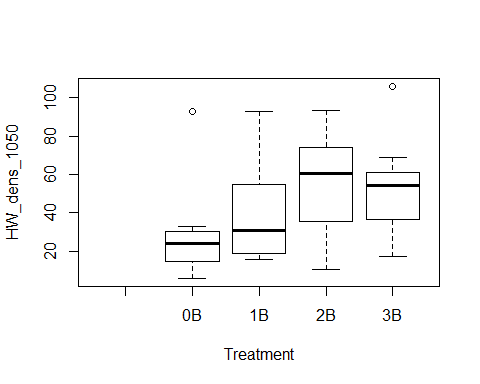
plot(BA ~ Treatment, data=vdata)



plot(FG\_herb ~ Treatment, data=vdata)



plot(HW\_dens\_1050 ~ Treatment, data=vdata)



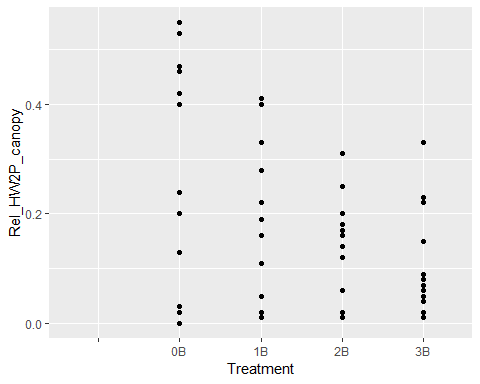
library("ggplot2")  
print(ggplot(vdata, aes(Treatment, Ccover)) + geom\_point())

## Warning: Removed 2 rows containing missing values (geom\_point).



print(ggplot(vdata, aes(Treatment, Rel\_HW2P\_canopy)) + geom\_point())

## Warning: Removed 2 rows containing missing values (geom\_point).



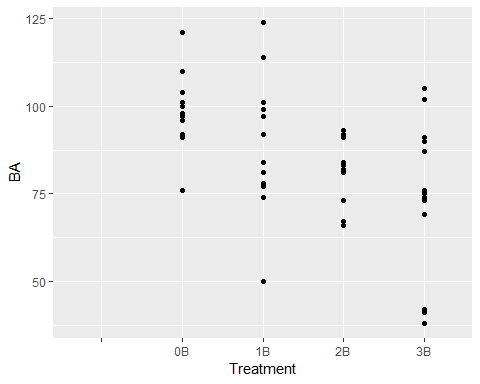
print(ggplot(vdata, aes(Treatment, Ldepth)) + geom\_point())

## Warning: Removed 2 rows containing missing values (geom\_point).



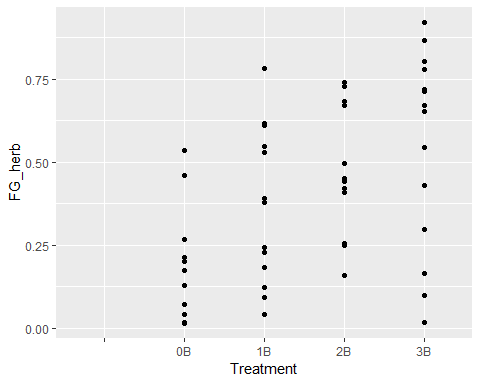
print(ggplot(vdata, aes(Treatment, BA)) + geom\_point())

## Warning: Removed 2 rows containing missing values (geom\_point).



print(ggplot(vdata, aes(Treatment, FG\_herb)) + geom\_point())

## Warning: Removed 2 rows containing missing values (geom\_point).



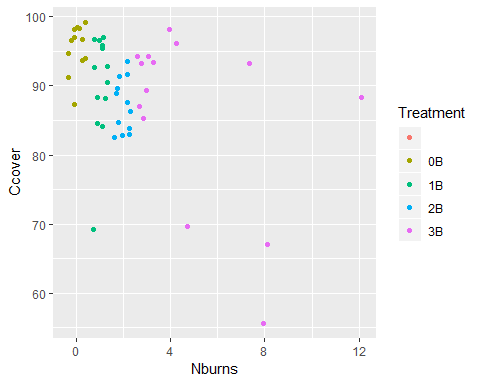
print(ggplot(vdata, aes(Treatment, HW\_dens\_1050)) + geom\_point())

## Warning: Removed 2 rows containing missing values (geom\_point).

 #jittered and color-coded treatment (nburns axis) instead

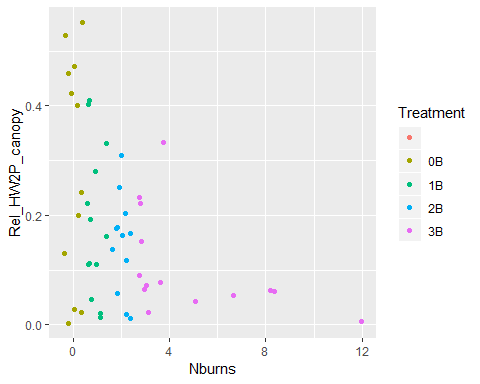
print(ggplot(vdata, aes(Nburns, Ccover, colour=Treatment)) + geom\_jitter())

## Warning: Removed 2 rows containing missing values (geom\_point).



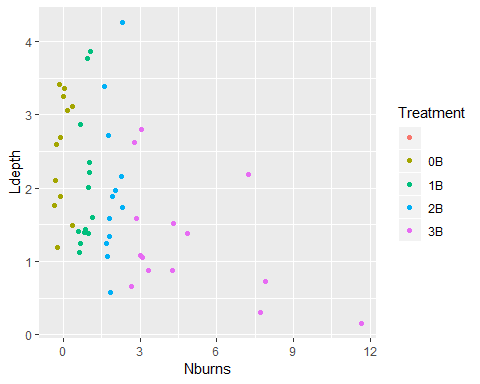
print(ggplot(vdata, aes(Nburns, Rel\_HW2P\_canopy, colour=Treatment)) + geom\_jitter())

## Warning: Removed 2 rows containing missing values (geom\_point).



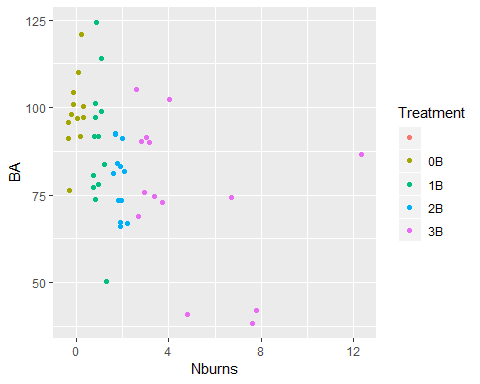
print(ggplot(vdata, aes(Nburns, Ldepth, colour=Treatment)) + geom\_jitter())

## Warning: Removed 2 rows containing missing values (geom\_point).



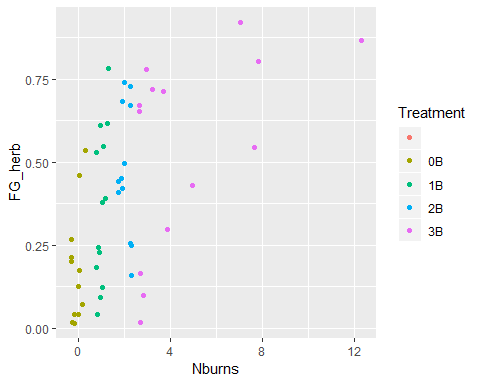
print(ggplot(vdata, aes(Nburns, BA, colour=Treatment)) + geom\_jitter())

## Warning: Removed 2 rows containing missing values (geom\_point).



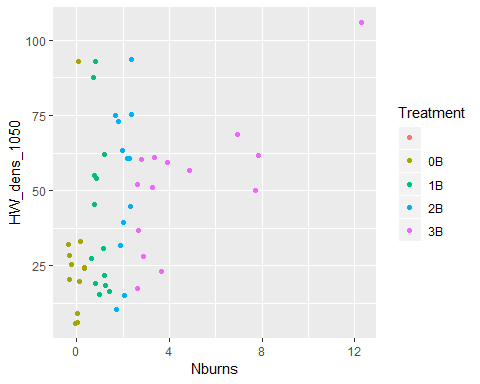
print(ggplot(vdata, aes(Nburns, FG\_herb, colour=Treatment)) + geom\_jitter())

## Warning: Removed 2 rows containing missing values (geom\_point).



print(ggplot(vdata, aes(Nburns, HW\_dens\_1050, colour=Treatment)) + geom\_jitter())

## Warning: Removed 2 rows containing missing values (geom\_point).



fig<-plot(Nthins ~ Age, vdata)  
abline(lm(vdata$Nthins ~ vdata$Age),col="blue")

