Analysis

An analysis using LiDAR data to detect moose browsing effects, with ground truthing

```r
library(readr)
library(ggplot2)
library(glmmTMB)
library(ggplot2)
library(ggpubr)
library(reshape2)
library(corrgram)
library(arm)
library(lme4)
library(MuMIn)
library(car)
library(pryr) # %<a-%
library(corrplot)
library(latticeExtra)
```

Get compiled dataset (see compile.R)

```r
dat <- read_csv("../data/compiledDataset.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   locality_and_treatment = col_character(),
##   LocalityCode = col_character(),
##   LocalityName = col_character(),
##   Treatment = col_character(),
##   resolution_m = col_character(),
##   region = col_character(),
##   trt = col_character()
## )

## See spec(...) for full column specifications.
```

```r
head(dat)
```

```
## # A tibble: 6 x 44
##   locality_and_tr~ LocalityCode LocalityName Treatment Longitude Latitude
##   <chr>            <chr>        <chr>        <chr>         <dbl>    <dbl>
## 1 bratsberg_b      BRB          Bratsberg    B              10.5     63.4
## 2 bratsberg_ub     BRUB         Bratsberg    UB             10.5     63.4
```

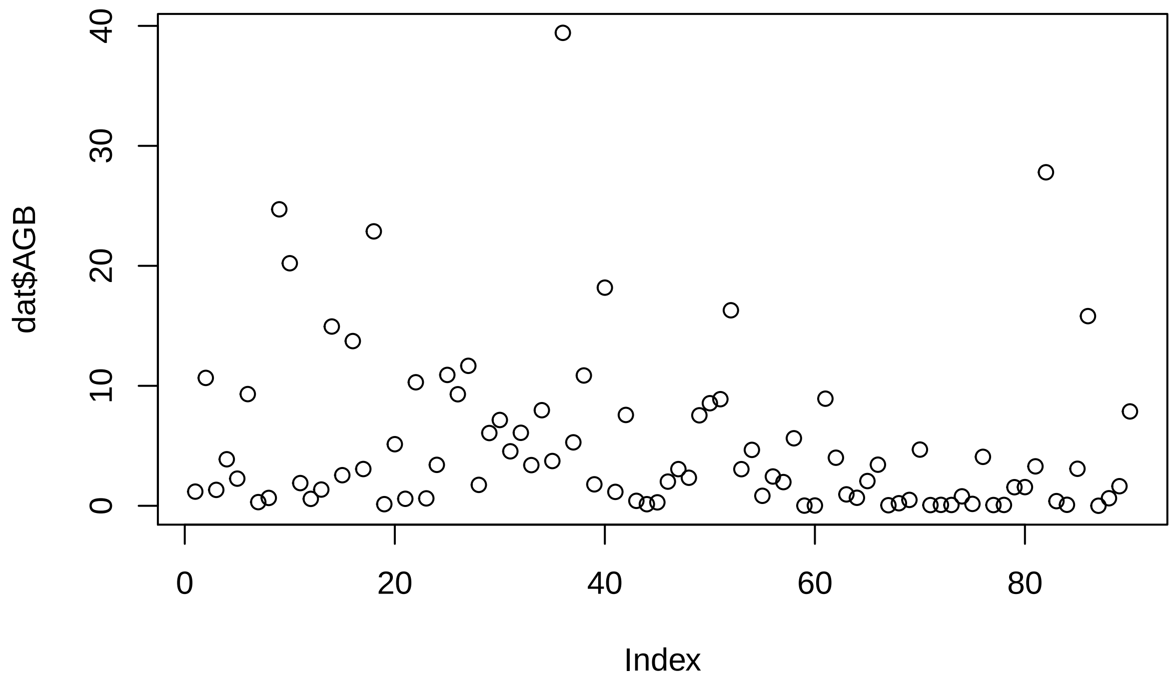
```
## 3 didrik_holmsen_b DHB          Didrik Holm~ B             11.4      59.9
## 4 didrik_holmsen_~ DHUB         Didrik Holm~ UB            11.4      59.9
## 5 drangedal1_b     1DRB         Drangedal1   B              9.15     59.1
## 6 drangedal1_ub    1DRUB        Drangedal1   UB             9.15     59.1
## # ... with 38 more variables: Clear.cut <dbl>, Year.initiated <dbl>,
## #   LiDAR.data.from.year <dbl>, plot_density_m2 <dbl>, resolution_m <chr>,
## #   region <chr>, Moose2015 <dbl>, Reddeer2015 <dbl>, Roedeer2015 <dbl>,
## #   YrsSinceExclosure <dbl>, field_mean <dbl>, field_median <dbl>, trt <chr>,
## #   mean <dbl>, n <dbl>, median <dbl>, sd <dbl>, min <dbl>, max <dbl>,
## #   ninetyfive <dbl>, first_qu <dbl>, third_qu <dbl>, mad <dbl>, rmad <dbl>,
## #   pf70 <dbl>, df1 <dbl>, dl2 <dbl>, rumple <dbl>, vci <dbl>, D6 <dbl>,
## #   D9 <dbl>, Hskew <dbl>, H30 <dbl>, lasR_D6 <dbl>, lasR_D9 <dbl>,
## #   lasR_Hskew <dbl>, lasR_H30 <dbl>, prod <dbl>
```

Housekeeping

```
dat$Treatment <- as.factor(dat$Treatment)
levels(dat$Treatment) <- c('Open plot','Exclosure')
```

Calculating AGB after Økseter et al 2015

```
dat$AGB <-
   exp(log(38.55)+0.75*log(dat$lasR_D6)+0.63*log(dat$lasR_D9)+1.68*asinh(dat$lasR_Hskew)+0.78*asinh(dat$

plot(dat$AGB)
```




```r
tapply(dat$AGB, dat$Treatment, FUN = mean, na.rm=T)
```

```
## Open plot Exclosure
##  3.052881  7.507299
```

We might need these:

```r
dat$canopygrowth <- dat$ninetyfive/dat$YrsSinceExclosure # in m
dat$MgAGBperYearAndHA <- dat$AGB/dat$YrsSinceExclosure
```

```r
dat$vci[is.na(dat$vci)] <- 0
```

This is the dataset after removing the two most productive sites (explained below)

```r
dat2 <- dat[dat$prod <0.7,]
```

A quick data check

```r
table(dat$Treatment, dat$Clear.cut)
```

```
##
##             2000 2002 2003 2004 2005 2006 2007 2008 2009
##   Open plot    1    4    3    7    8    4   10    7    1
##   Exclosure    1    4    3    7    8    4   10    7    1
```

```r
table(dat$Year.initiated, dat$LiDAR.data.from.year)
```

```
##
##        2010 2011 2013 2015 2016 2017 2018 2019
##   2007    0    0    0    0    0    0    0    6
##   2008    4    4    0   18    2    2    0    0
##   2009    0    0    0    0    4   24    0    0
##   2010    0    0    0    0    4    4    4    0
##   2011    0    0    2    0    4    0    0    8
```

```r
table(dat$plot_density_m2, dat$resolution_m)
```

```
##
##     0,25 0,5
##   2    2  54
##   5   32   2
```

Something odd there…

```r
table(dat$region, dat$Treatment)
```

```
##
##             Open plot Exclosure
##   Hedmark          16        16
##   Telemark         14        14
##   Trondelag        15        15
```

```r
table(dat$LocalityName, dat$Treatment)
```

```
##
##                         Open plot Exclosure
##   Bratsberg                     1         1
##   Didrik Holmsen                1         1
##   Drangedal1                    1         1
##   Drangedal3                    1         1
##   Drangedal4                    1         1
##   Eidskog                       1         1
##   Fet 3                         1         1
##   Fritsoe1                      1         1
##   Fritsoe2                      1         1
##   Furesdal                      1         1
##   Halvard Pramhus               1         1
```

```
## 	Hi_tydal                      1          1
## 	Kongsvinger 1                 1          1
## 	Kongsvinger 2                 1          1
## 	Kviteseid1                    1          1
## 	Kviteseid2                    1          1
## 	Kviteseid3                    1          1
## 	Maarud 1                      1          1
## 	Maarud 2                      1          1
## 	Maarud 3                      1          1
## 	Malvik                        1          1
## 	namdalseid_1kub               1          1
## 	Nes 1                         1          1
## 	Nes 2                         1          1
## 	Nome_Cappelen1                1          1
## 	Nome_Cappelen2                1          1
## 	Notodden3                     1          1
## 	Notodden5                     1          1
## 	Notodden6                     1          1
## 	Nsb_Verdal                    1          1
## 	Selbu_Flub                    1          1
## 	Selbu_kl                      1          1
## 	Selbu_Sl                      1          1
## 	Singsaas                      1          1
## 	Sl_Tydal                      1          1
## 	Soerum                        1          1
## 	Stangeskovene Aurskog         1          1
## 	Stangeskovene Eidskog         1          1
## 	steinkjer_1BBb                1          1
## 	steinkjer_2BBb                1          1
## 	Stig Dahlen                   1          1
## 	Sub_Namdalseid                1          1
## 	Truls Holm                    1          1
## 	verdal_1vb                    1          1
## 	verdal_2VB                    1          1
```

Looks good.

Investigating VCI

```
summary(dat$vci)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.1939  0.3830  0.3633  0.4917  0.7946
```

VCI function gives NA when the max point height is <1m (2* bin width set to 0.5). So these are super short statured plots. Its fine to set these to zero, as we have done above.

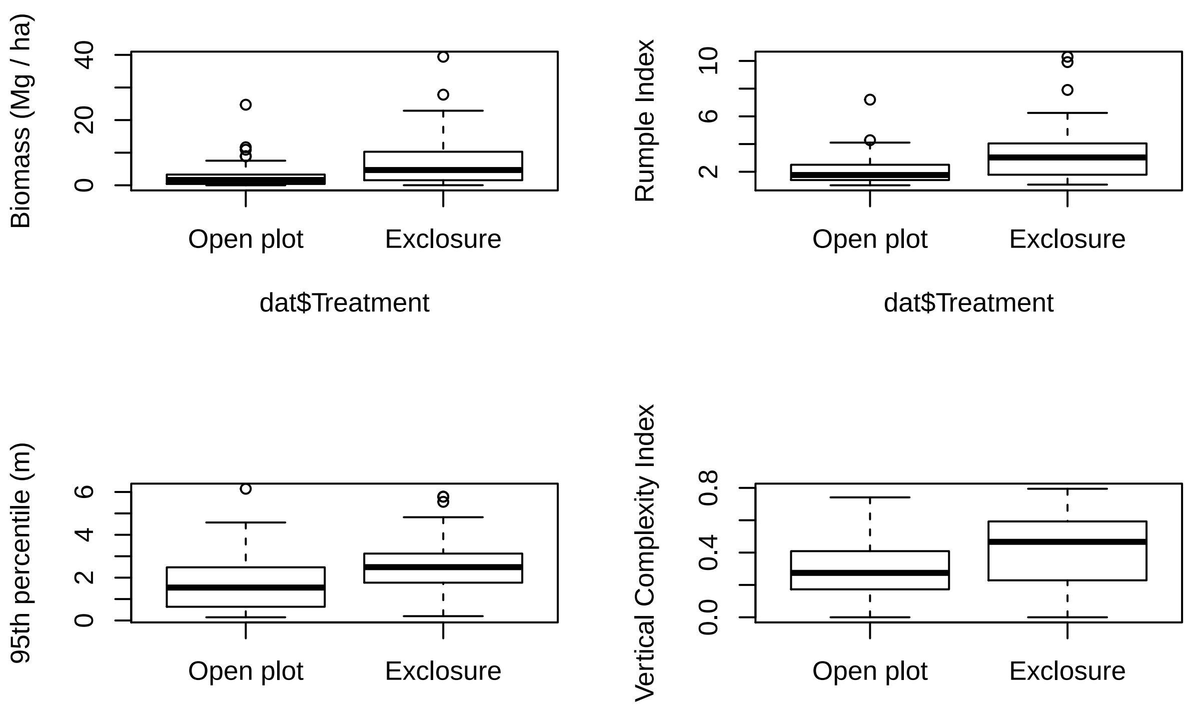
Site table

```r
sites <- dat[dat$Treatment=="Open plot",]
#write_csv(sites, "../output/sites.csv")
```

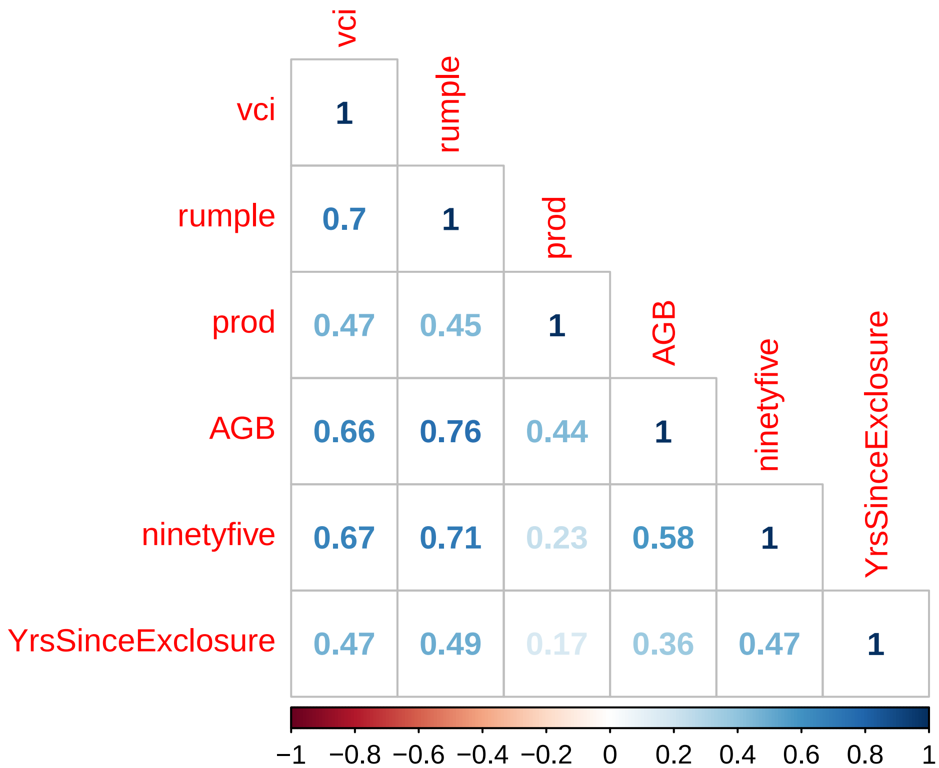
Investiaging other parameters

We have four parameters of interest.

```r
par(mfrow=c(2,2))
boxplot(dat$AGB~dat$Treatment, ylab="Biomass (Mg / ha)")
boxplot(dat$rumple~dat$Treatment, ylab="Rumple Index")
boxplot(dat$ninetyfive~dat$Treatment, ylab="95th percentile (m)")
boxplot(dat$vci~dat$Treatment, ylab="Vertical Complexity Index")
```

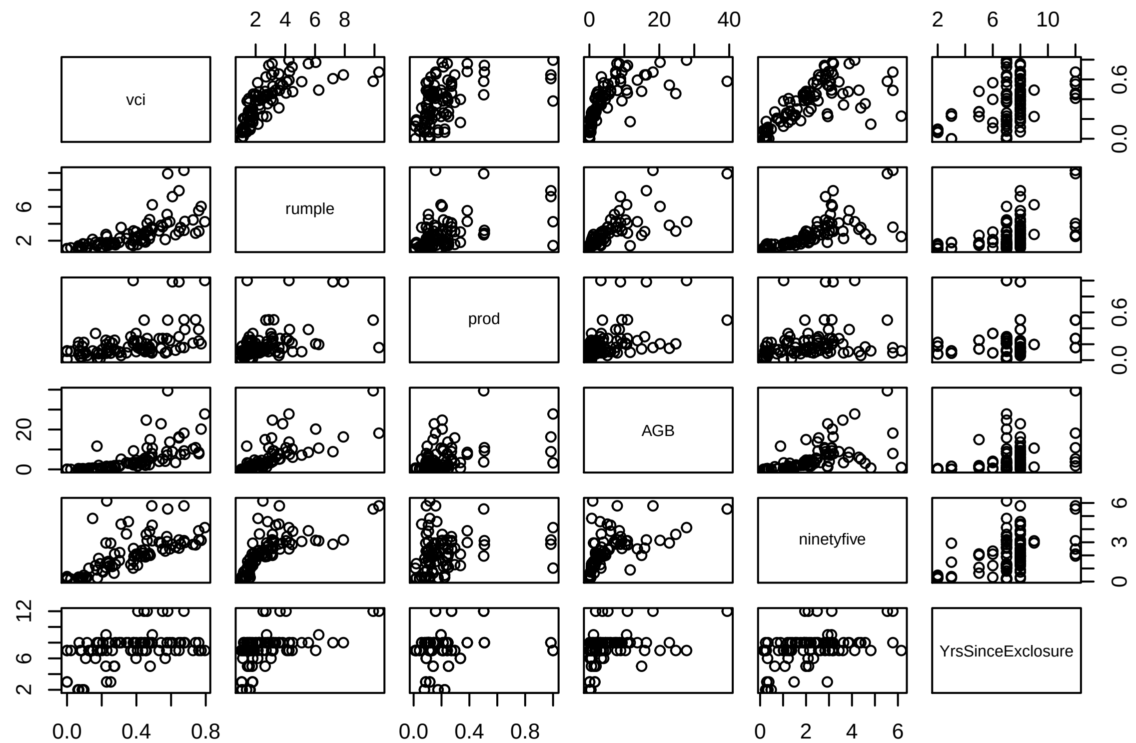



```r
M<-cor(dat[,c("vci", "rumple", "prod",
                 "AGB", "ninetyfive",
                 "YrsSinceExclosure")])

corrplot::corrplot(M, method="number", type="lower")
```



We have to measures of strucural complexity, rumple and VCI. They are correlated, but rumple is strongly correlated to biomass, so we will use VCI.

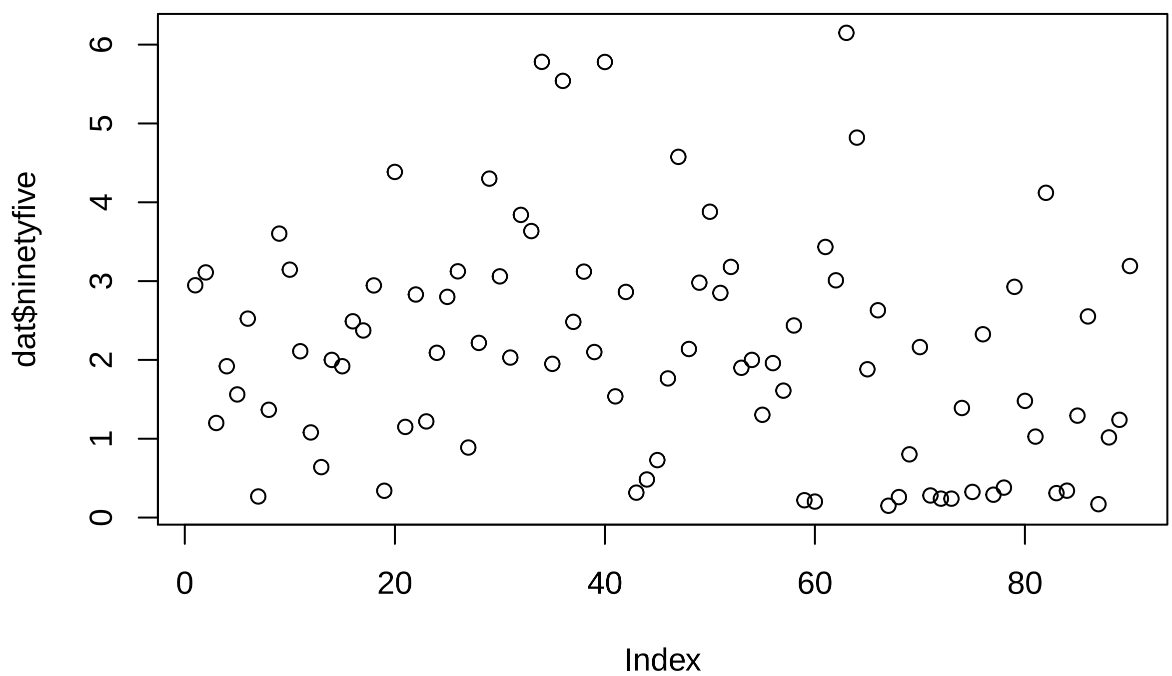
```
pairs(dat[,c("vci", "rumple", "prod",
                 "AGB", "ninetyfive",
                 "YrsSinceExclosure")])
```



Canopy height

We use the 95th percentil instead of Hmax because it's less sensitive to outliers and variation in point density.

```
plot(dat$ninetyfive)
```



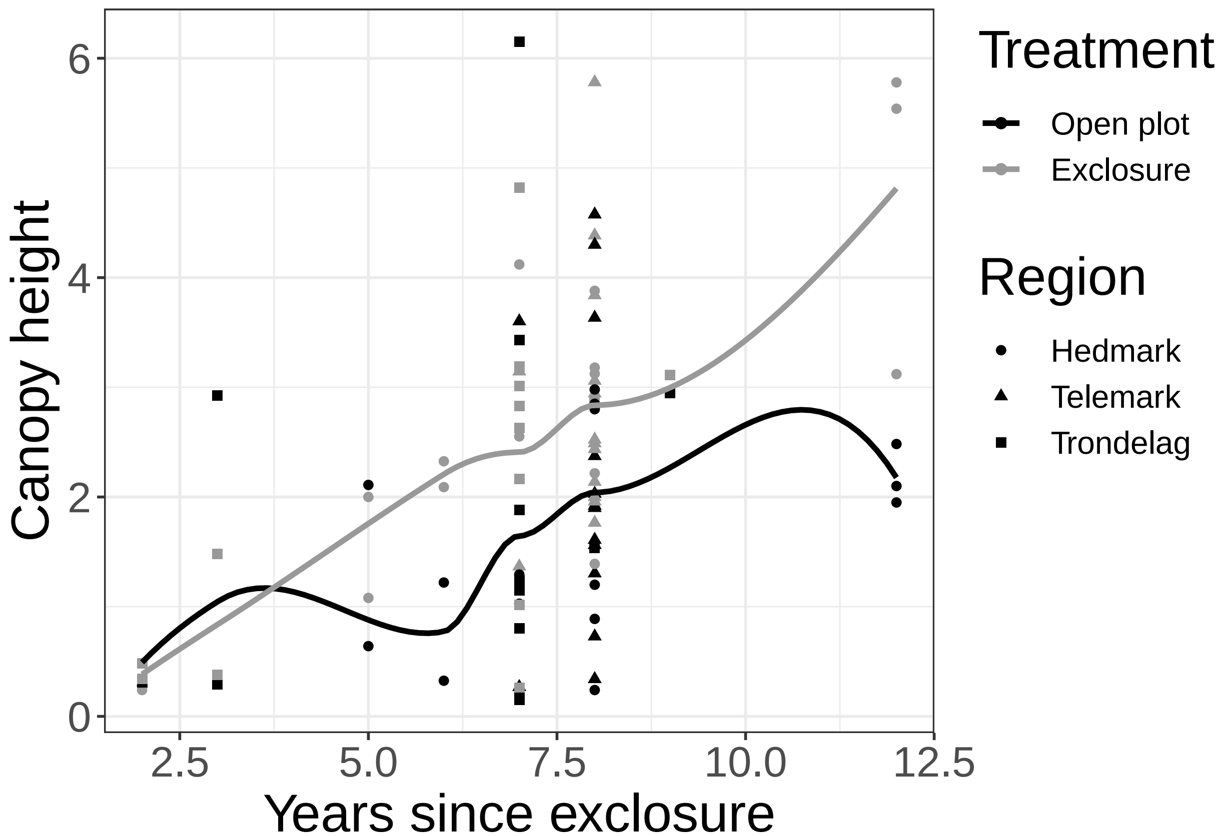
Canopy growth per year

```
ggplot(data = dat, aes(x = YrsSinceExclosure, y = ninetyfive))+
  geom_point(aes(colour= Treatment, shape=region))+
   geom_smooth(aes(colour= Treatment),
               method = "loess", se=F)+
  labs(y='Canopy height', x='Years since exclosure')+
  theme_bw()+
  scale_color_manual(values = c("gray0", "gray60"))+
  labs(colour="Treatment", shape="Region")+
  theme(text = element_text(size = 20))+
  theme(legend.position = 'right',
                             legend.justification = c("left", "top"),
                             legend.box.just = "left",
                             #legend.margin = margin(5, 5, 5, 5),
                             legend.text = element_text(size=12))
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 7
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1
```

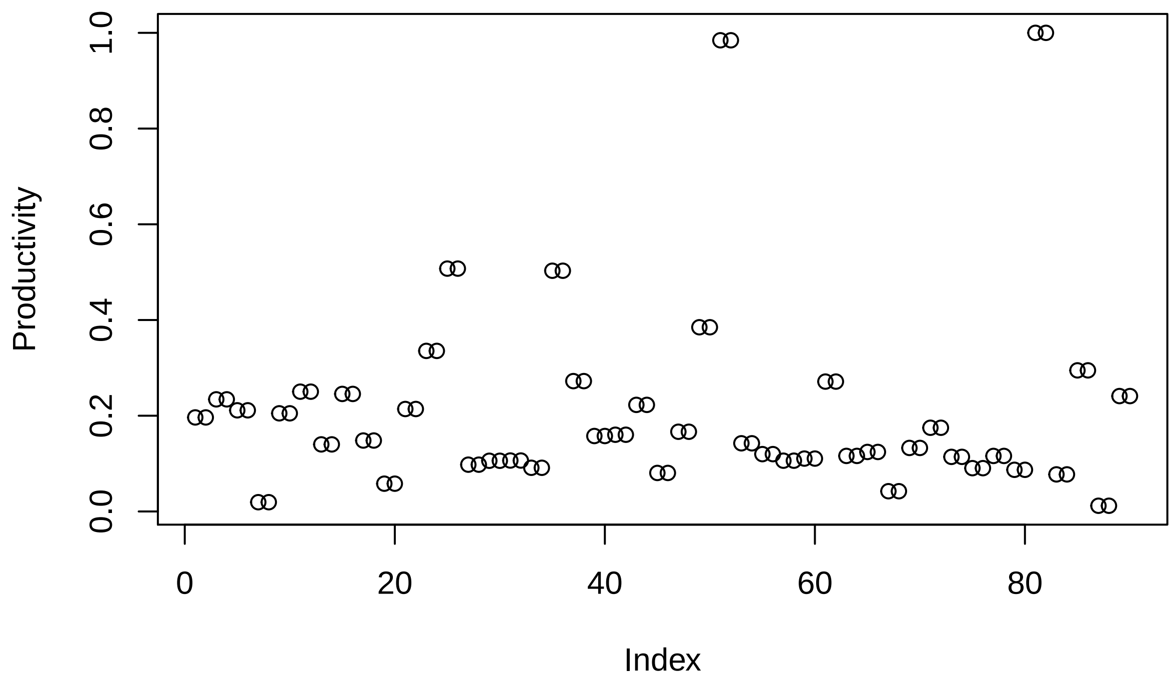
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 7

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1
```



Loess gave some warnings here, but still, this plot shows that it's quite linear.

One of the first things to decide on is what to do with productivity, as there are two outliers:

```
plot(dat$prod, ylab="Productivity")
```




```
dat[dat$prod>0.6,c("LocalityName", "region", "prod")]
```

```
## # A tibble: 4 x 3
##   LocalityName region   prod
##   <chr>        <chr>   <dbl>
## 1 Nes 2        Hedmark 0.984
## 2 Nes 2        Hedmark 0.984
## 3 Stig Dahlen  Hedmark 1
## 4 Stig Dahlen  Hedmark 1
```

These are two sites in Hedmark that probably really are very productive, although there could have been serious sampling error due to chance. But they are legitimate, and we should be careful not to drop them too willingly.

After careful sensitivity analysis, I decided to drop these two locations in dat2. Scripts are moved to the end of the file

modelling

```
dat2$prod2 <- dat2$prod^2
mod_sens1 <- glmmTMB(ninetyfive ~
             Treatment * prod + I(prod^2)
             + YrsSinceExclosure+
```

```r
               YrsSinceExclosure:Treatment + (1|region) + (1|LocalityName),
  data = dat2,  REML=F, family = gaussian)

mod_sens2 <- glmmTMB(ninetyfive ~ Treatment * prod + prod2
        + YrsSinceExclosure+
          YrsSinceExclosure:Treatment +  (1|LocalityName),
  data = dat2, REML=F, family = gaussian)
AIC(mod_sens1, mod_sens2)
```

```
##           df      AIC
## mod_sens1 10 275.5619
## mod_sens2  9 273.5619
```

Regions explains nothing, so I'm removing it

```r
summary(mod_sens2)
```

```
##  Family: gaussian  ( identity )
## Formula:
## ninetyfive ~ Treatment * prod + prod2 + YrsSinceExclosure + YrsSinceExclosure:Treatment +
##     (1 | LocalityName)
## Data: dat2
##
##      AIC      BIC   logLik deviance df.resid
##    273.6    295.7   -127.8    255.6       77
##
## Random effects:
##
## Conditional model:
##  Groups       Name        Variance Std.Dev.
##  LocalityName (Intercept) 0.6460   0.8037
##  Residual                 0.6671   0.8167
## Number of obs: 86, groups:  LocalityName, 43
##
## Dispersion estimate for gaussian family (sigma^2): 0.667
##
## Conditional model:
##                                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)                          -0.20701    0.74528  -0.278  0.78120
## TreatmentExclosure                   -0.95134    0.60788  -1.565  0.11758
## prod                                  6.31499    4.58497   1.377  0.16841
## prod2                                -8.17867    8.81396  -0.928  0.35345
## YrsSinceExclosure                     0.16505    0.08175   2.019  0.04350 *
## TreatmentExclosure:prod              -0.37178    1.66675  -0.223  0.82349
## TreatmentExclosure:YrsSinceExclosure  0.24555    0.08188   2.999  0.00271 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

glmmTMB is not supported by arm::standardise,

```r
dat2$prod_s <- scale(dat2$prod)
dat2$prod2_s <- scale(dat2$prod2)
dat2$YrsSinceExclosure_s <- scale(dat2$YrsSinceExclosure)
dat2$Treatment_c <- ifelse(dat2$Treatment == "Open plot", -0.5, 0.5)
dat2$ninetyfive_s <- scale(dat2$ninetyfive)[,1]
summary(dat2$ninetyfive_s)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -1.34669 -0.73508 -0.06492  0.00000  0.59086  2.81005
```

```r
sd(dat2$ninetyfive_s)
```

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

```r
mod_sens2_s <- glmmTMB(ninetyfive_s ~ Treatment_c * prod_s + prod2_s
                     + YrsSinceExclosure_s+ YrsSinceExclosure_s:Treatment+
                    (1|LocalityName), REML=F,
                       data = dat2)


summary(mod_sens2_s)
```

```
##  Family: gaussian  ( identity )
## Formula:
## ninetyfive_s ~ Treatment_c * prod_s + prod2_s + YrsSinceExclosure_s +
##     YrsSinceExclosure_s:Treatment + (1 | LocalityName)
## Data: dat2
##
##      AIC      BIC   logLik deviance df.resid
##    210.4    232.5    -96.2    192.4       77
##
## Random effects:
##
## Conditional model:
##  Groups       Name        Variance Std.Dev.
##  LocalityName (Intercept) 0.3101   0.5569
##  Residual                 0.3202   0.5659
## Number of obs: 86, groups:  LocalityName, 43
##
## Dispersion estimate for gaussian family (sigma^2): 0.32
##
## Conditional model:
##                                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)                           1.588e-07  1.046e-01   0.000  1.00000
## Treatment_c                           5.107e-01  1.220e-01   4.184 2.86e-05
## prod_s                                4.672e-01  3.437e-01   1.359  0.17401
## prod2_s                              -3.214e-01  3.463e-01  -0.928  0.35345
## YrsSinceExclosure_s                   2.561e-01  1.269e-01   2.019  0.04350
## Treatment_c:prod_s                   -2.834e-02  1.270e-01  -0.223  0.82349
## YrsSinceExclosure_s:TreatmentExclosure  3.810e-01  1.270e-01   2.999  0.00271
##
```

```
## (Intercept)
## Treatment_c                             ***
## prod_s
## prod2_s
## YrsSinceExclosure_s                     *
## Treatment_c:prod_s
## YrsSinceExclosure_s:TreatmentExclosure **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Scaling predictors make it easier to compare slopes, but unstandardised models are much easier to work with for making predictions ect.

```
#stdz_model <- standardize(mod_sens2, standardize.y = FALSE, unchanged="Treatment")
#summary(stdz_model)
```

Then we find all possible model configurations

```
options(na.action = "na.fail")
#cg_cand <- dredge(mod_sens2_s, beta="none", rank = "AICc")
#write_rds(cg_cand, "../data/cg_cand.RData")
cg_cand <- read_rds("../data/cg_cand.RData")
```

Lets compare this to the un.standardised model

```
#uns <- dredge(mod_sens2, beta="none", rank = "AICc")
#write_rds(uns, "../data/uns.RData")
uns <- read_rds("../data/uns.RData")
```

create confidence set with model less than 2 AICc units

```
(cg_cand2 <- subset(cg_cand, delta <2))
```

```
## Global model call: glmmTMB(formula = ninetyfive_s ~ Treatment_c * prod_s + prod2_s +
##     YrsSinceExclosure_s + YrsSinceExclosure_s:Treatment + (1 |
##     LocalityName), data = dat2, REML = F, ziformula = ~0, dispformula = ~1)
## ---
## Model selection table
##    cnd((Int)) dsp((Int)) cnd(pr2_s) cnd(prd_s) cnd(Trt_c) cnd(YSE_s)
## 45 -2.079e-06          +                           0.5107     0.2889
## 47 -6.082e-08          +                0.1647     0.5107     0.2465
## 46  1.387e-06          +     0.1252                0.5107     0.2534
## 48  7.143e-07          +    -0.3214     0.4672     0.5107     0.2598
##    cnd(Trt:YSE_s) df  logLik  AICc delta weight
## 45              +  6 -97.769 208.6  0.00  0.346
## 47              +  7 -96.675 208.8  0.18  0.315
## 46              +  7 -97.153 209.7  1.14  0.196
## 48              +  8 -96.249 210.4  1.77  0.143
## Models ranked by AICc(x)
## Random terms (all models):
## 'cond(1 | LocalityName)'
```

We exclude models with quadratic terms when the main effect is not in the model, if there are such.

```
cg_cand2 <- cg_cand2[-3,]
```

Lets compare to the unstandardised model

```
(uns2 <- subset(uns, delta <2))
```

```
## Global model call: glmmTMB(formula = ninetyfive ~ Treatment * prod + prod2 + YrsSinceExclosure +
##     YrsSinceExclosure:Treatment + (1 | LocalityName), data = dat2,
##     family = gaussian, REML = F, ziformula = ~0, dispformula = ~1)
## ---
## Model selection table
##    cnd((Int)) dsp((Int)) cnd(prd) cnd(pr2) cnd(Trt) cnd(YSE) cnd(Trt:YSE) df
## 45     0.3958          +                          +   0.1862            +  6
## 46     0.2149          +    2.161                 +   0.1588            +  7
## 47     0.4247          +             3.186        +   0.1633            +  7
## 48    -0.1914          +    6.129   -8.179        +   0.1674            +  8
##      logLik  AICc delta weight
## 45 -129.326 271.7  0.00  0.346
## 46 -128.232 271.9  0.18  0.315
## 47 -128.711 272.9  1.14  0.196
## 48 -127.806 273.5  1.77  0.143
## Models ranked by AICc(x)
## Random terms (all models):
## 'cond(1 | LocalityName)'
```

removing model 3

```
(uns2 <- uns2[-3,])
```

```
## Global model call: glmmTMB(formula = ninetyfive ~ Treatment * prod + prod2 + YrsSinceExclosure +
##     YrsSinceExclosure:Treatment + (1 | LocalityName), data = dat2,
##     family = gaussian, REML = F, ziformula = ~0, dispformula = ~1)
## ---
## Model selection table
##    cnd((Int)) dsp((Int)) cnd(prd) cnd(pr2) cnd(Trt) cnd(YSE) cnd(Trt:YSE) df
## 45     0.3958          +                          +   0.1862            +  6
## 46     0.2149          +    2.161                 +   0.1588            +  7
## 48    -0.1914          +    6.129   -8.179        +   0.1674            +  8
##      logLik  AICc delta weight
## 45 -129.326 271.7  0.00  0.430
## 46 -128.232 271.9  0.18  0.392
## 48 -127.806 273.5  1.77  0.178
## Models ranked by AICc(x)
## Random terms (all models):
## 'cond(1 | LocalityName)'
```

The model weights are exactly the same.

Lets export this as a table for the supplementary

```r
temp <- as.data.frame(uns2)
names(temp) <- c("Intercept",
                      "Productivity (P)",
                      "Productivity squared",
                 "Herbivore Exclusion (HE)",
                      "Experimental duration (ED)",
                      "HE x P",
                      "HE x ED",
                      "df",
                      "log likelihood",
                      "AICc",
                      "delta AICc",
                      "weight"
                      )
temp[is.na(temp)] <- 0
#write.csv(temp, "../output/95thModelSet_unstandardized.csv", row.names = F)
```

Average across these three models

```r
MA.ests<-model.avg(cg_cand2, revised.var = TRUE, fit=F)
MA.ests_uns <-model.avg(uns2, revised.var = TRUE, fit=T) # used for predictions
summary(MA.ests)
```

```
##
## Call:
## model.avg(object = cg_cand2, fit = F, revised.var = TRUE)
##
## Component model call:
## glmmTMB(formula = ninetyfive_s ~ <3 unique rhs>, data = dat2, REML = F,
##      ziformula = ~0, dispformula = ~1)
##
## Component models:
##       df logLik   AICc delta weight
## 345    6 -97.77 208.60  0.00   0.43
## 2345   7 -96.67 208.79  0.18   0.39
## 12345  8 -96.25 210.37  1.77   0.18
##
## Term codes:
##                       cond(prod2_s)                          cond(prod_s)
##                                   1                                     2
##                   cond(Treatment_c)           cond(YrsSinceExclosure_s)
##                                   3                                     4
## cond(Treatment:YrsSinceExclosure_s)
##                                   5
##
## Model-averaged coefficients:
## (full average)
##                                         Estimate Std. Error Adjusted SE
## cond((Int))                           -7.908e-07  1.066e-01   1.083e-01
## cond(Treatment_c)                      5.107e-01  1.221e-01   1.240e-01
## cond(YrsSinceExclosure_s)              2.671e-01  1.271e-01   1.290e-01
## cond(TreatmentExclosure:YrsSinceExclosure_s)  3.737e-01  1.228e-01   1.247e-01
## cond(prod_s)                           1.477e-01  2.311e-01   2.328e-01
```

```
## cond(prod2_s)                               -5.714e-02  1.908e-01   1.926e-01
##                                             z value Pr(>|z|)
## cond((Int))                                   0.000  0.99999
## cond(Treatment_c)                             4.118 3.82e-05 ***
## cond(YrsSinceExclosure_s)                     2.071  0.03839 *
## cond(TreatmentExclosure:YrsSinceExclosure_s)  2.996  0.00273 **
## cond(prod_s)                                  0.634  0.52592
## cond(prod2_s)                                 0.297  0.76673
##
## (conditional average)
##                                               Estimate Std. Error Adjusted SE
## cond((Int))                                  -7.908e-07  1.066e-01   1.083e-01
## cond(Treatment_c)                             5.107e-01  1.221e-01   1.240e-01
## cond(YrsSinceExclosure_s)                     2.671e-01  1.271e-01   1.290e-01
## cond(TreatmentExclosure:YrsSinceExclosure_s)  3.737e-01  1.228e-01   1.247e-01
## cond(prod_s)                                  2.591e-01  2.546e-01   2.574e-01
## cond(prod2_s)                                -3.214e-01  3.463e-01   3.518e-01
##                                             z value Pr(>|z|)
## cond((Int))                                   0.000  0.99999
## cond(Treatment_c)                             4.118 3.82e-05 ***
## cond(YrsSinceExclosure_s)                     2.071  0.03839 *
## cond(TreatmentExclosure:YrsSinceExclosure_s)  2.996  0.00273 **
## cond(prod_s)                                  1.007  0.31408
## cond(prod2_s)                                 0.914  0.36096
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(MA.ests_uns)
```

```
##
## Call:
## model.avg(object = get.models(object = uns2, subset = NA), revised.var = TRUE)
##
## Component model call:
## glmmTMB(formula = ninetyfive ~ <3 unique rhs>, data = dat2, family =
##      gaussian, ziformula = ~0, dispformula = ~1, REML = F)
##
## Component models:
##       df  logLik   AICc delta weight
## 345    6 -129.33 271.72  0.00   0.43
## 1345   7 -128.23 271.90  0.18   0.39
## 12345  8 -127.81 273.48  1.77   0.18
##
## Term codes:
##                        cond(prod)                        cond(prod2)
##                                 1                                  2
##                   cond(Treatment)            cond(YrsSinceExclosure)
##                                 3                                  4
## cond(Treatment:YrsSinceExclosure)
##                                 5
##
## Model-averaged coefficients:
## (full average)
##                                             Estimate Std. Error Adjusted SE
```

```
## cond((Int))                                  0.22045    0.66370     0.67299
## cond(TreatmentExclosure)                    -0.98246    0.59199     0.60117
## cond(YrsSinceExclosure)                      0.17213    0.08188     0.08312
## cond(TreatmentExclosure:YrsSinceExclosure)  0.24084    0.07916     0.08038
## cond(prod)                                   1.93715    3.03122     3.05428
## cond(prod2)                                 -1.45425    4.85719     4.90215
##                                             z value Pr(>|z|)
## cond((Int))                                   0.328  0.74324
## cond(TreatmentExclosure)                      1.634  0.10221
## cond(YrsSinceExclosure)                       2.071  0.03839 *
## cond(TreatmentExclosure:YrsSinceExclosure)   2.996  0.00273 **
## cond(prod)                                    0.634  0.52592
## cond(prod2)                                   0.297  0.76673
##
## (conditional average)
##                                             Estimate Std. Error Adjusted SE
## cond((Int))                                  0.22045    0.66370     0.67299
## cond(TreatmentExclosure)                    -0.98246    0.59199     0.60117
## cond(YrsSinceExclosure)                      0.17213    0.08188     0.08312
## cond(TreatmentExclosure:YrsSinceExclosure)  0.24084    0.07916     0.08038
## cond(prod)                                   3.39894    3.33965     3.37633
## cond(prod2)                                 -8.17867    8.81396     8.95285
##                                             z value Pr(>|z|)
## cond((Int))                                   0.328  0.74324
## cond(TreatmentExclosure)                      1.634  0.10221
## cond(YrsSinceExclosure)                       2.071  0.03839 *
## cond(TreatmentExclosure:YrsSinceExclosure)   2.996  0.00273 **
## cond(prod)                                    1.007  0.31408
## cond(prod2)                                   0.914  0.36097
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

No they're not the same. Not sure why, but part of the reason must be that treatment has been centred.

Get parameter weights

```r
importance(MA.ests)
```

```
##                        cond(Treatment_c) cond(YrsSinceExclosure_s)
## Sum of weights:        1.00              1.00
## N containing models:      3                 3
##                        cond(Treatment:YrsSinceExclosure_s) cond(prod_s)
## Sum of weights:        1.00                                0.57
## N containing models:      3                                   2
##                        cond(prod2_s)
## Sum of weights:        0.18
## N containing models:      1
```

prod and prod2 and highly collinear, and more so when using prod2 compared to I(prod^2)

We want to get at the values to plot them. The intercept dont make sense in averaged models, so remove intercept from table

```r
(macdf <- data.frame(summary(MA.ests)$coefmat.subset[-1,]))
```

```
##                                               Estimate Std..Error Adjusted.SE
## cond(Treatment_c)                            0.5106672  0.1221097   0.1240031
## cond(YrsSinceExclosure_s)                    0.2670830  0.1270571   0.1289811
## cond(TreatmentExclosure:YrsSinceExclosure_s)  0.3737120  0.1228259   0.1247304
## cond(prod_s)                                 0.2590859  0.2545665   0.2573620
## cond(prod2_s)                               -0.3213552  0.3463164   0.3517733
##                                               z.value  Pr...z..
## cond(Treatment_c)                           4.1181814 0.0000382
## cond(YrsSinceExclosure_s)                   2.0707145 0.0383855
## cond(TreatmentExclosure:YrsSinceExclosure_s) 2.9961587 0.0027340
## cond(prod_s)                                1.0066983 0.3140798
## cond(prod2_s)                               0.9135293 0.3609643
```

get the weights and add them to the same table

```r
impdf<-data.frame(importance(MA.ests))
impdf <- as.numeric(impdf[,1])
macdf$importance.MA.ests.<-impdf # this works cause thyr ordered in the same way
macdf
```

```
##                                               Estimate Std..Error Adjusted.SE
## cond(Treatment_c)                            0.5106672  0.1221097   0.1240031
## cond(YrsSinceExclosure_s)                    0.2670830  0.1270571   0.1289811
## cond(TreatmentExclosure:YrsSinceExclosure_s)  0.3737120  0.1228259   0.1247304
## cond(prod_s)                                 0.2590859  0.2545665   0.2573620
## cond(prod2_s)                               -0.3213552  0.3463164   0.3517733
##                                               z.value  Pr...z..
## cond(Treatment_c)                           4.1181814 0.0000382
## cond(YrsSinceExclosure_s)                   2.0707145 0.0383855
## cond(TreatmentExclosure:YrsSinceExclosure_s) 2.9961587 0.0027340
## cond(prod_s)                                1.0066983 0.3140798
## cond(prod2_s)                               0.9135293 0.3609643
##                                             importance.MA.ests.
## cond(Treatment_c)                                     1.0000000
## cond(YrsSinceExclosure_s)                             1.0000000
## cond(TreatmentExclosure:YrsSinceExclosure_s)          1.0000000
## cond(prod_s)                                          0.5699286
## cond(prod2_s)                                         0.1778098
```

Then we also need the 95 CIs

```r
cis <- confint(MA.ests)
cis <- cis[-1,]
cis <- as.data.frame(cis)
macdf$low <- cis[,1]
macdf$high <- cis[,2]
macdf
```

```
##                                               Estimate Std..Error Adjusted.SE
## cond(Treatment_c)                            0.5106672  0.1221097   0.1240031
```

```
## cond(YrsSinceExclosure_s)                    0.2670830  0.1270571   0.1289811
## cond(TreatmentExclosure:YrsSinceExclosure_s)  0.3737120  0.1228259   0.1247304
## cond(prod_s)                                 0.2590859  0.2545665   0.2573620
## cond(prod2_s)                               -0.3213552  0.3463164   0.3517733
##                                               z.value  Pr...z..
## cond(Treatment_c)                           4.1181814 0.0000382
## cond(YrsSinceExclosure_s)                   2.0707145 0.0383855
## cond(TreatmentExclosure:YrsSinceExclosure_s) 2.9961587 0.0027340
## cond(prod_s)                                1.0066983 0.3140798
## cond(prod2_s)                               0.9135293 0.3609643
##                                             importance.MA.ests.        low
## cond(Treatment_c)                                     1.0000000  0.26762563
## cond(YrsSinceExclosure_s)                             1.0000000  0.01428473
## cond(TreatmentExclosure:YrsSinceExclosure_s)          1.0000000  0.12924496
## cond(prod_s)                                          0.5699286 -0.24533440
## cond(prod2_s)                                         0.1778098 -1.01081807
##                                                   high
## cond(Treatment_c)                           0.7537088
## cond(YrsSinceExclosure_s)                   0.5198812
## cond(TreatmentExclosure:YrsSinceExclosure_s) 0.6181791
## cond(prod_s)                                0.7635062
## cond(prod2_s)                               0.3681077
```

Lets order them after effect size

```
macdf3<-macdf[order(macdf$Estimate),]
```

Fix names

```
Row.names<-c('Productivity squared',
             'Productivity',
             'Experimental duration (ED)',
             'HE x ED',
             'Herbivore Exclusion (HE)'
                    )
macdf3 <- cbind(Row.names, macdf3)
macdf3
```

```
##                                                               Row.names
## cond(prod2_s)                                      Productivity squared
## cond(prod_s)                                               Productivity
## cond(YrsSinceExclosure_s)                   Experimental duration (ED)
## cond(TreatmentExclosure:YrsSinceExclosure_s)                    HE x ED
## cond(Treatment_c)                             Herbivore Exclusion (HE)
##                                               Estimate Std..Error Adjusted.SE
## cond(prod2_s)                               -0.3213552  0.3463164   0.3517733
## cond(prod_s)                                 0.2590859  0.2545665   0.2573620
## cond(YrsSinceExclosure_s)                    0.2670830  0.1270571   0.1289811
## cond(TreatmentExclosure:YrsSinceExclosure_s)  0.3737120  0.1228259   0.1247304
## cond(Treatment_c)                            0.5106672  0.1221097   0.1240031
##                                               z.value  Pr...z..
## cond(prod2_s)                               0.9135293 0.3609643
## cond(prod_s)                                1.0066983 0.3140798
```

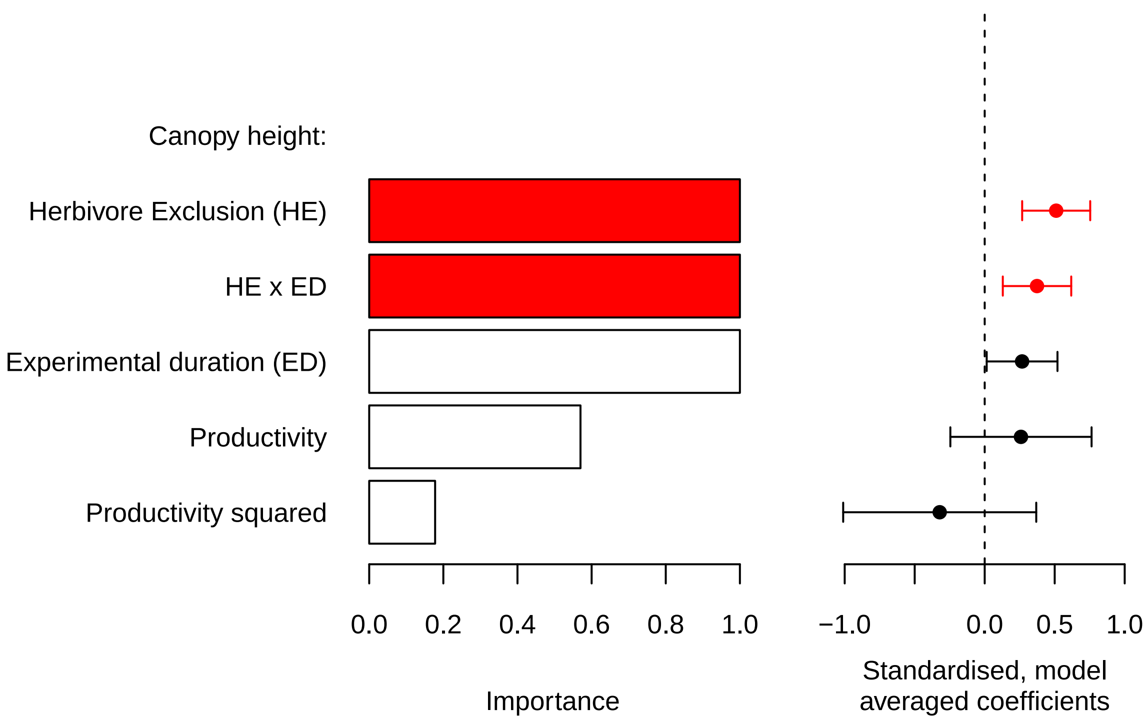
```
## cond(YrsSinceExclosure_s)                     2.0707145 0.0383855
## cond(TreatmentExclosure:YrsSinceExclosure_s) 2.9961587 0.0027340
## cond(Treatment_c)                            4.1181814 0.0000382
##                                              importance.MA.ests.         low
## cond(prod2_s)                                          0.1778098 -1.01081807
## cond(prod_s)                                           0.5699286 -0.24533440
## cond(YrsSinceExclosure_s)                              1.0000000  0.01428473
## cond(TreatmentExclosure:YrsSinceExclosure_s)           1.0000000  0.12924496
## cond(Treatment_c)                                      1.0000000  0.26762563
##                                                   high
## cond(prod2_s)                                0.3681077
## cond(prod_s)                                 0.7635062
## cond(YrsSinceExclosure_s)                    0.5198812
## cond(TreatmentExclosure:YrsSinceExclosure_s) 0.6181791
## cond(Treatment_c)                            0.7537088
```

Then lets make the figure First, adding empty row

```r
macdf4 <- macdf3[-(1:nrow(macdf3)),]
macdf4[1,] <- c(NA, NA, NA, NA, NA, NA, NA, NA, NA)
macdf4[2,] <- c(NA, 100, NA, NA, NA, NA, NA, NA, NA)

macdf3 <- rbind(macdf3, macdf4)
macdf3$Row.names <- as.character(macdf3$Row.names)
macdf3$Row.names[6] <- "Canopy height:"
```

```r
canopyAvg %<a-% {

par(oma=c(1,10,1,1))
par(mfrow=c(1,2))
par(mar=c(5,0,1,1))
par(xpd=T)

barplot(macdf3$importance.MA.ests.,
        beside=T,horiz=T,
        names.arg=macdf3$Row.names,
        las=1,
        xlab='Importance',
        cex.axis=0.8,
        cex.names=0.8,
        cex.lab=0.8,
        col=c(0,0,0,2,2, 0))
par(mar=c(5,1,1,1))
b1 <- barplot(macdf3[,2],
            horiz=T,
            col=F,
            border=F,
            xlim=c(-1.2,1.2),
            las=1,
            xlab='Standardised, model\naveraged coefficients',
            cex.axis=0.8,
            cex.lab=0.8)
points(macdf3[,2], #est
```

```
        b1,
        pch=16,
        col=c(1,1,1,2,2, 1))
arrows(macdf3[,9],
        b1,
        macdf3[,8],
        b1,
        code=3,
        angle=90,
        length=0.05,
        col=c(1,1,1,2,2, 1))
par(xpd=F)
abline(v=0,lty=2)
}
canopyAvg
```



This figure can be exported but I'll wait and do it later

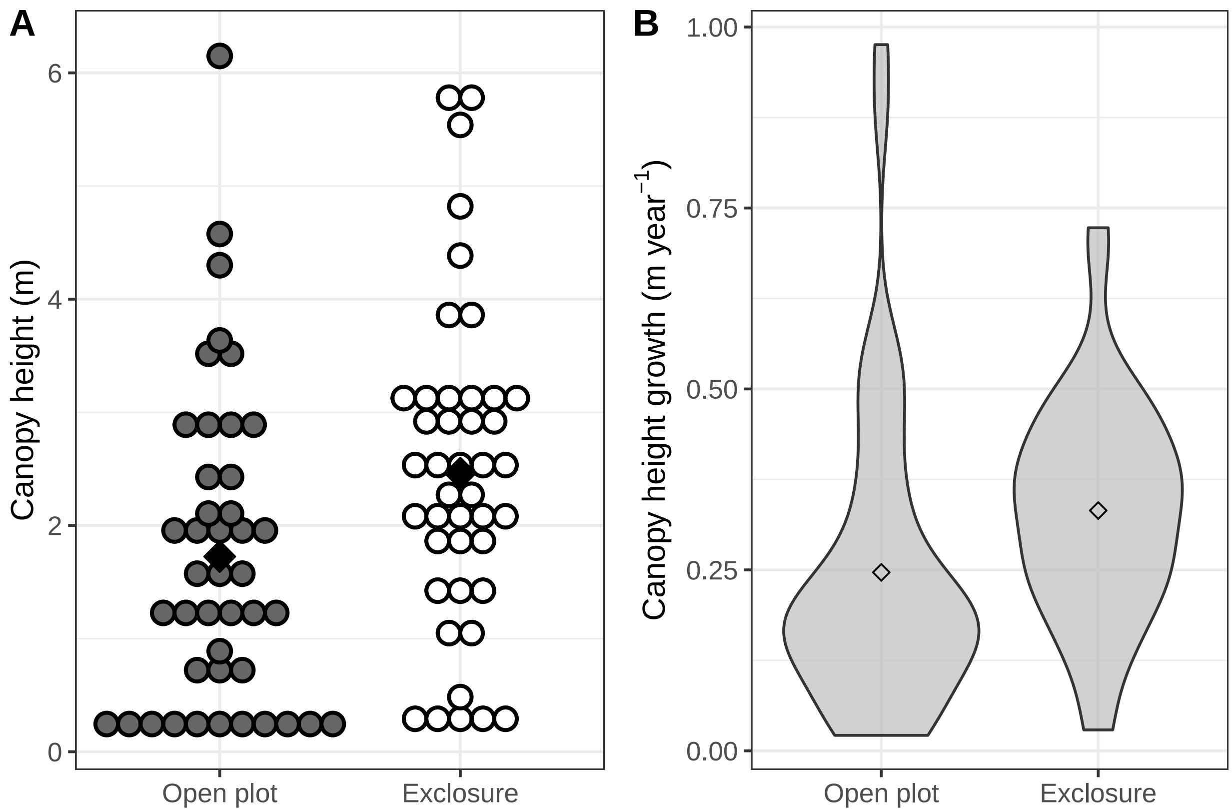
```
tiff("/home/anders/Documents/lidar ms/avgCanopyMod.tiff", height = 5, width=7, units="in", res=600)
canopyAvg
dev.off()
```

Dotplot

```
cg_viol <-
   ggplot(data = dat2, aes(x = Treatment, y = canopygrowth))+
  geom_violin(fill = "grey", alpha=0.7)+
  theme_bw()+
  theme(text = element_text(size = 12))+
  labs(y=expression(paste('Canopy height growth (m year'^'-1', ')')), x='')+
  stat_summary(fun.y=mean, geom="point", shape=23, size=2)+
  xlab("")


ch_viol <-
   ggplot(data = dat2, aes(x = Treatment, y = ninetyfive, fill=Treatment))+
 # geom_violin(fill = "grey", alpha=0.7)+
  theme_bw()+
  theme(text = element_text(size = 12))+
  labs(y='Canopy height (m)')+
  xlab("")+
  geom_dotplot(binaxis='y', stackdir='center',
               stroke=2)+
  scale_fill_manual(values = c("grey40", "white"))+
  guides(fill=F)+
  stat_summary(fun.y=mean, geom="point", shape=23, size=4, fill="black")

viol <- ggarrange(ch_viol, cg_viol,
                     labels = c("A", "B"),
                     ncol = 2, nrow = 1)
```

```
## `stat_bindot()` using `bins = 30`. Pick better value with `binwidth`.
```

```
viol
```




```
#tiff("/home/anders/Documents/lidar ms/violinPlot.tiff", height = 5, width=7, units="in", res=600)
viol
#dev.off()
```

Then we need to plot canopy height against experimental duration. There is a predict function in MuMIn.

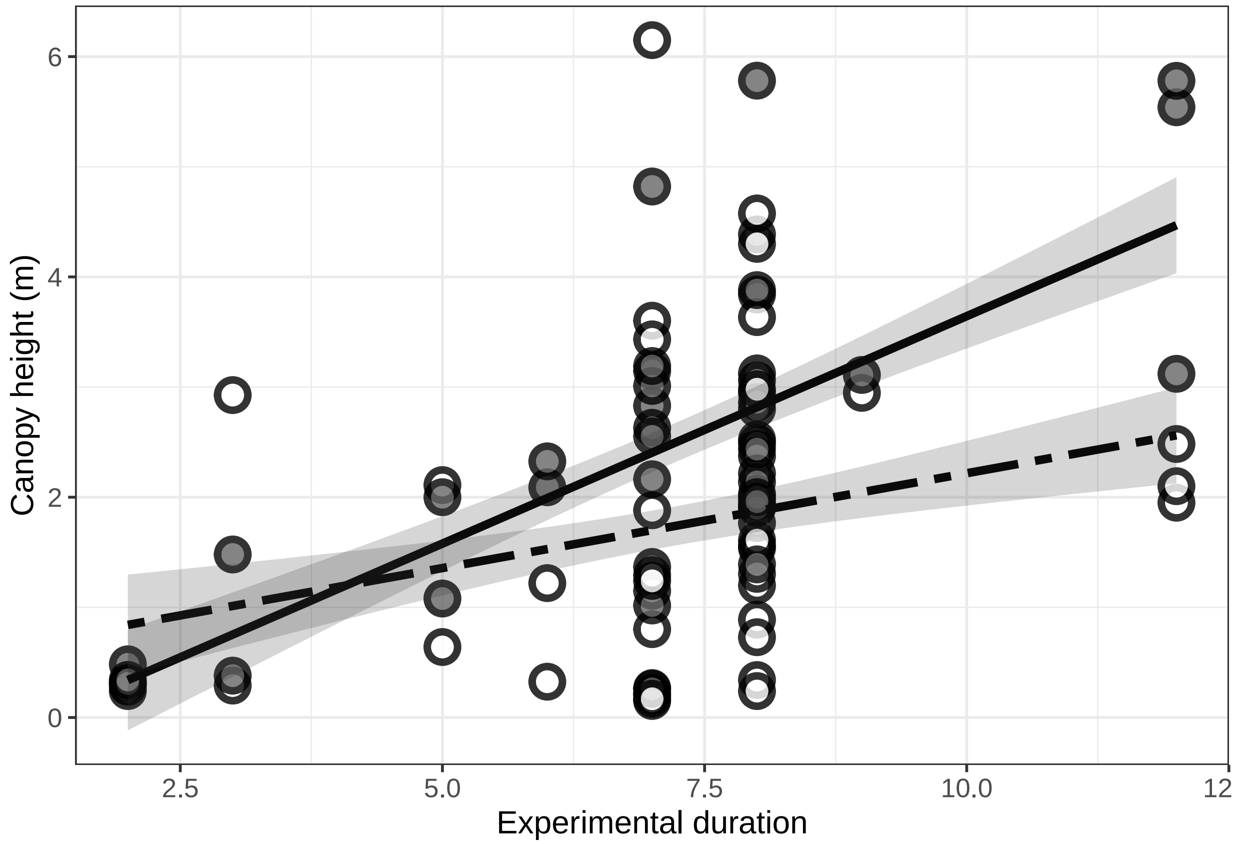
```
all.vars(formula(MA.ests_uns))
```

```
## [1] "ninetyfive"         "prod"               "prod2"
## [4] "Treatment"          "YrsSinceExclosure"
```

```
newd <- dat2
newd$YrsSinceExclosure <- rep(seq(from = min(dat2$YrsSinceExclosure),
                              to=max(dat2$YrsSinceExclosure), length.out = 43), 2)
newd$prod <- rep(mean(dat2$prod), nrow(newd))
newd$prod2 <- rep(mean(dat2$prod2), nrow(newd))
newd$LocalityName <- rep(NA, 86)
pred <- predict(MA.ests_uns,
                             se.fit = TRUE,
                             full=FALSE,
                             re.form=~0, # same as leaving out LocalityName
                newdata=newd
                )
```

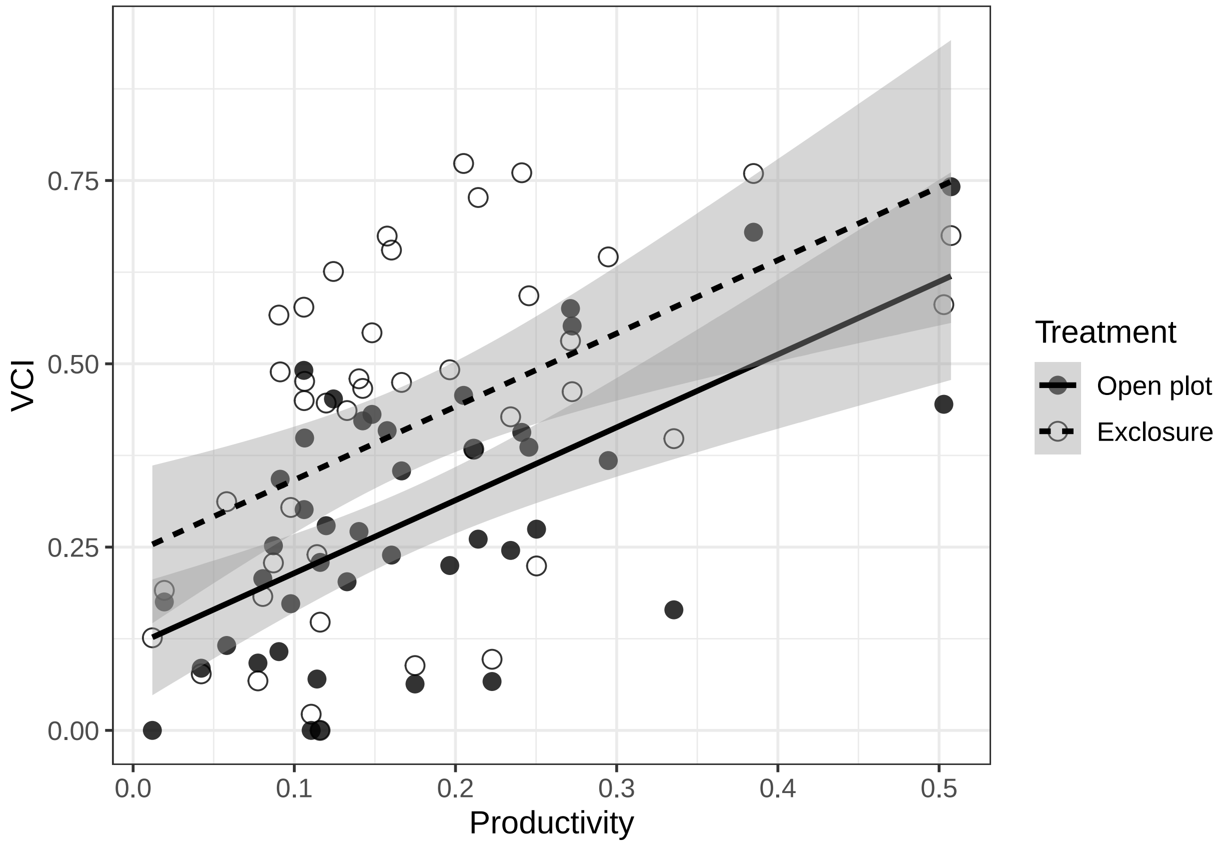
```
## Warning in predict.averaging(MA.ests_uns, se.fit = TRUE, full = FALSE, re.form =
## ~0, : argument 'full' ignored
```

```r
pred2 <- data.frame(Treatment = newd$Treatment,
                    YrsSinceExclosure      = newd$YrsSinceExclosure,
                    pred      = pred$fit,
                    se        = pred$se.fit)

(Canopy_line <-
    ggplot()+
  geom_point(data = dat2,
             aes(x = YrsSinceExclosure,
                 y = ninetyfive,
                 fill=Treatment),
             size=4, shape = 21, stroke=2, alpha=0.8)+
  scale_fill_manual(values = c("white", "grey40"))+
#   ,
#                     breaks=c("Exclosure", "Open plot"),
#                     labels=c("Exclosure", "Open plot"))+

  scale_linetype_manual("", values=c(6,1),
                       breaks=c("Exclosure", "Open plot"),
                    labels=c("Exclosure", "Open plot"))+

  geom_line(data=pred2,
            aes(x = YrsSinceExclosure, y=pred, linetype = Treatment),
            lwd=1.5)+

  geom_ribbon(data=pred2, aes(x = YrsSinceExclosure,
                 ymin=pred-se,
                 ymax=pred+se,
                 group = Treatment),
                 alpha=0.2,
                 linetype="blank")+
        theme_bw()+
  theme(legend.justification=c(1,0),
        legend.position=c(1,0),
        legend.background = element_blank(),
        legend.text = element_text(size=12),
        text = element_text(size = 12))+
guides(linetype=F, fill=F)+

  labs(y="Canopy height (m)")+
  xlab("Experimental duration"))
```

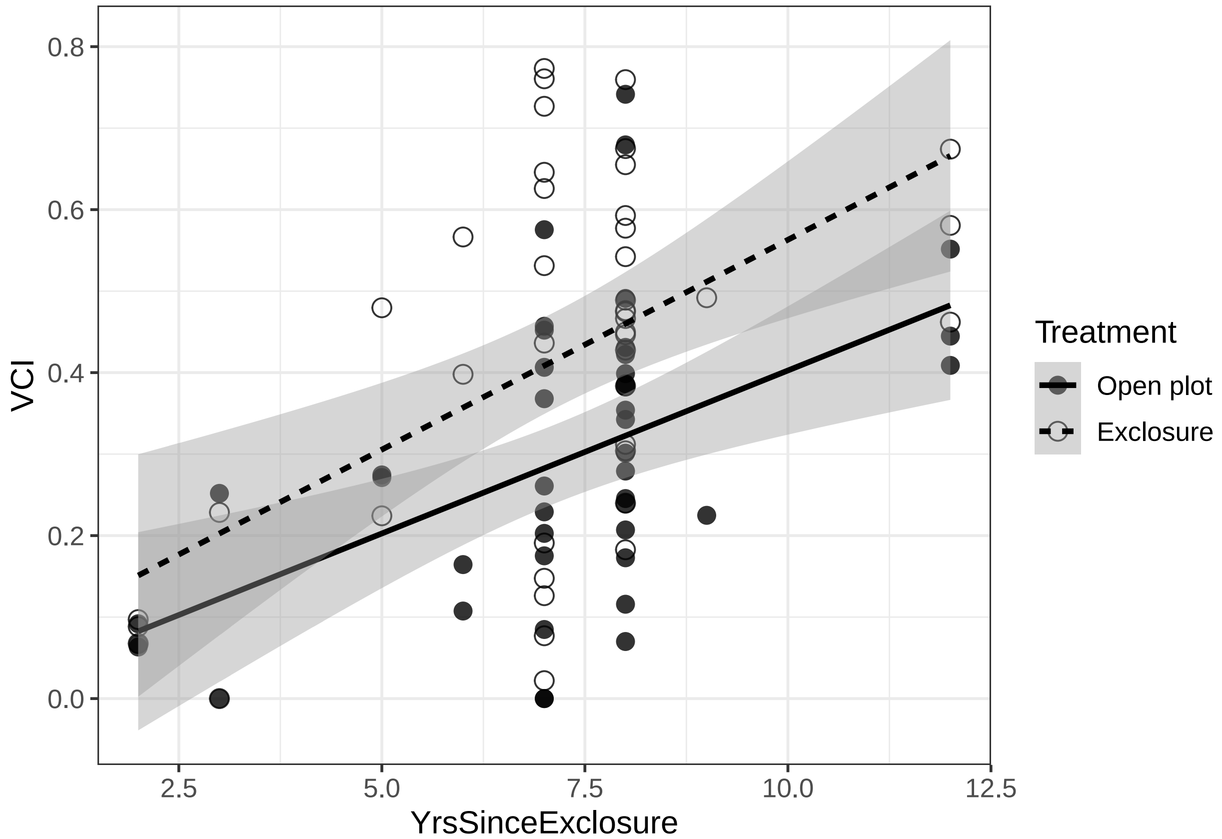


VCI

```
(vci_line <-
   ggplot(data = dat2, aes(x = prod, y = vci, shape=Treatment))+
  geom_point(size=3, alpha=0.8)+
  scale_shape_manual(values = c(16, 1))+
  geom_smooth(method="lm", aes(linetype=Treatment), colour="black")+
  theme_bw()+
  theme(text = element_text(size = 12))+
  labs(y="VCI")+
  xlab("Productivity"))
```




```
(vci_line2 <-
   ggplot(data = dat2, aes(x = YrsSinceExclosure, y = vci, shape=Treatment))+
  geom_point(size=3, alpha=0.8)+
  scale_shape_manual(values = c(16, 1))+
  geom_smooth(method="lm", aes(linetype=Treatment), colour="black")+
  theme_bw()+
  theme(text = element_text(size = 12))+
  labs(y="VCI")+
  xlab("YrsSinceExclosure"))
```




```r
dat2$vci_s <- scale(dat2$vci)[,1]

mod_vci <- glmmTMB(vci ~ Treatment * prod + YrsSinceExclosure + YrsSinceExclosure:Treatment+ prod2 +  (
  data = dat2, REML=F, family = gaussian)

mod_vci_s <- glmmTMB(vci_s ~ Treatment_c * prod_s + YrsSinceExclosure_s + YrsSinceExclosure_s:Treatment_
  data = dat2, REML=F,  family = gaussian)

summary(mod_vci)
```

```
##  Family: gaussian  ( identity )
## Formula:
## vci ~ Treatment * prod + YrsSinceExclosure + YrsSinceExclosure:Treatment +
##     prod2 + (1 | LocalityName)
## Data: dat2
##
##      AIC      BIC   logLik deviance df.resid
##    -85.7    -63.6     51.9   -103.7       77
##
## Random effects:
##
## Conditional model:
##  Groups       Name        Variance Std.Dev.
##  LocalityName (Intercept) 0.010808 0.10396
##  Residual                 0.009783 0.09891
```

```
## Number of obs: 86, groups:  LocalityName, 43
##
## Dispersion estimate for gaussian family (sigma^2): 0.00978
##
## Conditional model:
##                                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)                         -0.175823   0.093685  -1.877  0.06055 .
## TreatmentExclosure                   0.050445   0.073616   0.685  0.49319
## prod                                 1.895717   0.579650   3.270  0.00107 **
## YrsSinceExclosure                    0.031641   0.010239   3.090  0.00200 **
## prod2                               -2.175502   1.115863  -1.950  0.05122 .
## TreatmentExclosure:prod             -0.059660   0.201850  -0.296  0.76756
## TreatmentExclosure:YrsSinceExclosure  0.012251   0.009916   1.235  0.21665
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#dmod_vci <- dredge(mod_vci, beta="none", rank = "AICc")
#write_rds(dmod_vci, "../data/dmod_vci.RData")
dmod_vci <- read_rds("../data/dmod_vci.RData")
#dmod_vci_s <- dredge(mod_vci_s, beta="none", rank = "AICc")
#write_rds(dmod_vci_s, "../data/dmod_vci_s.RData")
dmod_vci_s <- read_rds("../data/dmod_vci_s.RData")
```

```
(dmod_vci2 <- subset(dmod_vci, delta <2))
```

```
## Global model call: glmmTMB(formula = vci ~ Treatment * prod + YrsSinceExclosure +
##     YrsSinceExclosure:Treatment + prod2 + (1 | LocalityName),
##     data = dat2, family = gaussian, REML = F, ziformula = ~0,
##     dispformula = ~1)
## ---
## Model selection table
##    cnd((Int)) dsp((Int)) cnd(prd) cnd(pr2) cnd(Trt) cnd(YSE) cnd(Trt:YSE) df
## 16    -0.2144          +   1.8660   -2.176        +  0.03777               7
## 48    -0.1733          +   1.8660   -2.176        +  0.03202            +  8
## 14    -0.1063          +   0.8104                 +  0.03549               6
##    logLik  AICc delta weight
## 16 51.115 -86.8  0.00  0.469
## 48 51.822 -85.8  1.02  0.282
## 14 49.294 -85.5  1.27  0.249
## Models ranked by AICc(x)
## Random terms (all models):
## 'cond(1 | LocalityName)'
```

```
(dmod_vci_s2 <- subset(dmod_vci_s, delta <2))
```

```
## Global model call: glmmTMB(formula = vci_s ~ Treatment_c * prod_s + YrsSinceExclosure_s +
##     YrsSinceExclosure_s:Treatment_c + prod2_s + (1 | LocalityName),
##     data = dat2, family = gaussian, REML = F, ziformula = ~0,
##     dispformula = ~1)
## ---
## Model selection table
##    cnd((Int)) dsp((Int)) cnd(pr2_s) cnd(prd_s) cnd(Trt_c) cnd(YSE_s)
```

```
## 16  6.058e-07          +    -0.5852     0.9737     0.6049     0.4012
## 48 -1.712e-07          +    -0.5852     0.9737     0.6049     0.4012
## 15 -1.645e-07          +                0.4229     0.6049     0.3770
##    cnd(Trt_c:YSE_s) df  logLik  AICc delta weight
## 16                   7 -82.767 181.0  0.00  0.469
## 48           0.1221  8 -82.060 182.0  1.02  0.282
## 15                   6 -84.588 182.2  1.27  0.249
## Models ranked by AICc(x)
## Random terms (all models):
## 'cond(1 | LocalityName)'
```

```r
importance(dmod_vci2)
```

```
##                      cond(prod) cond(Treatment) cond(YrsSinceExclosure)
## Sum of weights:      1.00       1.00            1.00
## N containing models:    3          3               3
##                      cond(prod2) cond(Treatment:YrsSinceExclosure)
## Sum of weights:      0.75        0.28
## N containing models:    2           1
```

Lets export this as a table for the supplementary

```r
temp <- as.data.frame(dmod_vci2)
temp <- temp[,-2]
names(temp) <- c("Intercept",
                      "Productivity (P)",
                      "Productivity squared",
                 "Herbivore Exclusion (HE)",
                      "Experimental duration (ED)",
                      "HE x P",
                      "HE x ED",
                      "df",
                      "log likelihood",
                      "AICc",
                      "delta AICc",
                      "weight"
                      )
temp[is.na(temp)] <- 0
```

```
## Warning in `[<-.factor`(`*tmp*`, thisvar, value = 0): invalid factor level, NA
## generated

## Warning in `[<-.factor`(`*tmp*`, thisvar, value = 0): invalid factor level, NA
## generated
```

```r
#write.csv(temp, "../output/VCIModelSet_unstandardized.csv", row.names = F)
```

Average across these three models

```r
VCIavg     <-model.avg(dmod_vci_s2, revised.var = TRUE, fit=F)
VCIavg_uns <-model.avg(dmod_vci2, revised.var = TRUE, fit=T) # used for predictions
summary(VCIavg_uns)
```

```
##
## Call:
## model.avg(object = get.models(object = dmod_vci2, subset = NA),
##     revised.var = TRUE)
##
## Component model call:
## glmmTMB(formula = vci ~ <3 unique rhs>, data = dat2, family = gaussian,
##      ziformula = ~0, dispformula = ~1, REML = F)
##
## Component models:
##       df logLik   AICc delta weight
## 1234   7  51.11 -86.79  0.00   0.47
## 12345  8  51.82 -85.77  1.02   0.28
## 134    6  49.29 -85.52  1.27   0.25
##
## Term codes:
##                      cond(prod)                        cond(prod2)
##                               1                                  2
##                 cond(Treatment)           cond(YrsSinceExclosure)
##                               3                                  4
## cond(Treatment:YrsSinceExclosure)
##                               5
##
## Model-averaged coefficients:
## (full average)
##                                          Estimate Std. Error Adjusted SE
## cond((Int))                             -0.175906   0.095406    0.096584
## cond(prod)                               1.603261   0.679582    0.685432
## cond(prod2)                             -1.634213   1.349045    1.359923
## cond(TreatmentExclosure)                 0.104397   0.056133    0.056635
## cond(YrsSinceExclosure)                  0.035580   0.009693    0.009835
## cond(TreatmentExclosure:YrsSinceExclosure)  0.003239   0.007256    0.007313
##                                         z value Pr(>|z|)
## cond((Int))                               1.821 0.068565 .
## cond(prod)                                2.339 0.019333 *
## cond(prod2)                               1.202 0.229482
## cond(TreatmentExclosure)                  1.843 0.065279 .
## cond(YrsSinceExclosure)                   3.618 0.000297 ***
## cond(TreatmentExclosure:YrsSinceExclosure)   0.443 0.657844
##
## (conditional average)
##                                          Estimate Std. Error Adjusted SE
## cond((Int))                             -0.175906   0.095406    0.096584
## cond(prod)                               1.603261   0.679582    0.685432
## cond(prod2)                             -2.175501   1.115862    1.133304
## cond(TreatmentExclosure)                 0.104397   0.056133    0.056635
## cond(YrsSinceExclosure)                  0.035580   0.009693    0.009835
## cond(TreatmentExclosure:YrsSinceExclosure)  0.011495   0.009590    0.009742
##                                         z value Pr(>|z|)
## cond((Int))                               1.821 0.068565 .
## cond(prod)                                2.339 0.019333 *
## cond(prod2)                               1.920 0.054907 .
## cond(TreatmentExclosure)                  1.843 0.065279 .
## cond(YrsSinceExclosure)                   3.618 0.000297 ***
```

```
## cond(TreatmentExclosure:YrsSinceExclosure)   1.180 0.237988
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Lets add the standardized stuff to the canopy height figure

```r
(figdat <- data.frame(summary(VCIavg)$coefmat.subset[-1,]))
```

```
##                                      Estimate Std..Error Adjusted.SE
## cond(prod2_s)                      -0.5852135 0.30017001  0.30486180
## cond(prod_s)                        0.8366758 0.35464661  0.35769925
## cond(Treatment_c)                   0.6048887 0.10249459  0.10408948
## cond(YrsSinceExclosure_s)           0.3951782 0.09656362  0.09804852
## cond(Treatment_c:YrsSinceExclosure_s)  0.1221165 0.10188056  0.10348589
##                                       z.value  Pr...z..
## cond(prod2_s)                      1.919603 0.0549081
## cond(prod_s)                       2.339048 0.0193329
## cond(Treatment_c)                  5.811238 0.0000000
## cond(YrsSinceExclosure_s)          4.030436 0.0000557
## cond(Treatment_c:YrsSinceExclosure_s) 1.180031 0.2379880
```

get the weights and add them to the same table

```r
VCIimpdf<-data.frame(importance(VCIavg))
VCIimpdf[c(4,1,2,3,5),]
```

```
## [1] 0.7511891 1.0000000 1.0000000 1.0000000 0.2817524
```

```r
VCIimpdf <- as.numeric(VCIimpdf[,1])
figdat$importance.MA.ests.<-VCIimpdf[c(4,1,2,3,5)]
figdat
```

```
##                                      Estimate Std..Error Adjusted.SE
## cond(prod2_s)                      -0.5852135 0.30017001  0.30486180
## cond(prod_s)                        0.8366758 0.35464661  0.35769925
## cond(Treatment_c)                   0.6048887 0.10249459  0.10408948
## cond(YrsSinceExclosure_s)           0.3951782 0.09656362  0.09804852
## cond(Treatment_c:YrsSinceExclosure_s)  0.1221165 0.10188056  0.10348589
##                                       z.value  Pr...z.. importance.MA.ests.
## cond(prod2_s)                      1.919603 0.0549081           0.7511891
## cond(prod_s)                       2.339048 0.0193329           1.0000000
## cond(Treatment_c)                  5.811238 0.0000000           1.0000000
## cond(YrsSinceExclosure_s)          4.030436 0.0000557           1.0000000
## cond(Treatment_c:YrsSinceExclosure_s) 1.180031 0.2379880           0.2817524
```

Then we also need the 95 CIs

```r
cis <- confint(VCIavg)
cis <- cis[-1,]
cis <- as.data.frame(cis)
figdat$low <- cis[,1]
figdat$high <- cis[,2]
figdat
```

```
##                                           Estimate Std..Error Adjusted.SE
## cond(prod2_s)                           -0.5852135 0.30017001  0.30486180
## cond(prod_s)                             0.8366758 0.35464661  0.35769925
## cond(Treatment_c)                        0.6048887 0.10249459  0.10408948
## cond(YrsSinceExclosure_s)                0.3951782 0.09656362  0.09804852
## cond(Treatment_c:YrsSinceExclosure_s)    0.1221165 0.10188056  0.10348589
##                                          z.value  Pr...z.. importance.MA.ests.
## cond(prod2_s)                           1.919603 0.0549081           0.7511891
## cond(prod_s)                            2.339048 0.0193329           1.0000000
## cond(Treatment_c)                       5.811238 0.0000000           1.0000000
## cond(YrsSinceExclosure_s)               4.030436 0.0000557           1.0000000
## cond(Treatment_c:YrsSinceExclosure_s)   1.180031 0.2379880           0.2817524
##                                                low       high
## cond(prod2_s)                           -1.1827316 0.01230466
## cond(prod_s)                             0.1355981 1.53775343
## cond(Treatment_c)                        0.4008771 0.80890034
## cond(YrsSinceExclosure_s)                0.2030067 0.58734979
## cond(Treatment_c:YrsSinceExclosure_s)   -0.0807121 0.32494515
```

Lets order them after effect size

```r
figdat<-figdat[order(figdat$Estimate),]
```

Fix names

```r
Row.names<-c('Productivity squared',
             'HE x ED',
             'Experimental duration (ED)',
             'Herbivore Exclusion (HE)',
             'Productivity'
                    )
figdat <- cbind(Row.names, figdat)
figdat
```

```
##                                                          Row.names   Estimate
## cond(prod2_s)                                 Productivity squared -0.5852135
## cond(Treatment_c:YrsSinceExclosure_s)                      HE x ED  0.1221165
## cond(YrsSinceExclosure_s)               Experimental duration (ED)  0.3951782
## cond(Treatment_c)                         Herbivore Exclusion (HE)  0.6048887
## cond(prod_s)                                          Productivity  0.8366758
##                                         Std..Error Adjusted.SE  z.value  Pr...z..
## cond(prod2_s)                           0.30017001  0.30486180 1.919603 0.0549081
## cond(Treatment_c:YrsSinceExclosure_s)   0.10188056  0.10348589 1.180031 0.2379880
## cond(YrsSinceExclosure_s)               0.09656362  0.09804852 4.030436 0.0000557
## cond(Treatment_c)                       0.10249459  0.10408948 5.811238 0.0000000
## cond(prod_s)                            0.35464661  0.35769925 2.339048 0.0193329
##                                         importance.MA.ests.        low       high
## cond(prod2_s)                                     0.7511891 -1.1827316 0.01230466
## cond(Treatment_c:YrsSinceExclosure_s)             0.2817524 -0.0807121 0.32494515
## cond(YrsSinceExclosure_s)                         1.0000000  0.2030067 0.58734979
## cond(Treatment_c)                                 1.0000000  0.4008771 0.80890034
## cond(prod_s)                                      1.0000000  0.1355981 1.53775343
```

Then lets make the figure First, adding empty row

```
b <- macdf3
colnames(macdf3)
```

```
## [1] "Row.names"            "Estimate"             "Std..Error"
## [4] "Adjusted.SE"          "z.value"              "Pr...z.."
## [7] "importance.MA.ests." "low"                  "high"
```

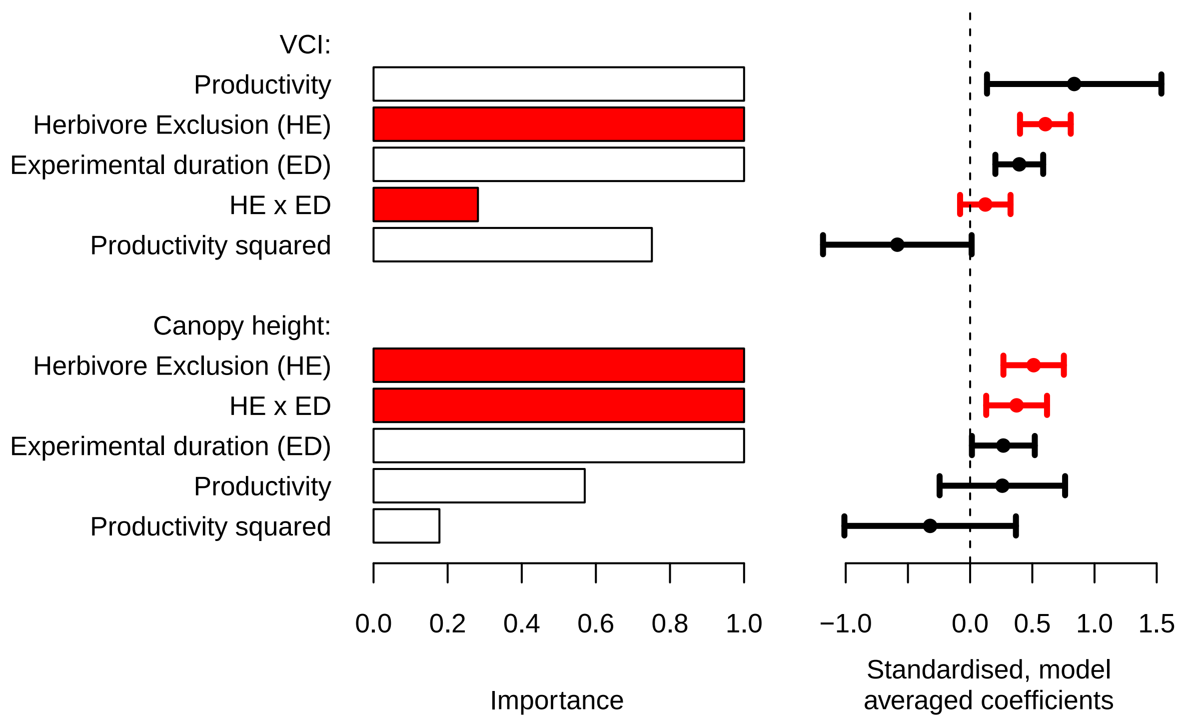
```
figdat2 <- figdat[-(1:nrow(figdat)),]
figdat2[1,] <- c(NA, 100, NA, NA, NA, NA, NA, NA, NA)
figdat3 <- rbind(figdat, figdat2)
figdat3$Row.names <- as.character(figdat3$Row.names)
figdat3$Row.names[6] <- "VCI:"
```

Combine:

```
figdat4 <- rbind(macdf3, figdat3)
```

```
Avg %<a-% {

par(oma=c(1,10,1,1))
par(mfrow=c(1,2))
par(mar=c(5,0,1,1))
par(xpd=T)

barplot(figdat4$importance.MA.ests.,
        beside=T,horiz=T,
        names.arg=figdat4$Row.names,
        las=1,
        xlab='Importance',
        cex.axis=0.8,
        cex.names=0.8,
        cex.lab=0.8,
        col=c(0,0,0,2,2,0,0,0,2,0,2,0,0))
par(mar=c(5,1,1,1))
b1 <- barplot(figdat4[,2],
            horiz=T,
            col=F,
            border=F,
            xlim=c(-1.2,1.5),
            las=1,
            xlab='Standardised, model\naveraged coefficients',
            cex.axis=0.8,
            cex.lab=0.8)
points(figdat4[,2], #est
       b1,
       pch=16,
       col=c(1,1,1,2,2,1,1,1,2,1,2,1,1))
arrows(figdat4[,9],
       b1,
       figdat4[,8],
```

```
        b1,
        code=3,
        angle=90,
        length=0.05,
        lwd=3,
        col=c(1,1,1,2,2,1,1,1,2,1,2,1,1))
par(xpd=F)
abline(v=0,lty=2)
}
Avg
```




```
tiff("/home/anders/Documents/lidar ms/modAveragedEst.tiff",
     height = 8, width=8, units="in", res=600)
Avg
dev.off()
```

Plot VCI against productivity

```
all.vars(formula(VCIavg_uns))
```

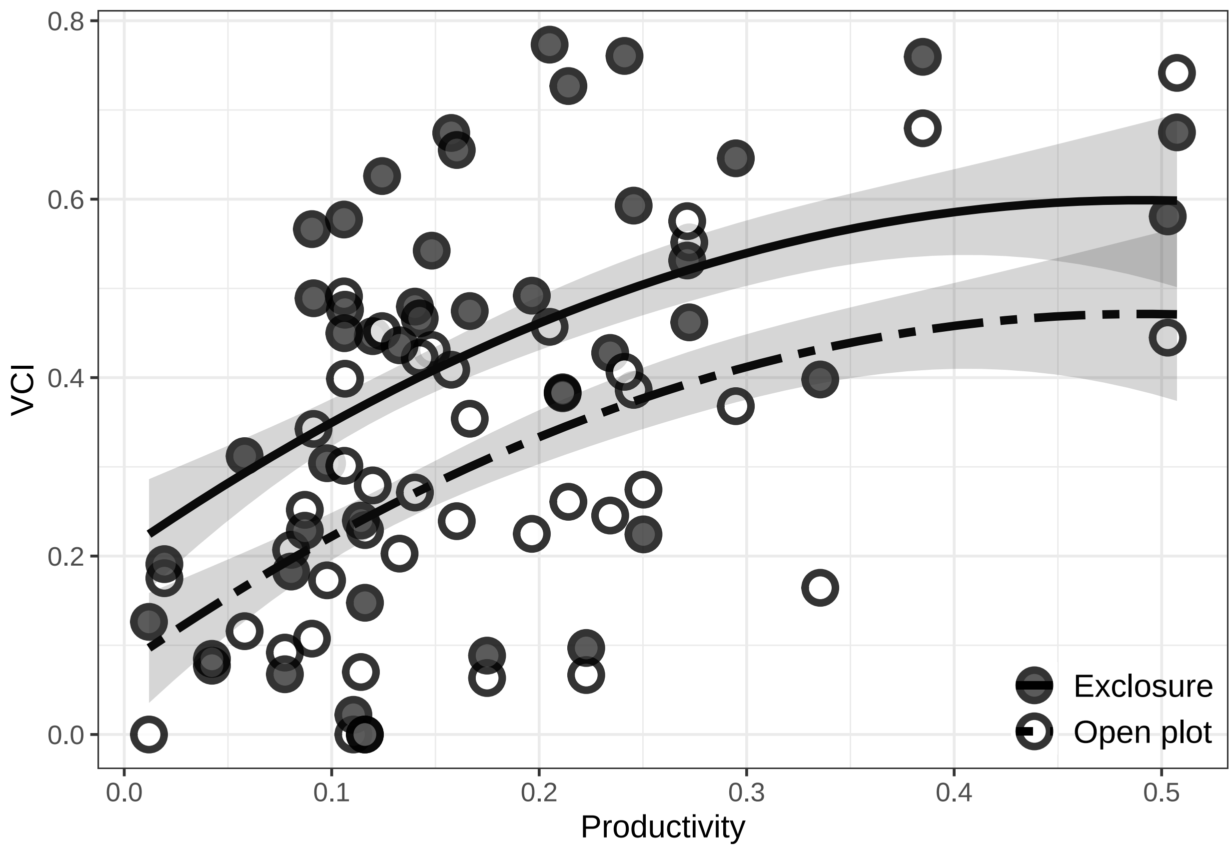
```
## [1] "vci"                "prod"               "prod2"
## [4] "Treatment"          "YrsSinceExclosure"
```

```r
newd <- dat2

newd$YrsSinceExclosure <-
    rep(mean(dat2$YrsSinceExclosure),length.out = 86)

newd$prod <-
     rep(seq(from = min(dat2$prod),
      to=max(dat2$prod), length.out = 43), 2)

newd$prod2 <- newd$prod*newd$prod

newd$LocalityName <- rep(NA, 86)

pred <- predict(VCIavg_uns,
                           se.fit = TRUE,
                           full=FALSE,
                           re.form=~0, # same as leaving out LocalityName
                newdata=newd
                )
```

```
## Warning in predict.averaging(VCIavg_uns, se.fit = TRUE, full = FALSE, re.form =
## ~0, : argument 'full' ignored
```

```r
pred2 <- data.frame(Treatment                = newd$Treatment,
                    YrsSinceExclosure        = newd$YrsSinceExclosure,
                    prod                     = newd$prod,
                    prod2                    = newd$prod2,
                    pred                     = pred$fit,
                    se                       = pred$se.fit)


  (vci_line <-    ggplot()+
     geom_point(data = dat2,
             aes(x = prod, y = vci, fill=Treatment),
             size=4, alpha=0.8, shape=21, stroke=2)+
      scale_fill_manual("",values = c("white", "grey20") ,
                    breaks=c("Exclosure", "Open plot"),
                    labels=c("Exclosure", "Open plot"))+
  geom_line(data=pred2,
            aes(x = prod, y=pred, linetype = Treatment),
            lwd=1.5)+

 scale_linetype_manual("", values=c(6,1),
                       breaks=c("Exclosure", "Open plot"),
                    labels=c("Exclosure", "Open plot"))+
  geom_ribbon(data=pred2, aes(x = prod,
                 ymin=pred-se,
                 ymax=pred+se,
                 group = Treatment),
                 alpha=0.2,
                 linetype="blank")+
       theme_bw()+
  theme(legend.justification=c(1,0),
```

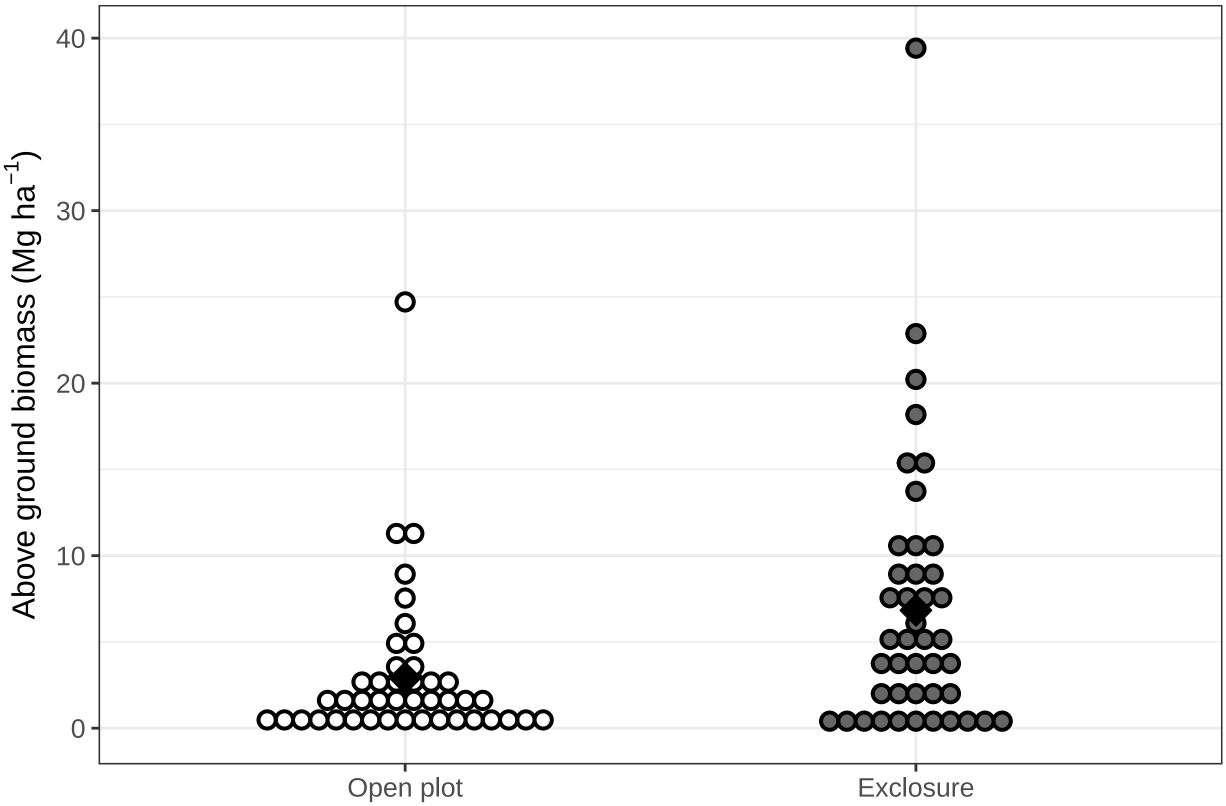
```
        legend.position=c(1,0),
        legend.background = element_blank(),
        legend.text = element_text(size=12),
        text = element_text(size = 12))+
  labs(y="VCI")+
  xlab("Productivity"))
```



Biomass

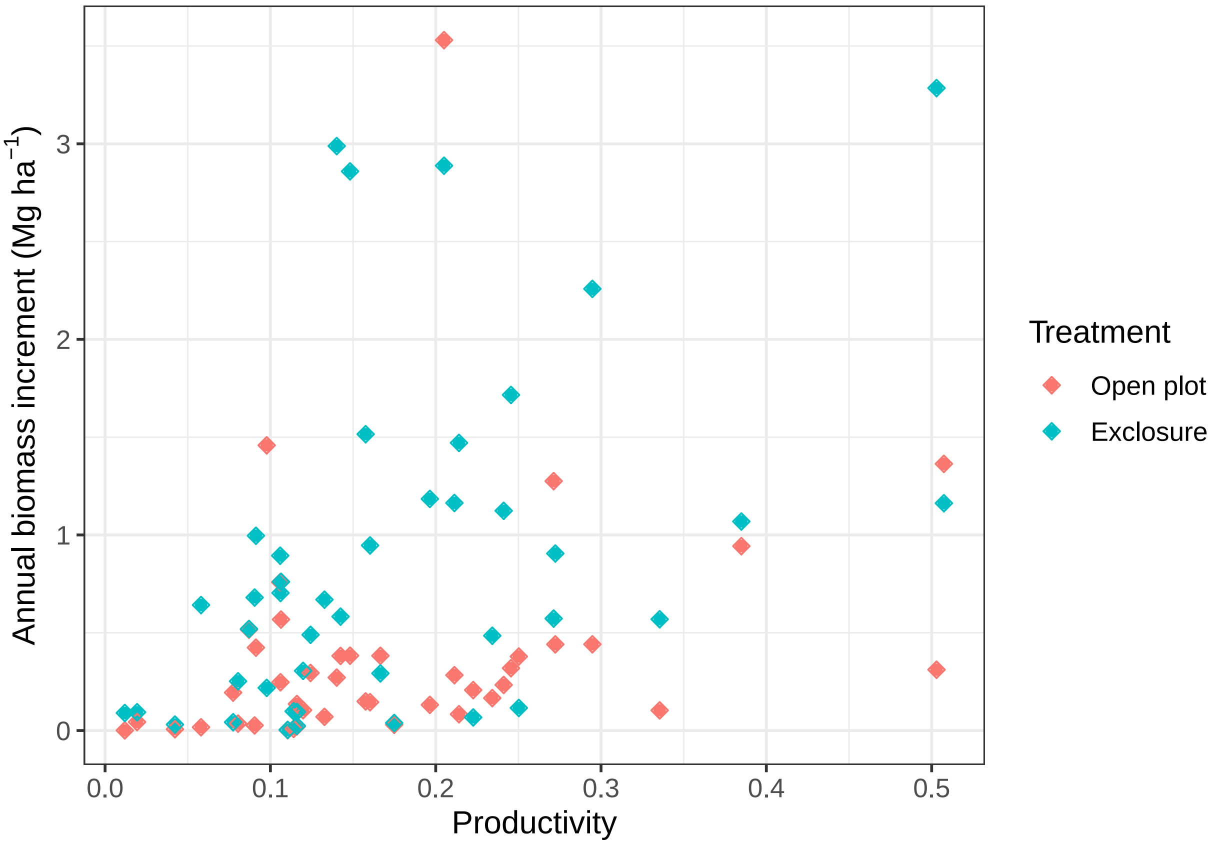
Violin/dotplot plot

```
(AGBviol <- ggplot(data =dat2, aes(x=Treatment, y=AGB,  fill=Treatment))+
  #geom_violin(fill = "grey", alpha=0.7)+
  theme_bw()+
  theme(text = element_text(size = 12))+
  labs(y=expression(paste('Above ground biomass (Mg ha'^'-1', ')')), x='')+
  xlab("")+
  geom_dotplot(binaxis='y', stackdir='center',
               stroke=2, binwidth = 1)+
   scale_fill_manual(values = c("white", "grey40"))+
  stat_summary(fun.y=mean, geom="point",
               shape=23, size=4, fill="black")+
  guides(fill = F))
```

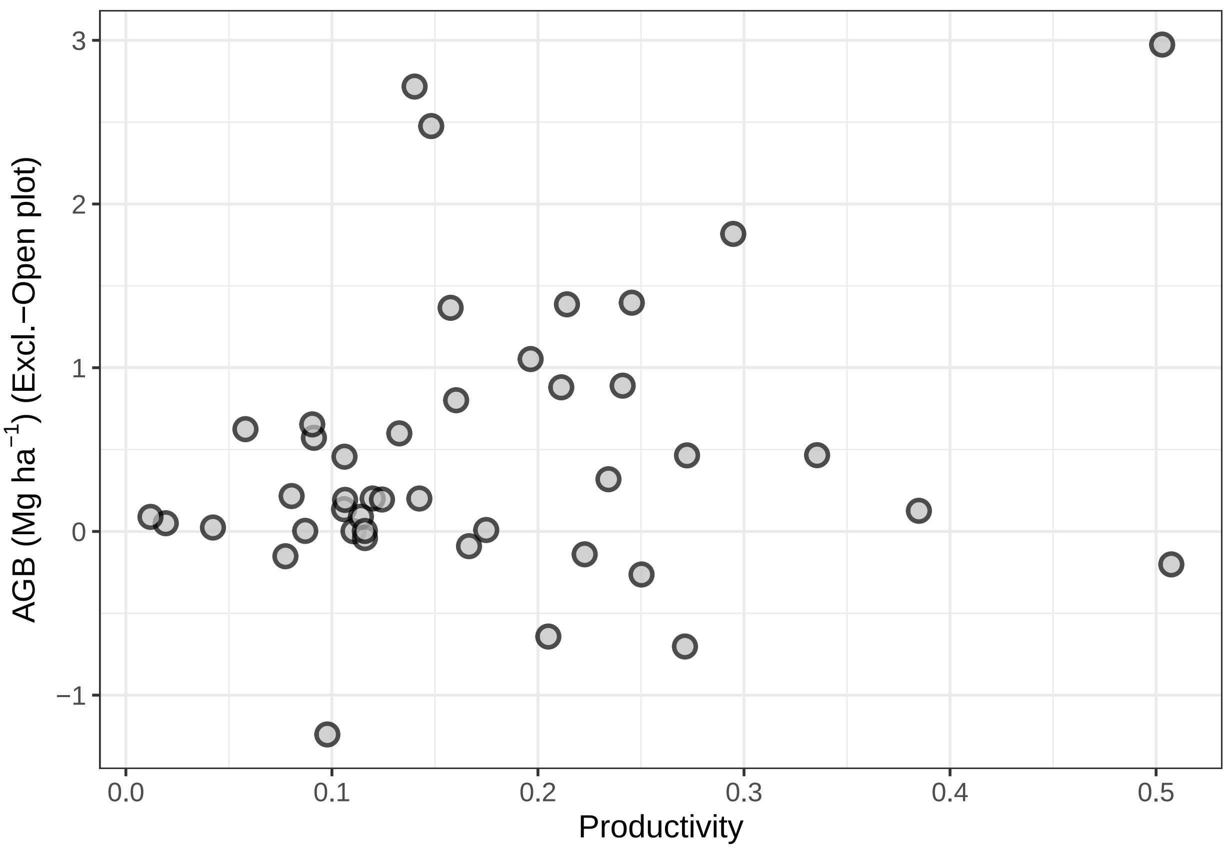


Plotting treatment difference could be done like this

```
library(reshape2)
datB <- dcast(data=dat2, value.var="MgAGBperYearAndHA", LocalityName+prod+YrsSinceExclosure~Treatment)
datB$diff <- datB$Exclosure - datB$`Open plot`
(agb_line <-
   ggplot(data = dat2, aes(x = prod, y = MgAGBperYearAndHA, colour=Treatment))+
  geom_point()+
  theme_bw()+
  theme(text = element_text(size = 12))+
  labs(y=expression(paste('Annual biomass increment (Mg ha'^'-1', ')')), x='')+
  stat_summary(fun.y=mean, geom="point", shape=23, size=2)+
  xlab("Productivity"))
```




```
(agb_line2 <-
   ggplot(data = datB, aes(x = prod, y = diff))+
  geom_point(fill="grey", colour="black", size=3, shape=21, stroke=1.2, alpha=.7)+
  theme_bw()+
  theme(text = element_text(size = 12))+
  #labs(y=expression(atop("Exclosure - Open plot",
  #             paste('annual biomass increment (Mg ha'^'-1', ')')), x=''))+
  labs(y=expression(
             paste('AGB (Mg ha'^'-1', ') (Excl.-Open plot)')))+
  xlab("Productivity"))
```




```
#tiff("/home/anders/Documents/lidar ms/AGB.tiff", height = 4, width=4, units="in", res=600)
#agb_line2
#dev.off()
```

Heatmap with smotting

```
devtools::source_gist('306e4b7e69c87b1826db')
```

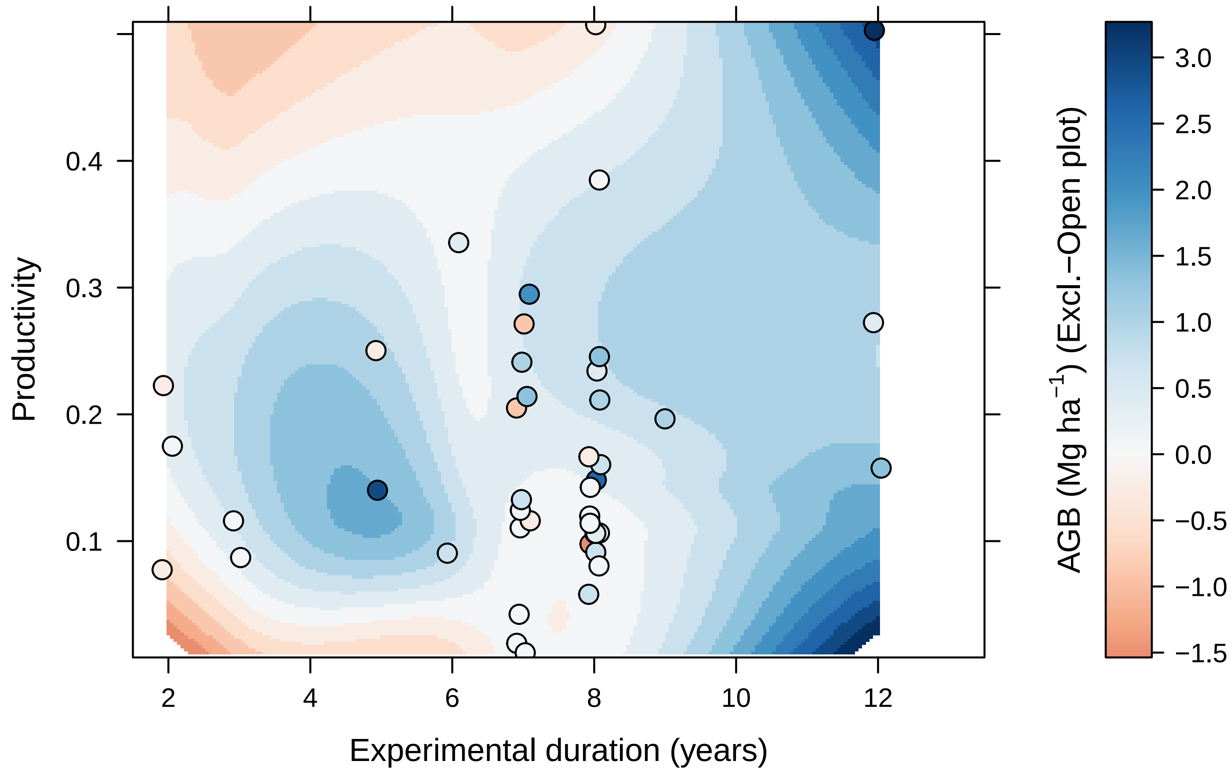
```
## Sourcing https://gist.githubusercontent.com/johnbaums/306e4b7e69c87b1826db/raw/cab2d177c92e8e766ee42
```

```
## SHA-1 hash of file is 4db2d0d95dff406e59eabdd25c41b6a20216847f
```

```
heat <-
levelplot(diff ~ YrsSinceExclosure * prod, datB,
          panel = panel.levelplot.points, cex = 1.2, col="black",
          jitter.x = TRUE,
          xlab="Experimental duration (years)",
          ylab="Productivity",
          col.regions=heat.colors(100, rev = T),
          ylab.right = expression(
              paste('AGB (Mg ha'^'-1', ') (Excl.-Open plot)')),
          par.settings =
```

```r
            list(layout.widths = list(axis.key.padding = 0,
                                      ylab.right = 2))
    ) +
    layer_(panel.2dsmoother(..., n = 200))
#tiff("/home/anders/Documents/lidar ms/AGBheatmap.tiff", height = 4, width=5, units="in", res=600)
diverge0(heat, ramp='RdBu')
```

```
## Loading required package: RColorBrewer

## Loading required package: rasterVis

## Loading required package: raster

## Loading required package: sp

##
## Attaching package: 'raster'

## The following object is masked from 'package:pryr':
##
##     subs

## The following object is masked from 'package:lme4':
##
##     getData

## The following objects are masked from 'package:MASS':
##
##     area, select

## The following object is masked from 'package:ggpubr':
##
##     rotate
```




```
#dev.off()
```

Modelling

```
dat2$AGB_s <- scale(dat2$AGB)[,1]

mod_agb <- glmmTMB(AGB ~ Treatment * prod + YrsSinceExclosure + YrsSinceExclosure:Treatment+ prod2 +  (
  data = dat2, REML=F, family = gaussian)
# Removing prod2 after first seing that it is not important. I had to do this because it overfitted the
mod_agb <- glmmTMB(AGB ~ Treatment * prod + YrsSinceExclosure + YrsSinceExclosure:Treatment +  (1|Locali
  data = dat2, REML=F, family = gaussian)

mod_agb_s <- glmmTMB(AGB_s ~ Treatment_c * prod_s + YrsSinceExclosure_s + YrsSinceExclosure_s:Treatment_
  data = dat2, REML=F,  family = gaussian)

summary(mod_agb)
```

```
##  Family: gaussian  ( identity )
## Formula:
## AGB ~ Treatment * prod + YrsSinceExclosure + YrsSinceExclosure:Treatment +
##     (1 | LocalityName)
## Data: dat2
##
```

```
##      AIC      BIC   logLik deviance df.resid
##    532.9    552.5   -258.4    516.9       78
##
## Random effects:
##
## Conditional model:
##  Groups       Name        Variance Std.Dev.
##  LocalityName (Intercept)  3.779   1.944
##  Residual                 20.380   4.514
## Number of obs: 86, groups:  LocalityName, 43
##
## Dispersion estimate for gaussian family (sigma^2): 20.4
##
## Conditional model:
##                                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)                           -0.9465     2.5868  -0.366  0.71446
## TreatmentExclosure                    -8.0245     3.3600  -2.388  0.01693 *
## prod                                  12.1550     7.0929   1.714  0.08659 .
## YrsSinceExclosure                      0.2439     0.3484   0.700  0.48387
## TreatmentExclosure:prod               17.6572     9.2129   1.917  0.05529 .
## TreatmentExclosure:YrsSinceExclosure   1.2423     0.4526   2.745  0.00605 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#dmod_agb <- dredge(mod_agb, beta="none", rank = "AICc")
#write_rds(dmod_agb, "../data/dmod_agb.RData")
dmod_agb <- read_rds("../data/dmod_agb.RData")
#dmod_agb_s <- dredge(mod_agb_s, beta="none", rank = "AICc")
#write_rds(dmod_agb_s, "../data/dmod_agb_s.RData")
dmod_agb_s <- read_rds("../data/dmod_agb_s.RData")
```

```
dmod_agb2 <- subset(dmod_agb, delta <2)
(dmod_agb_s2 <- subset(dmod_agb_s, delta <2))
```

```
## Global model call: glmmTMB(formula = AGB_s ~ Treatment_c * prod_s + YrsSinceExclosure_s +
##     YrsSinceExclosure_s:Treatment_c + prod2_s + (1 | LocalityName),
##     data = dat2, family = gaussian, REML = F, ziformula = ~0,
##     dispformula = ~1)
## ---
## Model selection table
##    cnd((Int)) dsp((Int)) cnd(prd_s) cnd(Trt_c) cnd(YSE_s) cnd(prd_s:Trt_c)
## 63  4.132e-07          +     0.3509     0.5957     0.2944           0.2952
## 47 -5.938e-08          +     0.3509     0.5957     0.2944
##    cnd(Trt_c:YSE_s) df  logLik  AICc delta weight
## 63           0.4228  8 -96.414 210.7  0.00  0.633
## 47           0.4989  7 -98.176 211.8  1.09  0.367
## Models ranked by AICc(x)
## Random terms (all models):
## 'cond(1 | LocalityName)'
```

Just two models

```r
importance(dmod_agb_s2)
```

```
##                      cond(prod_s) cond(Treatment_c) cond(YrsSinceExclosure_s)
## Sum of weights:      1.00         1.00              1.00
## N containing models:    2            2                 2
##                      cond(Treatment_c:YrsSinceExclosure_s)
## Sum of weights:      1.00
## N containing models:    2
##                      cond(prod_s:Treatment_c)
## Sum of weights:      0.63
## N containing models:    1
```

Lets export this as a table for the supplementary

```r
temp <- as.data.frame(dmod_agb2)
temp <- temp[,-2]
names(temp) <- c("Intercept",
                      "Productivity (P)",

                 "Herbivore Exclusion (HE)",
                      "Experimental duration (ED)",
                      "HE x P",
                      "HE x ED",
                      "df",
                      "log likelihood",
                      "AICc",
                      "delta AICc",
                      "weight"
                      )
temp <- temp[-3,]
#write.csv(temp, "../output/AGBModelSet_unstandardized.csv", row.names = F)
```

Average

Average across these three models

```r
AGBavg     <- model.avg(dmod_agb_s2, revised.var = TRUE, fit=F)
AGBavg_uns <- model.avg(dmod_agb2, revised.var = TRUE, fit=T) # used for predictions
summary(AGBavg_uns)
```

```
##
## Call:
## model.avg(object = get.models(object = dmod_agb2, subset = NA),
##     revised.var = TRUE)
##
## Component model call:
## glmmTMB(formula = AGB ~ <2 unique rhs>, data = dat2, family = gaussian,
##      ziformula = ~0, dispformula = ~1, REML = F)
##
## Component models:
##       df  logLik   AICc delta weight
```

```
## 12345  8 -258.44 534.75  0.00   0.63
## 1235   7 -260.20 535.84  1.09   0.37
##
## Term codes:
##                       cond(prod)                    cond(Treatment)
##                                1                                  2
##             cond(YrsSinceExclosure)            cond(prod:Treatment)
##                                3                                  4
## cond(Treatment:YrsSinceExclosure)
##                                5
##
## Model-averaged coefficients:
## (full average)
##                                              Estimate Std. Error Adjusted SE
## cond((Int))                                   -1.2177     2.6168      2.6571
## cond(prod)                                    15.3948     7.7864      7.8724
## cond(TreatmentExclosure)                      -7.4820     3.4517      3.5036
## cond(YrsSinceExclosure)                        0.2029     0.3529      0.3583
## cond(prod:TreatmentExclosure)                 11.1777    11.2320     11.3077
## cond(TreatmentExclosure:YrsSinceExclosure)     1.3243     0.4663      0.4732
##                                              z value Pr(>|z|)
## cond((Int))                                    0.458  0.64676
## cond(prod)                                     1.956  0.05052 .
## cond(TreatmentExclosure)                       2.136  0.03272 *
## cond(YrsSinceExclosure)                        0.566  0.57122
## cond(prod:TreatmentExclosure)                  0.989  0.32291
## cond(TreatmentExclosure:YrsSinceExclosure)     2.798  0.00514 **
##
## (conditional average)
##                                              Estimate Std. Error Adjusted SE
## cond((Int))                                   -1.2177     2.6168      2.6571
## cond(prod)                                    15.3948     7.7864      7.8724
## cond(TreatmentExclosure)                      -7.4820     3.4517      3.5036
## cond(YrsSinceExclosure)                        0.2029     0.3529      0.3583
## cond(prod:TreatmentExclosure)                 17.6572     9.2129      9.3581
## cond(TreatmentExclosure:YrsSinceExclosure)     1.3243     0.4663      0.4732
##                                              z value Pr(>|z|)
## cond((Int))                                    0.458  0.64676
## cond(prod)                                     1.956  0.05052 .
## cond(TreatmentExclosure)                       2.136  0.03272 *
## cond(YrsSinceExclosure)                        0.566  0.57122
## cond(prod:TreatmentExclosure)                  1.887  0.05918 .
## cond(TreatmentExclosure:YrsSinceExclosure)     2.798  0.00514 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The treatment effect increases with time

Lets add the standardized stuff to the canopy height figure

```
(figdat <- data.frame(summary(AGBavg)$coefmat.subset[-1,]))
```

```
##                                          Estimate Std..Error Adjusted.SE  z.value
## cond(prod_s)                            0.3508580 0.09018181  0.09159611 3.830491
```

```
## cond(Treatment_c)                    0.5956628 0.15027248  0.15262857 3.902696
## cond(YrsSinceExclosure_s)            0.2944422 0.09018181  0.09159611 3.214572
## cond(prod_s:Treatment_c)             0.2952405 0.15404547  0.15647276 1.886849
## cond(Treatment_c:YrsSinceExclosure_s) 0.4507557 0.15871807  0.16107511 2.798419
##                                       Pr...z..
## cond(prod_s)                         0.0001279
## cond(Treatment_c)                    0.0000951
## cond(YrsSinceExclosure_s)            0.0013064
## cond(prod_s:Treatment_c)             0.0591806
## cond(Treatment_c:YrsSinceExclosure_s) 0.0051353
```

get the weights and add them to the same table

```
AGBimpdf<-data.frame(importance(AGBavg))
AGBimpdf[c(1,2,3,5,4),]
```

```
## [1] 1.0000000 1.0000000 1.0000000 0.6330384 1.0000000
```

```
AGBimpdf <- as.numeric(AGBimpdf[,1])
figdat$importance.MA.ests.<-AGBimpdf[c(1,2,3,5,4)]
figdat
```

```
##                                       Estimate Std..Error Adjusted.SE  z.value
## cond(prod_s)                         0.3508580 0.09018181  0.09159611 3.830491
## cond(Treatment_c)                    0.5956628 0.15027248  0.15262857 3.902696
## cond(YrsSinceExclosure_s)            0.2944422 0.09018181  0.09159611 3.214572
## cond(prod_s:Treatment_c)             0.2952405 0.15404547  0.15647276 1.886849
## cond(Treatment_c:YrsSinceExclosure_s) 0.4507557 0.15871807  0.16107511 2.798419
##                                       Pr...z.. importance.MA.ests.
## cond(prod_s)                         0.0001279           1.0000000
## cond(Treatment_c)                    0.0000951           1.0000000
## cond(YrsSinceExclosure_s)            0.0013064           1.0000000
## cond(prod_s:Treatment_c)             0.0591806           0.6330384
## cond(Treatment_c:YrsSinceExclosure_s) 0.0051353           1.0000000
```

Then we also need the 95 CIs

```
cis <- confint(AGBavg)
cis <- cis[-1,]
cis <- as.data.frame(cis)
figdat$low <- cis[,1]
figdat$high <- cis[,2]
figdat
```

```
##                                       Estimate Std..Error Adjusted.SE  z.value
## cond(prod_s)                         0.3508580 0.09018181  0.09159611 3.830491
## cond(Treatment_c)                    0.5956628 0.15027248  0.15262857 3.902696
## cond(YrsSinceExclosure_s)            0.2944422 0.09018181  0.09159611 3.214572
## cond(prod_s:Treatment_c)             0.2952405 0.15404547  0.15647276 1.886849
## cond(Treatment_c:YrsSinceExclosure_s) 0.4507557 0.15871807  0.16107511 2.798419
##                                       Pr...z.. importance.MA.ests.        low
```

```
## cond(prod_s)                           0.0001279           1.0000000  0.1713330
## cond(Treatment_c)                      0.0000951           1.0000000  0.2965164
## cond(YrsSinceExclosure_s)              0.0013064           1.0000000  0.1149172
## cond(prod_s:Treatment_c)               0.0591806           0.6330384 -0.0114405
## cond(Treatment_c:YrsSinceExclosure_s)  0.0051353           1.0000000  0.1350542
##                                             high
## cond(prod_s)                           0.5303831
## cond(Treatment_c)                      0.8948093
## cond(YrsSinceExclosure_s)              0.4739673
## cond(prod_s:Treatment_c)               0.6019215
## cond(Treatment_c:YrsSinceExclosure_s)  0.7664571
```

Lets order them after effect size

```
figdat<-figdat[order(figdat$Estimate),]
```

Fix names

```
Row.names<-c('Experimental duration (ED)',
             'P x HE',
             'Productivity (P)',
             'HE x ED',
             'Herbivore Exclusion (HE)')
figdat <- cbind(Row.names, figdat)
figdat
```

```
##                                                          Row.names  Estimate
## cond(YrsSinceExclosure_s)              Experimental duration (ED) 0.2944422
## cond(prod_s:Treatment_c)                                   P x HE 0.2952405
## cond(prod_s)                                     Productivity (P) 0.3508580
## cond(Treatment_c:YrsSinceExclosure_s)                     HE x ED 0.4507557
## cond(Treatment_c)                        Herbivore Exclusion (HE) 0.5956628
##                                        Std..Error Adjusted.SE  z.value  Pr...z..
## cond(YrsSinceExclosure_s)              0.09018181  0.09159611 3.214572 0.0013064
## cond(prod_s:Treatment_c)               0.15404547  0.15647276 1.886849 0.0591806
## cond(prod_s)                           0.09018181  0.09159611 3.830491 0.0001279
## cond(Treatment_c:YrsSinceExclosure_s)  0.15871807  0.16107511 2.798419 0.0051353
## cond(Treatment_c)                      0.15027248  0.15262857 3.902696 0.0000951
##                                        importance.MA.ests.        low      high
## cond(YrsSinceExclosure_s)                        1.0000000  0.1149172 0.4739673
## cond(prod_s:Treatment_c)                         0.6330384 -0.0114405 0.6019215
## cond(prod_s)                                     1.0000000  0.1713330 0.5303831
## cond(Treatment_c:YrsSinceExclosure_s)            1.0000000  0.1350542 0.7664571
## cond(Treatment_c)                                1.0000000  0.2965164 0.8948093
```

Then lets make the figure First, adding empty row

```
# add to figdat4
figdat2 <- figdat[-(1:nrow(figdat)),]
figdat2[1,] <- c(NA, 100, NA, NA, NA, NA, NA, NA, NA)
figdat3 <- rbind(figdat, figdat2)
figdat3$Row.names <- as.character(figdat3$Row.names)
figdat3$Row.names[6] <- "AGB:"
```

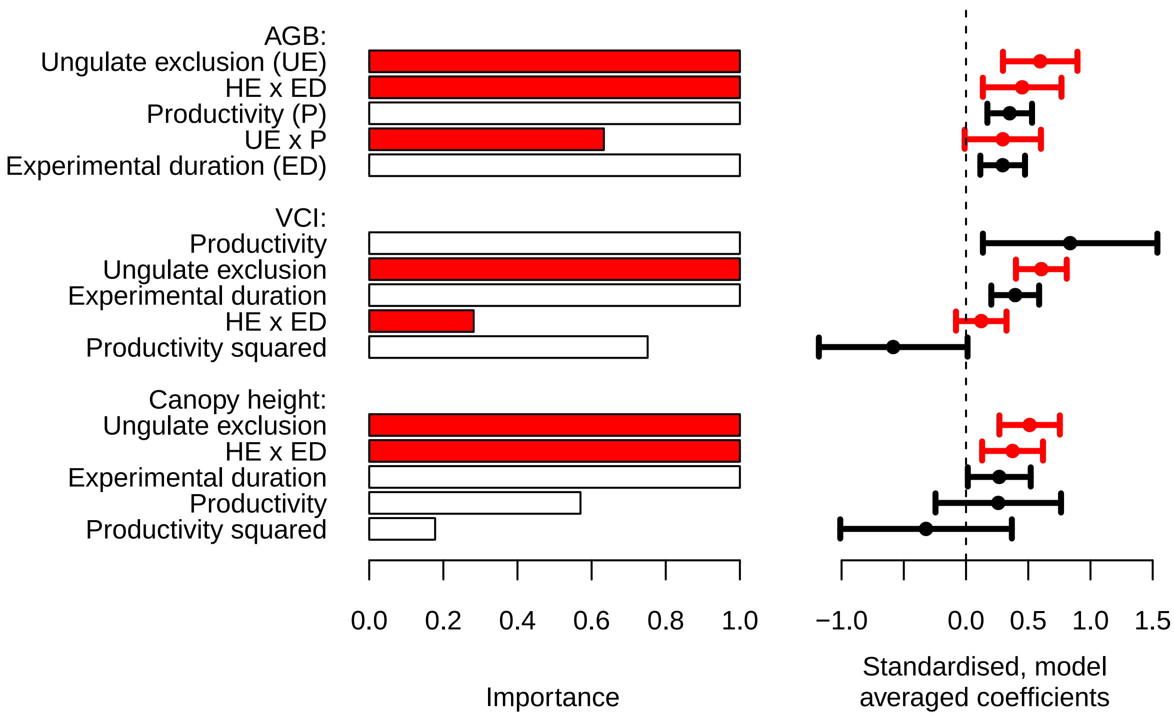
Combine:

```
figdat5 <- rbind(figdat4, figdat3)
```

```
rownames(figdat5) <-  1:19
figdat5 <- rbind(figdat5[1:13,], figdat5[7,], figdat5[14:19,])
figdat5$Row.names[figdat5$Row.names=="Herbivore Exclusion (HE)"] <-
  "Ungulate exclusion (UE)"
figdat5$Row.names[figdat5$Row.names=="P x HE"] <- "UE x P"
figdat5$Row.names[11] <- "Ungulate exclusion"
figdat5$Row.names[10] <- "Experimental duration"
figdat5$Row.names[5] <- "Ungulate exclusion"
figdat5$Row.names[3] <- "Experimental duration"
```

```
Avg %<a-% {

par(oma=c(1,10,1,1))
par(mfrow=c(1,2))
par(mar=c(5,0,1,1))
par(xpd=T)

barplot(figdat5$importance.MA.ests.,
        beside=T,horiz=T,
        names.arg=figdat5$Row.names,
        las=1,
        xlab='Importance',
        cex.axis=0.8,
        cex.names=0.8,
        cex.lab=0.8,
        col=c(0,0,0,2,2,
              0,0,
              0,2,0,2,0,
              0,0,
              0,2,0,2,2,
              0))
par(mar=c(5,1,1,1))
b1 <- barplot(figdat5[,2],
            horiz=T,
            col=F,
            border=F,
            xlim=c(-1.2,1.5),
            las=1,
            xlab='Standardised, model\naveraged coefficients',
            cex.axis=0.8,
            cex.lab=0.8)
points(figdat5[,2], #est
       b1,
       pch=16,
       col=c(1,1,1,2,2,
             1,1,
             1,2,1,2,1,
             1,1,
             1,2,1,2,2,
```

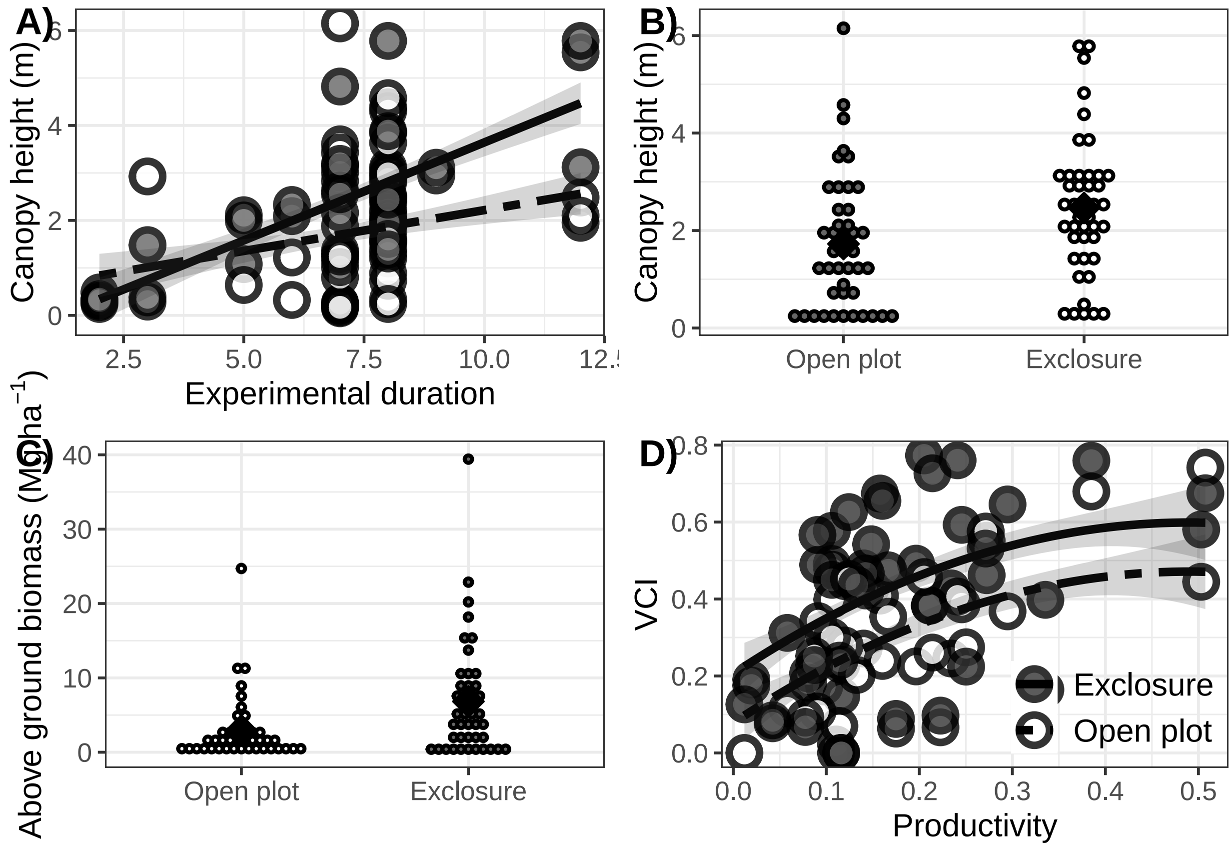
```
              1))
arrows(figdat5[,9],
       b1,
       figdat5[,8],
       b1,
       code=3,
       angle=90,
       length=0.05,
       lwd=3,
       col=c(1,1,1,2,2,
             1,1,
             1,2,1,2,1,
             1,1,
             1,2,1,2,2,
             1))
par(xpd=F)
abline(v=0,lty=2)
}
Avg
```




```
tiff("/home/anders/Documents/lidar ms/modAveragedEst.tiff",
     height = 8, width=8, units="in", res=600)
Avg
dev.off()
```

```r
(allPlots <- ggarrange(Canopy_line, ch_viol,AGBviol, vci_line,
                       labels = c("A)", "B)", "C)", "D)"),
                       ncol = 2, nrow = 2))
```

```
## `stat_bindot()` using `bins = 30`. Pick better value with `binwidth`.
```



4 plots

```r
tiff("/home/anders/Documents/lidar ms/Figures/4 fourPlots.tiff",
     height = 12, width=12, units="in", res=600)

allPlots

dev.off()
```