Plotting model results

Eva

2025-05-09

Case 1. Regular way, with effect or predict function

Example of model:

```
model1 <- glmmTMB(Abundance_sdl ~
                            scale(elev_30m_dem) +
                            scale(Forest_500m) +
                            treatment +
                            legacy *scale(sqrt(RegTime)) +
                            (1 | plot),
                          data = subset(mydata, legacy != "old-growth"),
                          ziformula = ~1,
                          family = nbinom1)
```

I obtain the effect of the forest according to the model, and store in “forest_data”

```
forest_data <- as.data.frame(effect("scale(Forest_500m)", model1,
                                    xlevels = 1000))
```

```
## Warning in Effect.glmmTMB(predictors, mod, vcov. = vcov., ...): overriding
## variance function for effects/dev.resids: computed variances may be incorrect
```

```
## Warning in Analyze.model(focal.predictors, mod, xlevels, default.levels, : the
## predictors scale(elev_30m_dem), scale(Forest_500m), scale(sqrt(RegTime)) are
## one-column matrices that were converted to vectors
```

Plot case 1

When plotting, I use the “real” data from the original database (“mydata”) for: - the point cloud in the plot - the lm lines of data that are not in the model (like Old-growth in my case)

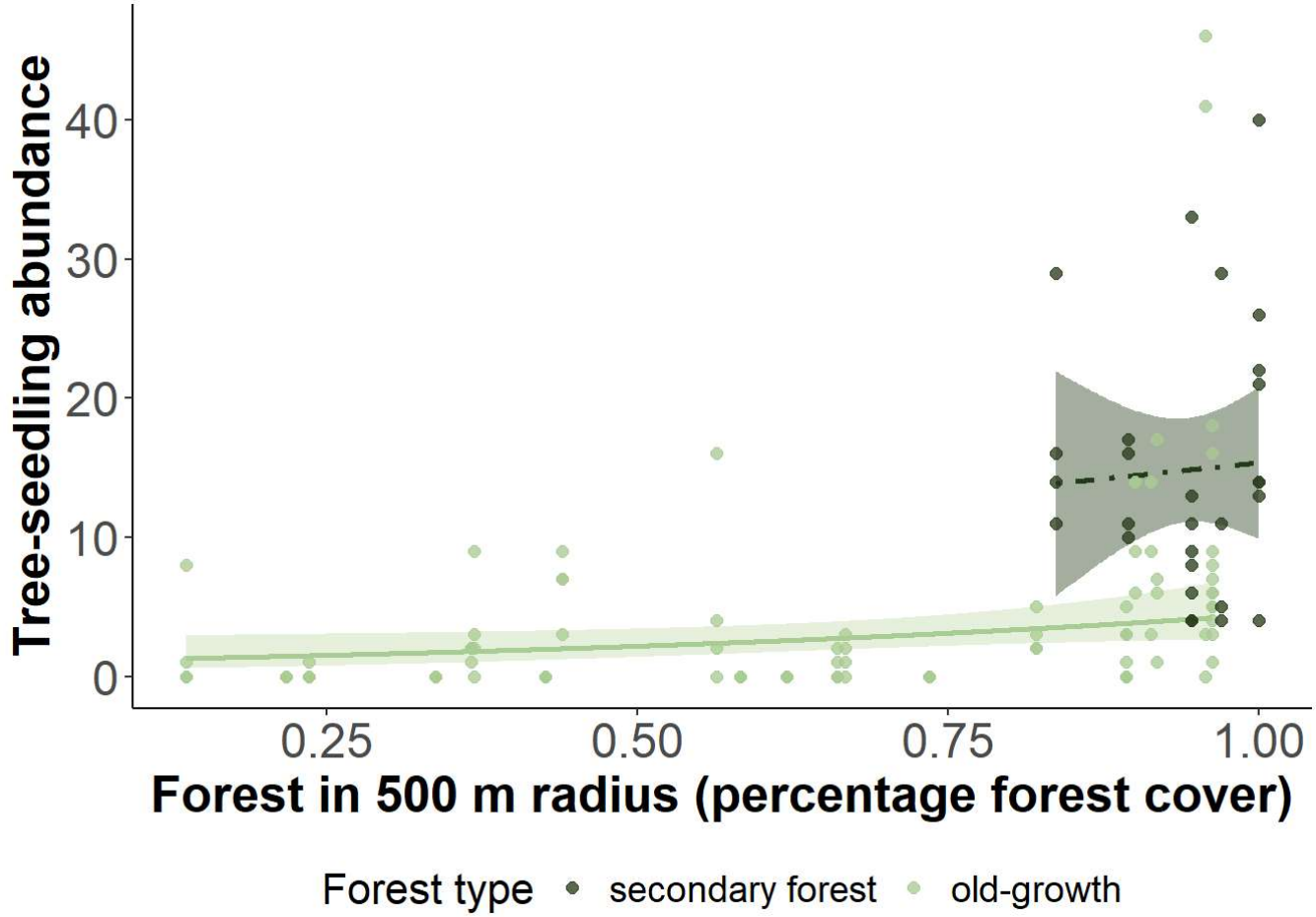
I use the data obtained with effect (“forest_data”) for the predicted line of the results

```
Ab_Forest <- ggplot(mydata) +
  aes(y=Abundance_sdl, x=Forest_500m) +
  labs(x="Forest in 500 m radius (percentage forest cover)",
       y="Tree-seedling abundance",
       color = "Forest status") +
  scale_colour_manual(values=c("#273c1a", "#acce97"),
                      labels=c("secondary forest", "old-growth"),
                      name="Forest type") +
  geom_ribbon(data=forest_data,
              aes(y=fit,ymin=lower, ymax=upper),
              alpha=0.3,linetype=0, fill ="#acce97")  +
  geom_line(data=forest_data, aes(y=fit),
            linetype=1, color = "#acce97", size = 1, alpha = 1) +
  geom_smooth(data = subset(mydata, forest.type == "old-growth"),
              formula = y ~ x, method='lm', linetype=4,
              fill ="#273c1a") +
  geom_line(data=subset(mydata, forest.type == "old-growth"),
            stat = "smooth",formula = y ~ x, method = 'lm',
            linetype=4, color = "#273c1a", size = 1, alpha = 1) +
  geom_point(aes(color=forest.type), alpha=0.75, width=0.3, size=2) +
  theme_classic() +
  theme(
    axis.text=element_text(size=18),
    axis.title=element_text(size=19,face="bold"),
    legend.position = "bottom",
    legend.text = element_text(size = 14),
    legend.title = element_text(size = 16)
  )
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## Warning in geom_point(aes(color = forest.type), alpha = 0.75, width = 0.3, :
## Ignoring unknown parameters: `width`
```

```
Ab_Forest
```



Case 2. Rescaling the effect of the predictor.

Useful in some cases (when the predictor was scaled and centered before introducing it in the model)

Example of the model:

```
model2 <- glmmTMB(ab_sd ~ Elevation_m_c +
                    clim_bio12_c +
                    BA.trees_above5cm_c +
                    Tree.species.richness_c*F_dist_c +
                    Forest_c*F_dist_c,
                  ziformula = ~1,
                  family = nbinom2,
                  data = bp1)
```

Plot case 2

I obtain the effect of the elevation according to the model, and store in “elevation_effect”

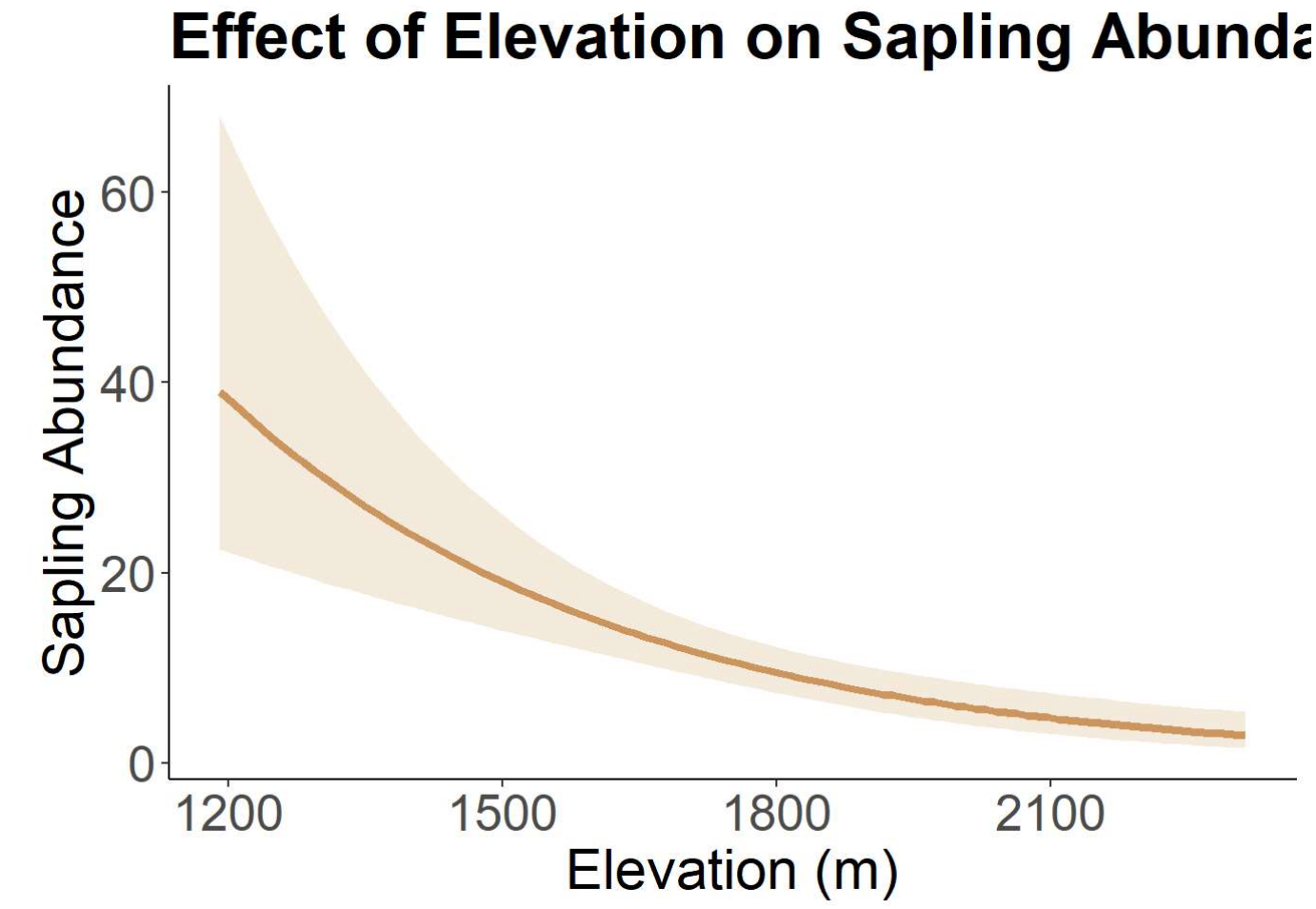
Same as before, when plotting, I use the “real” data from the original database (“original”) for: - the point cloud in the plot - the lm lines of data that are not in the model (like Old-growth in my case)

I use the data obtained with effect (“elevation_effect”) for the predicted line of the results. In this case, I need to transform the predicted data obtained with the function effect, since it was scaled and center, to be able to plot the x-axis in the same scale values as the original data

```
# Obtaining the marginal effect of Elevation_m_c
elevation_effect <- as.data.frame(effect("Elevation_m_c", model2, xlevels = 100))
```

```
## Warning in Effect.glmmTMB(predictors, mod, vcov. = vcov., ...): overriding
## variance function for effects/dev.resids: computed variances may be incorrect
```

```
# Plotting the effect

#I reescale the elevation effect in the first line of the code

ggplot(elevation_effect, aes(x = Elevation_m_c * attr(scale(bp1$Elevation_m), "scaled:scale")
+ attr(scale(bp1$Elevation_m), "scaled:center"),
                             y = fit)) +
  geom_line(color = "#cd9963", size = 1.5) +
  geom_ribbon(aes(ymin = lower, ymax = upper), alpha = 0.2, fill = "#cd9963") +
  labs(x = "Elevation (m)", y = "Sapling Abundance",
       title = "Effect of Elevation on Sapling Abundance") +
  theme_classic() +
  theme(
    axis.text = element_text(size = 20),
    axis.title = element_text(size = 22),
    plot.title = element_text(size = 25, face = "bold")
  )
```



Case 3 Obtaining the effect manually

Some model functions or families don’t work well with predict or effects. At least that is what I found sometimes with my data.

In these cases I need to obtain the database manually (newdata_forest) for then being able to use predict This database contains the predictor I am interested in with the same values as the original database. The rest of the predictors used in the model, but not like in the original database, but with their mean value, or, in case they are categorical, with one of the values. This is the same as the predict or effect functions do, but we do it manually. We control for all the predictors we are not interested to see, for then predicting the effect of our predictor according to our model.

```
# Create a dataframe with the values of Forest_500m
newdata_forest <- data.frame(Forest_500m = mydata2$Forest_500m)

# If there are other predictors in the model that need to be controled for, add their mean
newdata_forest$elev_30m_dem <- mean(mydata2$elev_30m_dem, na.rm = TRUE)
newdata_forest$treatment <- "FU"  # adjust when needed
newdata_forest$legacy <- "cacao"
newdata_forest$RegTime<-mean(mydata2$RegTime, na.rm = TRUE)
newdata_forest$plot <- factor("CA63",
                              levels = levels(mydata2$plot))
```

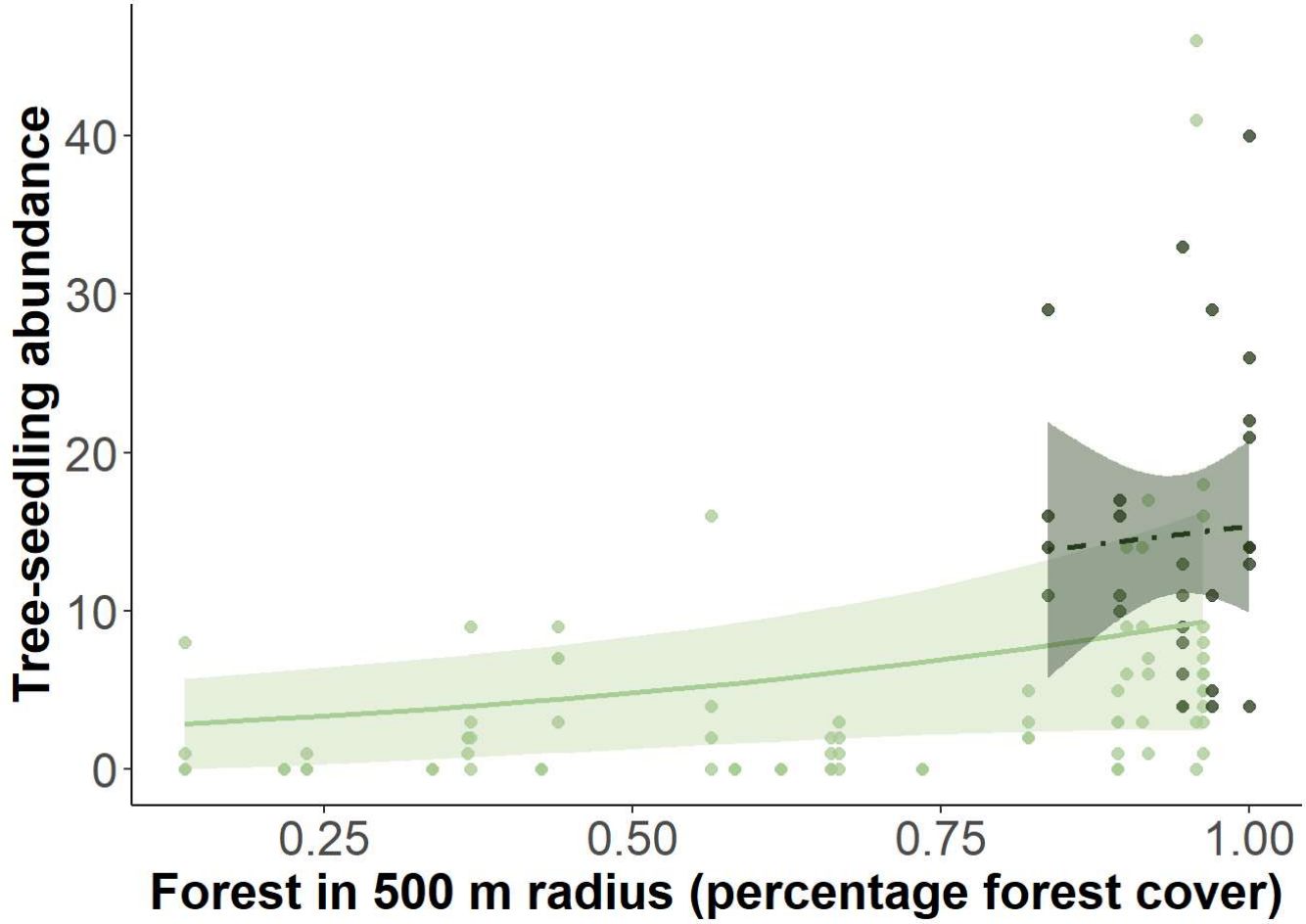
We then obtain the predictions for the predictor, with the new database.

```
# Making predictions with the model
forest_predsAb <- predict(model1,
                          newdata = newdata_forest,
                          type = "response",
                          se.fit = TRUE,
                          re.form = NULL)

# Adding predictions and their confidence intervals
mydata2$predictedAb <- forest_predsAb$fit
mydata2$conf.lowAb <- forest_predsAb$fit - 1.96 * forest_predsAb$se.fit
mydata2$conf.highAb <- forest_predsAb$fit + 1.96 * forest_predsAb$se.fit
```

Plot case 3

```
Ab_Forest <- ggplot() +
  # Points are the original data
  geom_point(data = mydata2, aes(x = Forest_500m, y = Abundance_sdl),
             color = "#acce97", alpha = 0.75, size = 2) +

  # Original points for Old-Growth
  geom_point(data = subset(mydata, forest.type == "old-growth"),
             aes(x = Forest_500m, y = Abundance_sdl),
             color = "#273c1a", alpha = 0.75, size = 2) +

  # Predictions for Secondary Forest
  geom_ribbon(data = mydata2,
              aes(x = Forest_500m, y = predictedAb, ymin = conf.lowAb, ymax = conf.highAb),
              alpha = 0.3, fill = "#acce97") +
  geom_line(data=mydata2, aes(x = Forest_500m, y = predictedAb),
            color = "#acce97", size = 1) +
  # Linear regression for Old-Growth, that is not in the model
  geom_smooth(data = subset(mydata, forest.type == "old-growth"),
              aes(x = Forest_500m, y = Abundance_sdl),
              method = "lm", color = "#273c1a", linetype = 4, fill = "#273c1a") +
    labs(x = "Forest in 500 m radius (percentage forest cover)",
       y = "Tree-seedling abundance",
       color = "Forest status") +
  scale_colour_manual(values = c("#acce97", "#273c1a"),
                      labels = c("secondary forest", "old-growth"),
                      name = "Forest type") +
  theme_classic() +
  theme(axis.text = element_text(size = 18),
        axis.title = element_text(size = 19, face = "bold"),
        legend.position = "bottom",
        legend.text = element_text(size = 14),
        legend.title = element_text(size = 16))

Ab_Forest
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Case 4 Boxplots

Learning to plot effects of categorical variables was hard for me! I put here how I managed.

I first create the new database manually, like in case 3. Controlling for all predictors except the one I am interested in, that I take from the original database.

```
newdata_treatment <- data.frame(treatment = mydata2$treatment)# Use directly the rows of your original database
newdata_treatment$elev_30m_dem <- mean(mydata2$elev_30m_dem)
newdata_treatment$Forest_500m <- mean(mydata2$Forest_500m)
newdata_treatment$legacy <- "cacao"
newdata_treatment$RegTime <- mean(mydata2$RegTime)
newdata_treatment$plot <- factor("CA63",
                                 levels = levels(mydata2$plot))
```

I obtain with this data the predictions for the response variable with the function simulate.

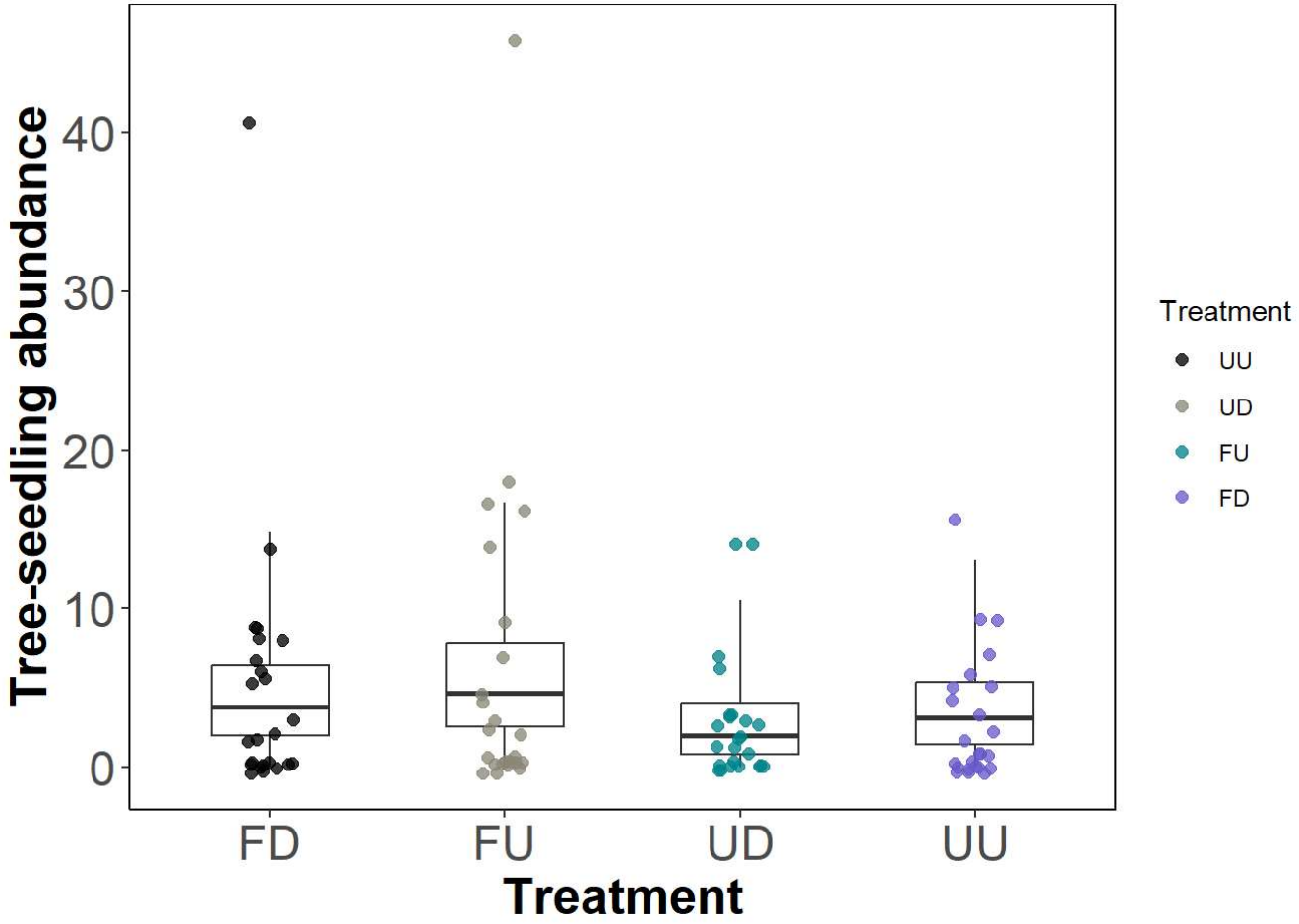
```
sims <- simulate(model1, nsim = 1000, newdata = newdata_treatment)
```

Since I want to do boxplots I need to obtain the top and bottom values of the box, as well as the median. Top value of the box is the third quartile. Bottom is the first quartile. The lower and upper limits, to create the whiskers, are obtained in pred_min and pred_max

```
pred_df <- newdata_treatment %>%
  mutate(
    pred_median = apply(sims, 1, median),
    pred_Q1 = apply(sims, 1, function(x) quantile(x, 0.25)),
    pred_Q3 = apply(sims, 1, function(x) quantile(x, 0.75)),
    pred_min = apply(sims, 1, function(x) quantile(x, 0.025)),
    pred_max = apply(sims, 1, function(x) quantile(x, 0.975))
  )


pred_df_summarized <- pred_df %>%
  group_by(treatment) %>%
  summarise(
    pred_median = mean(pred_median, na.rm = TRUE),
    pred_Q1 = mean(pred_Q1, na.rm = TRUE),
    pred_Q3 = mean(pred_Q3, na.rm = TRUE),
    pred_min = mean(pred_min, na.rm = TRUE),
    pred_max = mean(pred_max, na.rm = TRUE)
  )
```

Plot Case 4

```
treatment_colors <- c("UU" = "#6A5ACD", "UD" = "#00868B", "FU" = "#8B8878", "FD" = "black")

Ab_treat <- ggplot(pred_df_summarized, aes(x = factor(treatment), y = pred_median, group = tr
eatment))+
  labs(x="Treatment", y = "Tree-seedling abundance")+
  geom_boxplot(aes(ymin = pred_min,
                   ymax = pred_max,
                   lower = pred_Q1,
                   upper = pred_Q3,
                   middle = pred_median),
               stat = "identity", width = 0.5) +
  geom_jitter(data = mydata2, aes(x = treatment, y = Abundance_sdl, color = treatment),
              size=2, alpha=0.75, width = 0.1) +
  scale_color_manual(values = treatment_colors,
                     labels= c("UU", "UD", "FU", "FD"),
                     name="Treatment")+
  theme_bw() +
  theme(
    panel.border = element_rect(color = "black", fill = NA, linewidth = 0.5),
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    axis.title = element_text(size = 19, face = "bold"),
    axis.text = element_text(size = 18),
    axis.line = element_line(color = "black"),
    axis.line.y = element_line(),
    axis.line.x = element_line(),
    )

Ab_treat
```



Case 5 Interaction terms with predict/effect function

This is a little more complex. Both predictors interacting are numeric, so for 1 of them I divided it in three categories.

I first obtain the predicted values of the predictors according to our model, but creating these categories for one of the predictors (F_dist (forest distance)).

Example model

```
model3 <- glmmTMB(ab_sd ~ Elevation_m_c +
                    clim_bio12_c +
                    BA.trees_above5cm_c +
                    Tree.species.richness_c*F_dist_c +
                    Forest_c*F_dist_c,
                  ziformula = ~1,
                  family = nbinom2,
                  data = bp1)
```

```
# Obtain the effects of interaction between Tree.species.richness_c and F_dist_c
interaction_effect <- as.data.frame(effect("Tree.species.richness_c:F_dist_c", model3,
                                           xlevels = list(Tree.species.richness_c =
                                                            seq(min(bp1$Tree.species.richness
_c, na.rm = TRUE),                                                  max(bp1$Tree.species.rich
ness_c, na.rm = TRUE),
             length.out = 50),
F_dist_c = quantile(bp1$F_dist_c, probs = c(0.25, 0.5, 0.75), na.rm = TRUE))))
```

```
## Warning in Effect.glmmTMB(predictors, mod, vcov. = vcov., ...): overriding
## variance function for effects/dev.resids: computed variances may be incorrect
```

since they are scaled and centered, I transform the values obtained so they are in the same scale as the original data.

```
interaction_effect$Tree.species.richness <- (interaction_effect$Tree.species.richness_c * sd
(bp1$rich_ad, na.rm = TRUE)) + mean(bp1$rich_ad, na.rm = TRUE)
interaction_effect$F_dist <- (interaction_effect$F_dist_c * sd(bp1$F_dist, na.rm = TRUE)) + m
ean(bp1$F_dist, na.rm = TRUE)
```

I check the values for the three quantiles of Forest distance (F_dist) to write these values in the plot (for more clarity). I also define the colour scale.

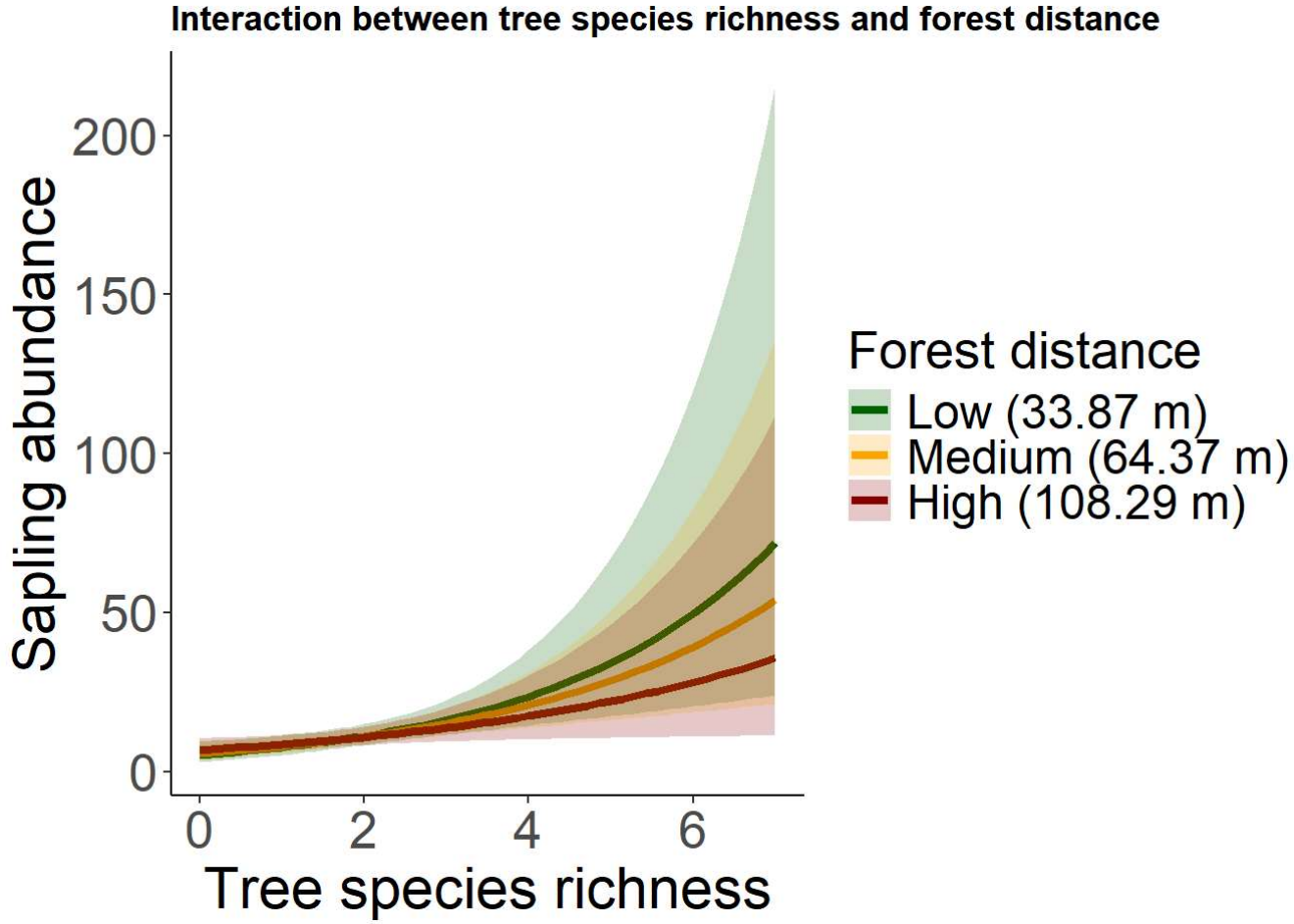
```
quantiles_F_dist <- quantile(bp1$F_dist, probs = c(0.25, 0.5, 0.75), na.rm = TRUE)
quantiles_F_dist
```

```
##       25%       50%       75%
##  33.87438  64.37406 108.28564
```

```
# Convert F_dist to factor with descriptive tags
interaction_effect$F_dist_factor <- factor(interaction_effect$F_dist,
                                           levels = unique(interaction_effect$F_dist),
                                           labels = c("Low (33.87 m)", "Medium (64.37 m)", "H
igh (108.29 m)"))

# Define the colours
colors <- c("Low (33.87 m)" = "darkgreen", "Medium (64.37 m)" = "orange", "High (108.29 m)" =
"darkred")
```

Plotting case 5

```
# Graficar la interacción
ggplot(interaction_effect, aes(x = Tree.species.richness, y = fit, color = F_dist_factor)) +
  geom_line(size = 1.5) +  # Línea más gruesa
  geom_ribbon(aes(ymin = lower, ymax = upper, fill = F_dist_factor), alpha = 0.2,  color = N
A) +
  scale_color_manual(values = colors, name = "Forest distance") +  # Asignar colores
  scale_fill_manual(values = colors, name = "Forest distance") +  # Para las bandas de confia
nza
  labs(x = "Tree species richness",
       y = "Sapling abundance",
       title = "Interaction between tree species richness and forest distance") +
  theme_classic() +
  theme(
    axis.text = element_text(size = 20),
    axis.title = element_text(size = 22),
    plot.title = element_text(face = "bold"),
    legend.text = element_text(size = 18),
    legend.title = element_text(size = 20)
  )
```



Case 6 Interaction terms manually

Example model

```
model4 <- glmmTMB(expH_sdl ~
                    scale(elev_30m_dem) +
                    scale(Forest_500m) +
                    treatment +
                    legacy*scale(sqrt(RegTime))+
                    (1 | plot),
                  data = subset(mydata, legacy != "old-growth") ,
                  family = tweedie())
```

I create the new database manually in newdata_interactionDv

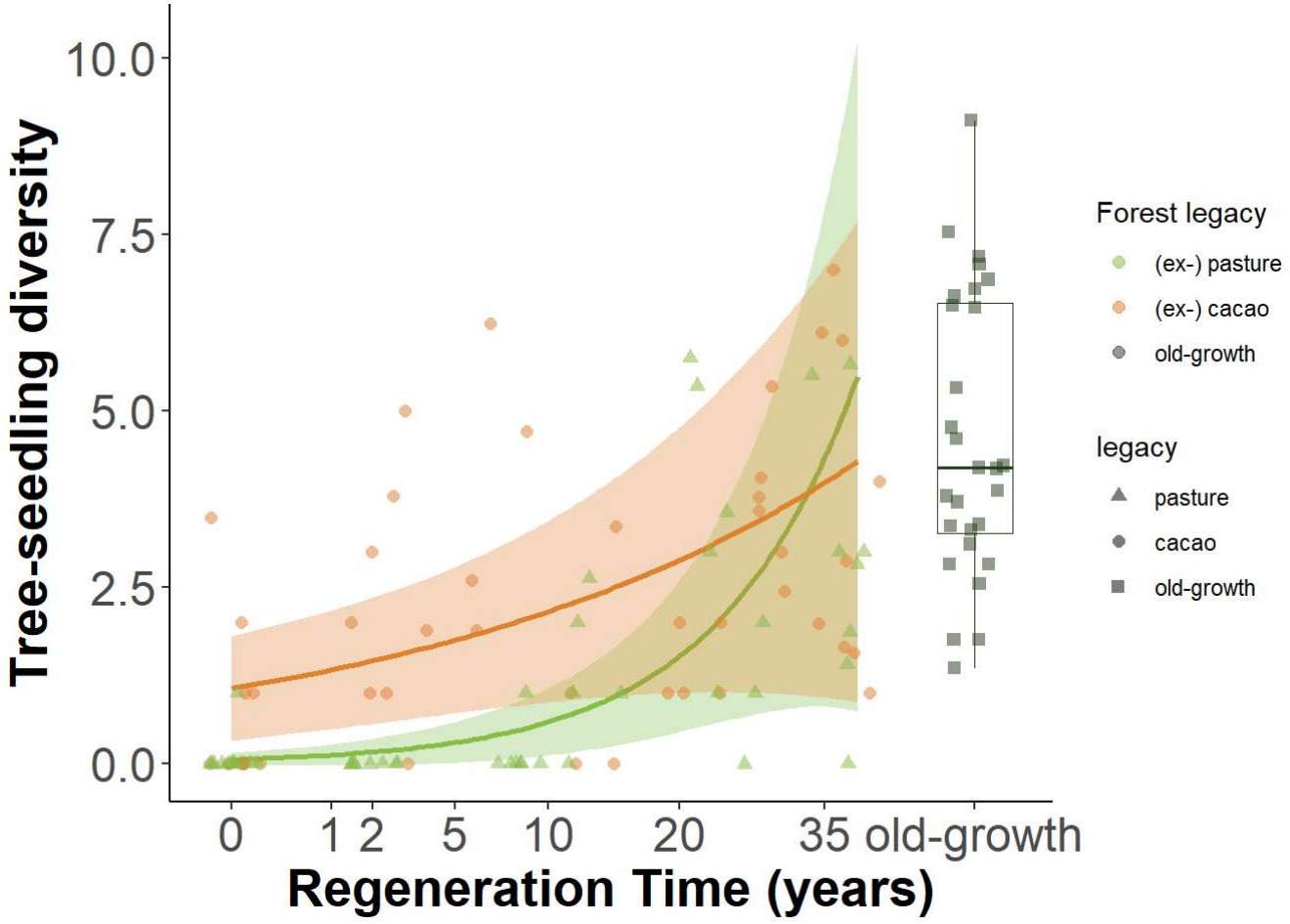
```
newdata_interactionDv <- expand.grid(
  legacy = levels(mydata2$legacy)[levels(mydata2$legacy) != "old-growth"],  # all legacy leve
ls
  RegTime = seq(from = min(mydata2$RegTime), to = max(mydata2$RegTime), length.out = 100),  #
Values for RegTime
  elev_30m_dem = mean(mydata2$elev_30m_dem),  # Mantain other variables constant
  treatment = factor("FU", levels = levels(mydata2$treatment)),
  Forest_500m = mean(mydata2$Forest_500m),
  plot = factor("CA63", levels = levels(mydata2$plot))
)
```

Obtaining the predicted values

```
predictions_interactionDv <- predict(model4,
                                     newdata = newdata_interactionDv,
                                     type = "response",
                                     se.fit = TRUE)


# Add predictions and their confidence intervals
newdata_interactionDv$predictedDv <- predictions_interactionDv$fit
newdata_interactionDv$conf.lowDv <- predictions_interactionDv$fit - 1.96 * predictions_intera
ctionDv$se.fit
newdata_interactionDv$conf.highDv <- predictions_interactionDv$fit + 1.96 * predictions_inter
actionDv$se.fit
```

Plotting case 6

```
Dv_time_legacy <- ggplot(mydata) +
  aes(y= expH_sdl, x=RegTime)+
  scale_colour_manual(values=c("#87bf4a","#e58331",  "#273c1a"), labels=c( "(ex-) pasture","
(ex-) cacao", "old-growth"))+
  labs(x="Regeneration Time (years)", y = "Tree-seedling diversity")+
  geom_boxplot(data = subset(mydata, legacy=="old-growth"), fill = NA, size=0.3, color = "#27
3c1a", outlier.alpha = 0)+
  geom_ribbon(data=subset(newdata_interactionDv, legacy=="pasture"), aes(y=predictedDv,ymin=c
onf.lowDv, ymax=conf.highDv),
              alpha=0.3,linetype=0, fill ="#87bf4a")  +
  geom_line(data=subset(newdata_interactionDv, legacy=="pasture"), aes(y=predictedDv),
            linetype=1, color = "#87bf4a", size = 1, alpha = 1) +
  geom_ribbon(data=subset(newdata_interactionDv, legacy=="cacao"), aes(y=predictedDv,ymin=con
f.lowDv, ymax=conf.highDv),
              alpha=0.3,linetype=0, fill ="#e58331")  +
  geom_line(data=subset(newdata_interactionDv, legacy=="cacao"), aes(y=predictedDv),
            linetype=1, color = "#e58331", size = 1, alpha = 1) +
  geom_jitter(aes(color= legacy, shape= legacy), alpha=0.5, width=0.3, size=2)+
  scale_shape_manual(values=c(17,16,15))+
  scale_x_sqrt(breaks = c(0,1,2,5,10,20,35,55), label = c(0,1,2,5,10,20,35,"old-growth"))+
  theme_classic() +
  labs(color = "Forest legacy") +
  theme(
    axis.text=element_text(size=18),
    axis.title=element_text(size=20,face="bold")
  )

Dv_time_legacy
```


Tree-seedling diversity
10.0
7.5
5.0
2.5
0.0
0
1
2
5
10
20
35
old-growth
Regeneration Time (years)
Forest legacy
(ex-) pasture
(ex-) cacao
old-growth
legacy
pasture
cacao
old-growth