

Scripps's Murrelet Egg Size Prediction

Amelia J. DuVall & Marcela Todd

7/23/2021

Introduction

This document details steps taken to predict Scripps's Murrelet (*Synthliboramphus scrippsi*) egg size at Santa Barbara Island within Channel Islands National Park using top model results.

This is v.2021-07-24

Prediction

Model prediction for the 2 top models

Run the top models and create a table of AIC scores.

```
topmod1 <- lmer(Size ~ EggOrder + BEUTI + NPGO + (1 | Plot), data = SCMUdf, REML = TRUE)
topmod2 <- lmer(Size ~ EggOrder + NPGO + ONI + (1 | Plot), data = SCMUdf, REML = TRUE)
```

```
# AIC tab for 2 top models
```

```
AIC.tab2 <- matrix(NA, nrow = 2, ncol = 3)
AIC.tab2[1,1] <- AIC(topmod1)
AIC.tab2[2,1] <- AIC(topmod2)
AIC.tab2[,2] <- AIC.tab2[,1] - min(AIC.tab2[,1])
AIC.tab2[,3] <- exp(-0.5*AIC.tab2[,2])/(sum(exp(-0.5*AIC.tab2[,2])))
print(AIC.tab2)
```

```
##           [,1]      [,2]      [,3]
## [1,] 10558.27 0.000000 0.6651142
## [2,] 10559.64 1.372339 0.3348858
```

We want predictions with all predictors but one held constant, and then we'll do this one-at-a-time for each of the predictors. This will allow us to produce plots looking at the marginal effect of each predictor. Set-up a dataset with everything at its mean value (except Egg Order).

```
set.seed(1)
SCMU.null <- SCMUdf[c(1:200),]
SCMU.null$EggOrder <- c(rep("Egg1",100),rep("Egg2",100)) # the first 100 eggs with be Egg 1; the second
SCMU.null$BEUTI <- mean(SCMUdf$BEUTI)
SCMU.null$NPGO <- mean(SCMUdf$NPGO)
SCMU.null$ONI <- mean(SCMUdf$ONI)
SCMU.null$ANCHL <- mean(SCMUdf$ANCHL)
SCMU.null$PDO <- mean(SCMUdf$PDO)
SCMU.null$SST <- mean(SCMUdf$SST)
SCMU.null$Plot <- sample(SCMUdf$Plot,200,replace = TRUE)
```

```
## look at min/max covariate values
```

```
summary(SCMUdf$BEUTI)
```

```
##      Min.   1st Qu.   Median     Mean   3rd Qu.     Max.
## -1.76150 -0.48909  0.02571 -0.12028  0.39502  1.51698
```

```
summary(SCMUdf$NPGO)
```

```
##      Min.   1st Qu.   Median     Mean   3rd Qu.     Max.
## -1.61193 -0.76135  0.52847  0.02653  1.10971  1.39094
```

```
summary(SCMUdf$ONI)
```

```
##      Min.   1st Qu.   Median     Mean   3rd Qu.     Max.
## -1.34238 -0.49482 -0.35178  0.07779  0.78612  1.72334
```

Model prediction for 2 top models

BEUTI

Vary just one predictor at a time for prediction. For example, create the dataset SCMU. BEUTI that will be used for predictions where the only thing that values is BEUTI.

```
SCMU.BEUTI <- SCMU.null
```

```
SCMU.BEUTI$BEUTI <- rep(seq(from = -2, to = 2, length.out = 100),2)
```

```
# SCMU.BEUTI$BEUTI <- rep(seq(from = min(SCMUdf$BEUTI),to = max(SCMUdf$BEUTI),length.out = 100),2)
```

```
#PREDICT FOR SCMU.BEUTI
```

```
plot.sd.1 <- as.data.frame(VarCorr(topmod1))$sdcor[1]
```

```
resid.sd.1 <- as.data.frame(VarCorr(topmod1))$sdcor[2]
```

```
plot.sd.2 <- as.data.frame(VarCorr(topmod2))$sdcor[1]
```

```
resid.sd.2 <- as.data.frame(VarCorr(topmod2))$sdcor[2]
```

```
sims <- 10000 # run 100 to check, otherwise update to 10,000
```

```
## model prediction for 2 top models
```

```
pv.BEUTI <- matrix(nrow = sims, ncol = nrow(SCMU.BEUTI))
```

```
for(i in 1:sims){
```

```
  #choose a model
```

```
  model <- which(rmultinom(n = 1, size = 1,prob = c(AIC.tab2[,3]))==1)
```

```
  if(model == 1){
```

```
    #we simulate conditioning on no specific random effects levels
```

```
    y <- unlist(simulate(topmod1))
```

```
    bmod <- refit(topmod1,y)
```

```
    pv.BEUTI[i,] <- predict(bmod, re.form = ~0, newdata = SCMU.BEUTI) + rnorm(1,0,sd=plot.sd.1) + rnorm
```

```
  }
```

```
  if(model == 2){
```

```
    #we simulate conditioning on no specific random effects levels
```

```
    y <- unlist(simulate(topmod2))
```

```
    bmod <- refit(topmod2,y)
```

```
    pv.BEUTI[i,] <- predict(bmod, re.form = ~0, newdata = SCMU.BEUTI) + rnorm(1,0,sd=plot.sd.2) + rnorm
```

```
  }
```

```
}
```

```
## boundary (singular) fit: see ?isSingular
```

```
## boundary (singular) fit: see ?isSingular
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

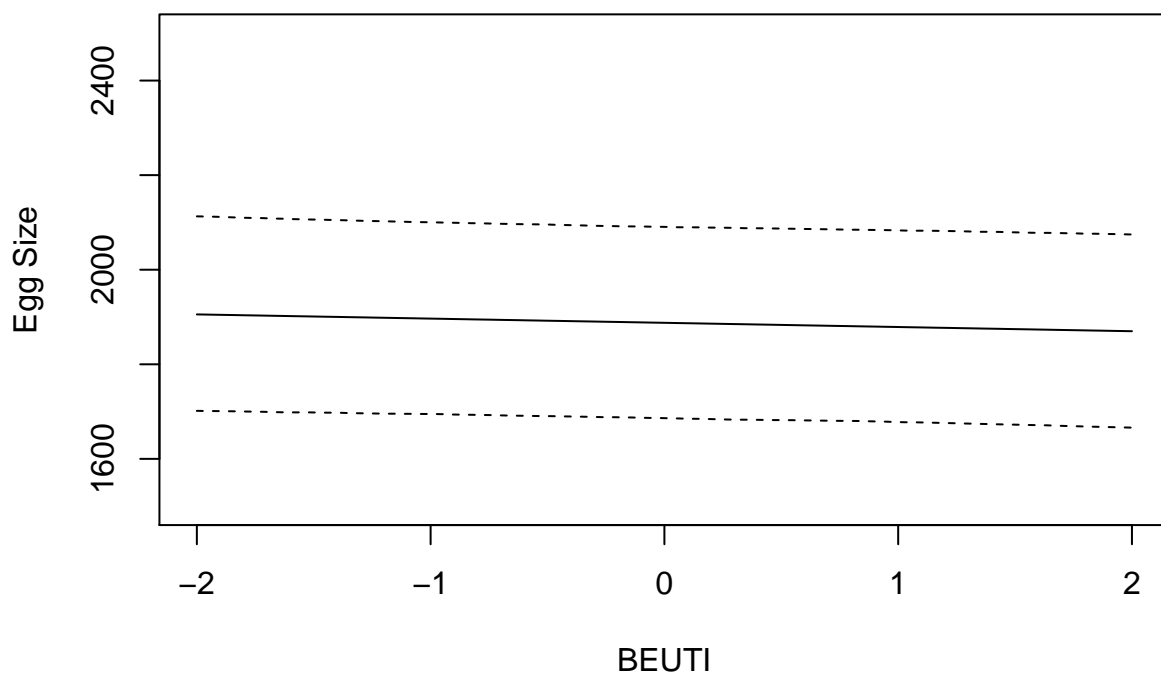
[illegible]

[illegible]

[illegible]

```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular

## check plot
plot(SCMU.BEUTI$BEUTI[1:100],y = apply(pv.BEUTI,2,mean)[1:100],xlab = "BEUTI",ylab = "Egg Size",type = "l",lty=2)
lines(SCMU.BEUTI$BEUTI[1:100],y = apply(pv.BEUTI,2,function(x)quantile(x,probs= 0.025))[1:100],lty=2)
lines(SCMU.BEUTI$BEUTI[1:100],y = apply(pv.BEUTI,2,function(x)quantile(x,probs= 0.975))[1:100],lty=2)
```



NPGO

```
SCMU.NPGO <- SCMU.null
SCMU.NPGO$NPGO <- rep(seq(from = -2, to = 2, length.out = 100),2)
# SCMU.NPGO$NPGO <- rep(seq(from = min(SCMUdf$NPGO),to = max(SCMUdf$NPGO),length.out = 100),2)

plot.sd.1 <- as.data.frame(VarCorr(topmod1))$sdcor[1]
resid.sd.1 <- as.data.frame(VarCorr(topmod1))$sdcor[2]

plot.sd.2 <- as.data.frame(VarCorr(topmod2))$sdcor[1]
resid.sd.2 <- as.data.frame(VarCorr(topmod2))$sdcor[2]

sims <- 10000 # run 100 to check, otherwise update to 10,000
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

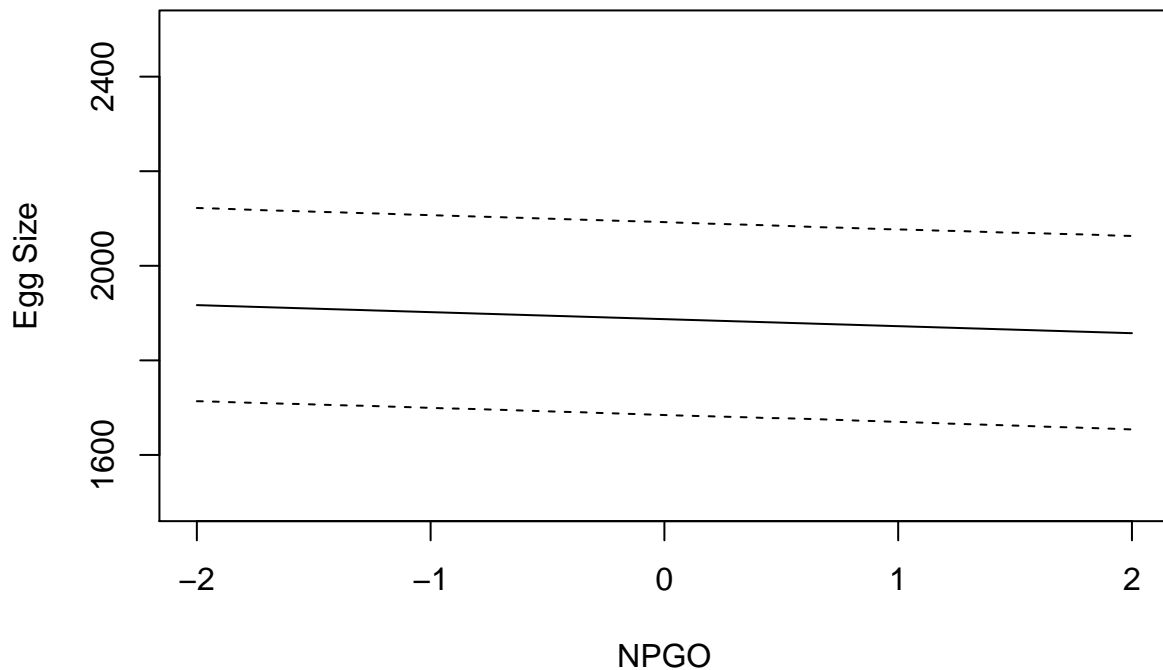
[illegible]

[illegible]

[illegible]


```
## check plots
```

```
plot(SCMU.NPGO$NPGO[1:100], y = apply(pv.NPGO, 2, mean)[1:100], xlab = "NPGO", ylab = "Egg Size", type = "l",
lines(SCMU.NPGO$NPGO[1:100], y = apply(pv.NPGO, 2, function(x) quantile(x, probs = 0.025))[1:100], lty = 2)
lines(SCMU.NPGO$NPGO[1:100], y = apply(pv.NPGO, 2, function(x) quantile(x, probs = 0.975))[1:100], lty = 2)
```



ONI

```
SCMU.ONI <- SCMU.null
SCMU.ONI$ONI <- rep(seq(from = -2, to = 2, length.out = 100),2)
# SCMU.ONI$ONI <- rep(seq(from = min(SCMUdf$ONI),to = max(SCMUdf$ONI),length.out = 100),2)

plot.sd.1 <- as.data.frame(VarCorr(topmod1))$sdcor[1]
resid.sd.1 <- as.data.frame(VarCorr(topmod1))$sdcor[2]

plot.sd.2 <- as.data.frame(VarCorr(topmod2))$sdcor[1]
resid.sd.2 <- as.data.frame(VarCorr(topmod2))$sdcor[2]

sims <- 10000 # run 100 to check, otherwise update to 10,000

pv.ONI <- matrix(nrow = sims, ncol = nrow(SCMU.ONI))
for(i in 1:sims){
  #choose a model
  model <- which(rmultinom(n = 1, size = 1,prob = c(AIC.tab2[,3]))==1)
  if(model == 1){
    #we simulate conditioning on no specific random effects levels
    y <- unlist(simulate(topmod1))
    bmod <- refit(topmod1,y)
    pv.ONI[i,] <- predict(bmod, re.form = ~0, newdata = SCMU.ONI) + rnorm(1,0,sd=plot.sd.1) + rnorm(1,0,sd=resid.sd.1)
  }
  if(model == 2){
```

```

#we simulate conditioning on no specific random effects levels
y <- unlist(simulate(topmod2))
bmod <- refit(topmod2,y)
pv.ONI[i,] <- predict(bmod, re.form = ~0, newdata = SCMU.ONI) + rnorm(1,0,sd=plot.sd.2) + rnorm(1,0)
}
}

```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

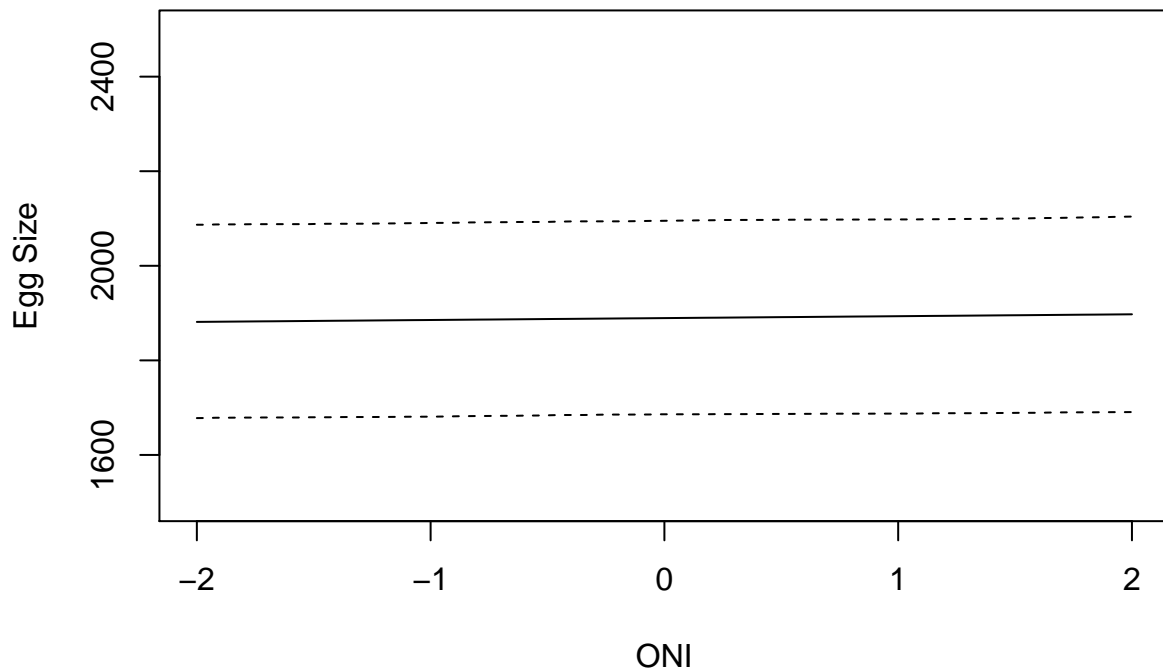
[illegible]

[illegible]

[illegible]

```
## check plots
```

```
plot(SCMU.ONI$ONI[1:100],y = apply(pv.ONI,2,mean)[1:100],xlab = "ONI",ylab = "Egg Size",type = "l",ylim = 0.025,las=1)
lines(SCMU.ONI$ONI[1:100],y = apply(pv.ONI,2,function(x)quantile(x,probs= 0.025))[1:100],lty=2)
lines(SCMU.ONI$ONI[1:100],y = apply(pv.ONI,2,function(x)quantile(x,probs= 0.975))[1:100],lty=2)
```



Plots

```
# Updated plot - include both egg order
# plot(SCMU.BEUTI$BEUTI[1:100], y = apply(pv.BEUTI, 2, mean)[1:100], xlab = "BEUTI", ylab = "Egg Size", type = "l", col = "red")
# lines(SCMU.BEUTI$BEUTI[101:200], y = apply(pv.BEUTI, 2, mean)[101:200], col = "red")
# lines(SCMU.BEUTI$BEUTI[101:200], y = apply(pv.BEUTI, 2, function(x) quantile(x, probs = 0.025))[101:200], lty = 2, col = "red")
# lines(SCMU.BEUTI$BEUTI[101:200], y = apply(pv.BEUTI, 2, function(x) quantile(x, probs = 0.975))[101:200], lty = 2, col = "red")

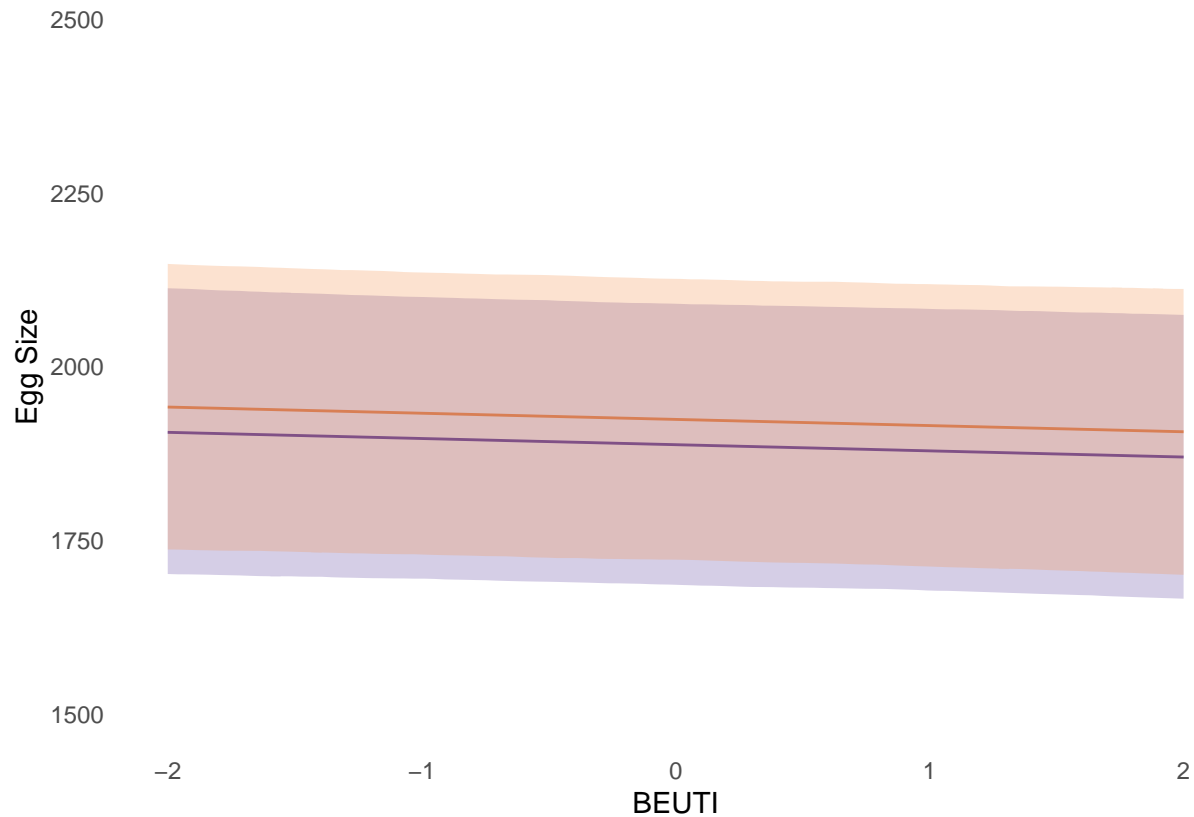
## ggplot
color_pal <- c("#593d9cff", "#f68f46ff")

### BEUTI
y1 <- as.data.frame(apply(pv.BEUTI, 2, mean))
quantile.low1 <- as.data.frame(apply(pv.BEUTI, 2, function(x) quantile(x, probs = 0.025)))
quantile.high1 <- as.data.frame(apply(pv.BEUTI, 2, function(x) quantile(x, probs = 0.975)))
BEUTI.pred <- cbind(y1, quantile.low1, quantile.high1, SCMU.BEUTI$BEUTI) %>%
  mutate(EggOrder = c(rep("Egg1", 100), rep("Egg2", 100)))
colnames(BEUTI.pred) <- c("mean", "low", "high", "BEUTI", "egg_order")

p1 <- ggplot(data = BEUTI.pred) +
  geom_line(aes(x = BEUTI, y = mean, color = egg_order)) +
  geom_ribbon(aes(ymin = low, ymax = high, x = BEUTI, fill = egg_order), alpha = 0.25, linetype = "dashed") +
  ylim(c(1500, 2500)) +
  scale_color_manual(values = color_pal) +
  scale_fill_manual(values = color_pal) +
```

```
ylab("Egg Size") +
theme_minimal() +
theme(panel.grid.major = element_blank(),
      panel.grid.minor = element_blank(),
      legend.position = "none")
```

p1



```
### NPGO
```

```
y2 <- as.data.frame(apply(pv.NPGO,2,mean))
quantile.low2 <- as.data.frame(apply(pv.NPGO,2,function(x)quantile(x,probs= 0.025)))
quantile.high2 <- as.data.frame(apply(pv.NPGO,2,function(x)quantile(x,probs= 0.975)))
NPGO.pred <- cbind(y2, quantile.low2, quantile.high2, SCMU.NPGO$NPGO) %>%
  mutate(EggOrder = c(rep("Egg1",100),rep("Egg2",100)))
colnames(NPGO.pred) <- c("mean", "low", "high", "NPGO", "egg_order")
```

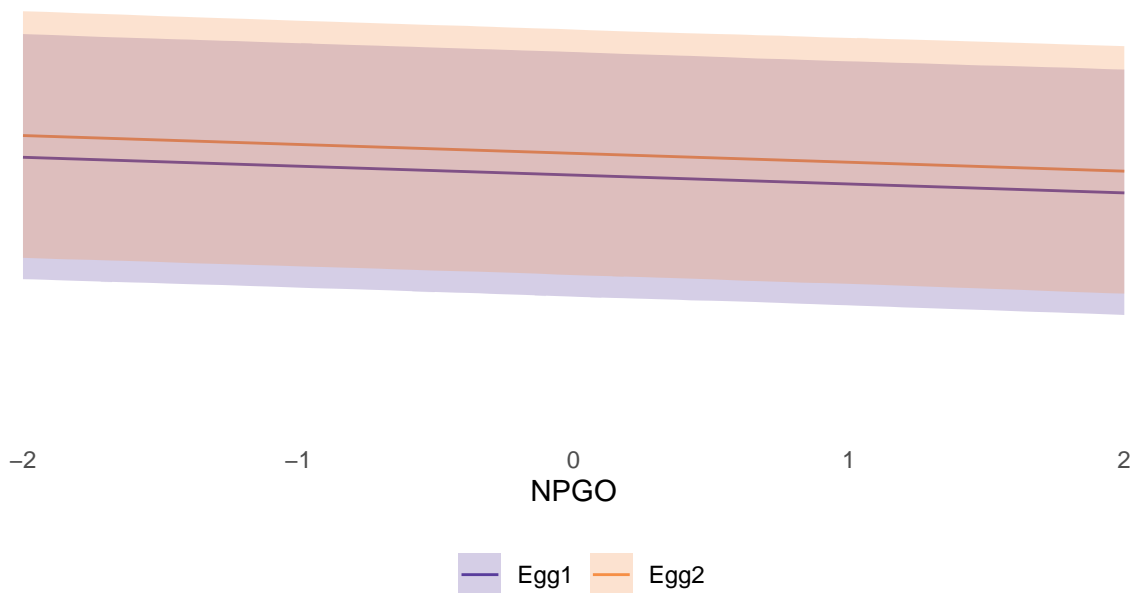
```
p2 <- ggplot(data = NPGO.pred) +
  geom_line(aes(x = NPGO, y = mean, color = egg_order)) +
  geom_ribbon(aes(ymin = low, ymax = high, x = NPGO, fill = egg_order), alpha = 0.25, linetype = "dashed") +
  ylim(c(1500,2500)) +
  scale_color_manual(values = color_pal) +
  scale_fill_manual(values = color_pal) +
  ylab("Egg Size") +
  theme_minimal() +
  guides(size = FALSE) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
```

```

axis.title.y = element_blank(),
axis.text.y = element_blank(),
axis.ticks.y = element_blank(),
legend.title = element_blank(),
legend.position = "bottom")

```

p2



```

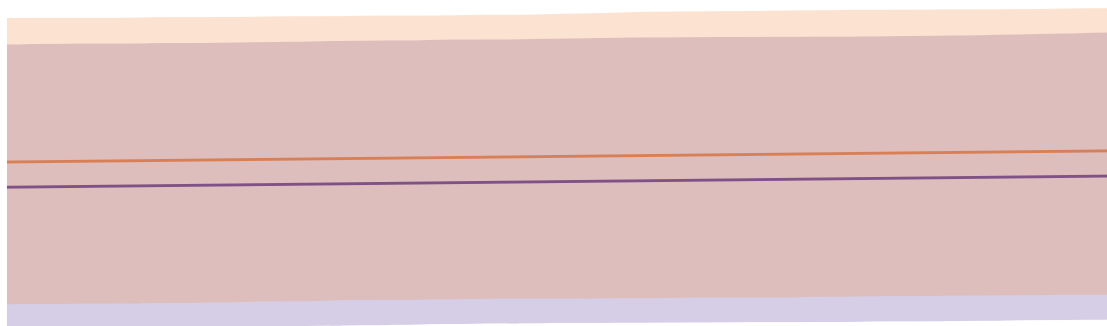
### ONI
y3 <- as.data.frame(apply(pv.ONI,2,mean))
quantile.low3 <- as.data.frame(apply(pv.ONI,2,function(x)quantile(x,probs= 0.025)))
quantile.high3 <- as.data.frame(apply(pv.ONI,2,function(x)quantile(x,probs= 0.975)))
ONI.pred <- cbind(y3, quantile.low3, quantile.high3, SCMU.ONI$ONI) %>%
  mutate(EggOrder = c(rep("Egg1",100),rep("Egg2",100)))
colnames(ONI.pred) <- c("mean", "low", "high", "ONI", "egg_order")

p3 <- ggplot(data = ONI.pred) +
  geom_line(aes(x = ONI, y = mean, color = egg_order)) +
  geom_ribbon(aes(ymin = low, ymax = high, x = ONI, fill = egg_order), alpha = 0.25, linetype = "dashed") +
  ylim(c(1500,2500)) +
  scale_color_manual(values = color_pal) +
  scale_fill_manual(values = color_pal) +
  ylab("Egg Size") +
  theme_minimal() +
  guides(size = FALSE) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),

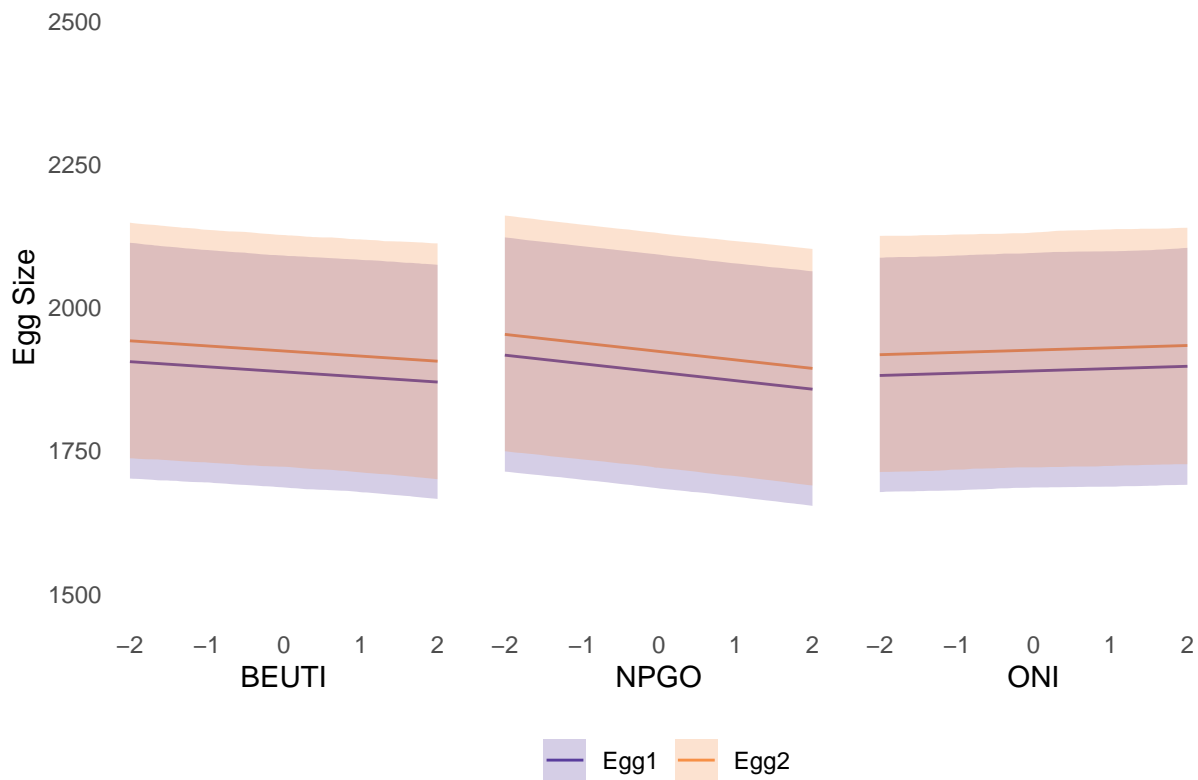
```


p3

```
axis.title.y = element_blank(),  
axis.text.y = element_blank(),  
axis.ticks.y = element_blank(),  
legend.position = "none")
```



```
## plot together  
library(patchwork)  
all <- p1 + p2 + p3  
all
```



```
# ggsave(here("results", "pred_plots.pdf"), width = 10, height = 7)
```

Model prediction for the top model only

BEUTI

```
# model prediction for one model only (top model)
pv.BEUTiv2 <- matrix(nrow = sims, ncol = nrow(SCMU.BEUTI))
for(i in 1:sims){
  #we simulate conditioning on no specific random effects levels
  y <- unlist(simulate(topmod1))
  bmod <- refit(topmod1,y)
  pv.BEUTiv2[i,] <- predict(bmod, re.form = ~0, newdata = SCMU.BEUTI) + rnorm(1,0,sd=plot.sd.1) + rnorm(1,0,sd=plot.sd.2)
}
```

```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

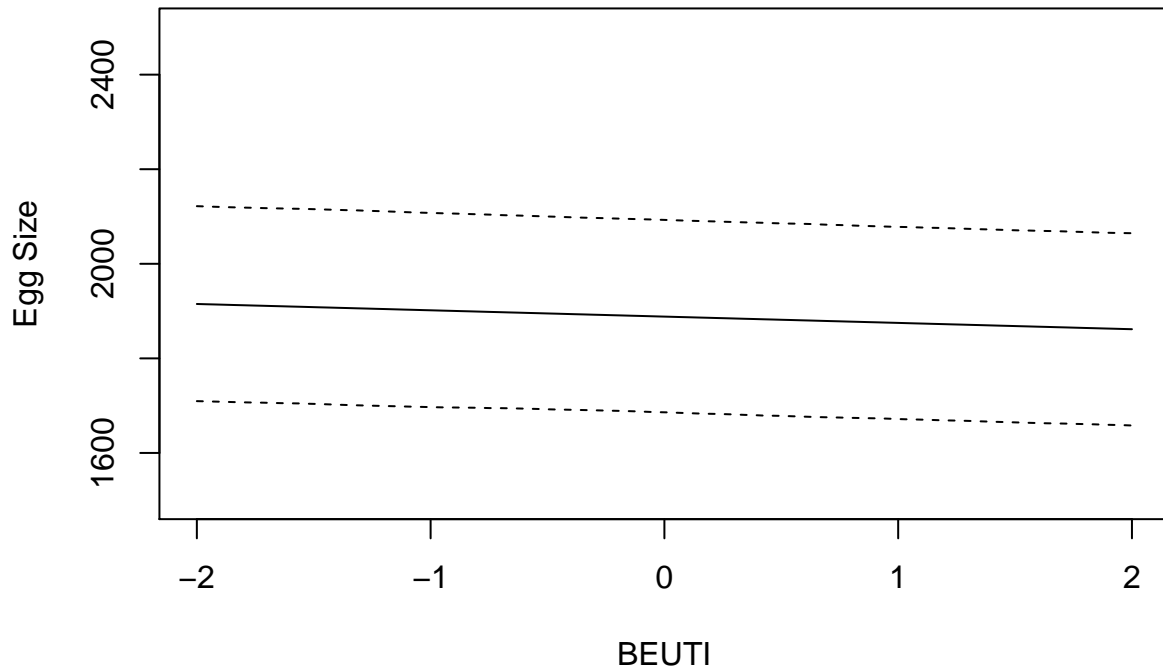
[illegible]

[illegible]

```

## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## check plots
plot(SCMU.BEUTI$BEUTI[1:100],y = apply(pv.BEUTiv2,2,mean)[1:100],xlab = "BEUTI",ylab = "Egg Size",type = "n")
lines(SCMU.BEUTI$BEUTI[1:100],y = apply(pv.BEUTiv2,2,function(x)quantile(x,probs= 0.025))[1:100],lty=2)
lines(SCMU.BEUTI$BEUTI[1:100],y = apply(pv.BEUTiv2,2,function(x)quantile(x,probs= 0.975))[1:100],lty=2)

```



NPGO

```
# model prediction for one model only (top model)
pv.NPG0v2 <- matrix(nrow = sims, ncol = nrow(SCMU.NPGO))
for(i in 1:sims){
  #we simulate conditioning on no specific random effects levels
  y <- unlist(simulate(topmod1))
  bmod <- refit(topmod1,y)
  pv.NPG0v2[i,] <- predict(bmod, re.form = ~0, newdata = SCMU.NPGO) + rnorm(1,0,sd=plot.sd.1) + rnorm
}
```

```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

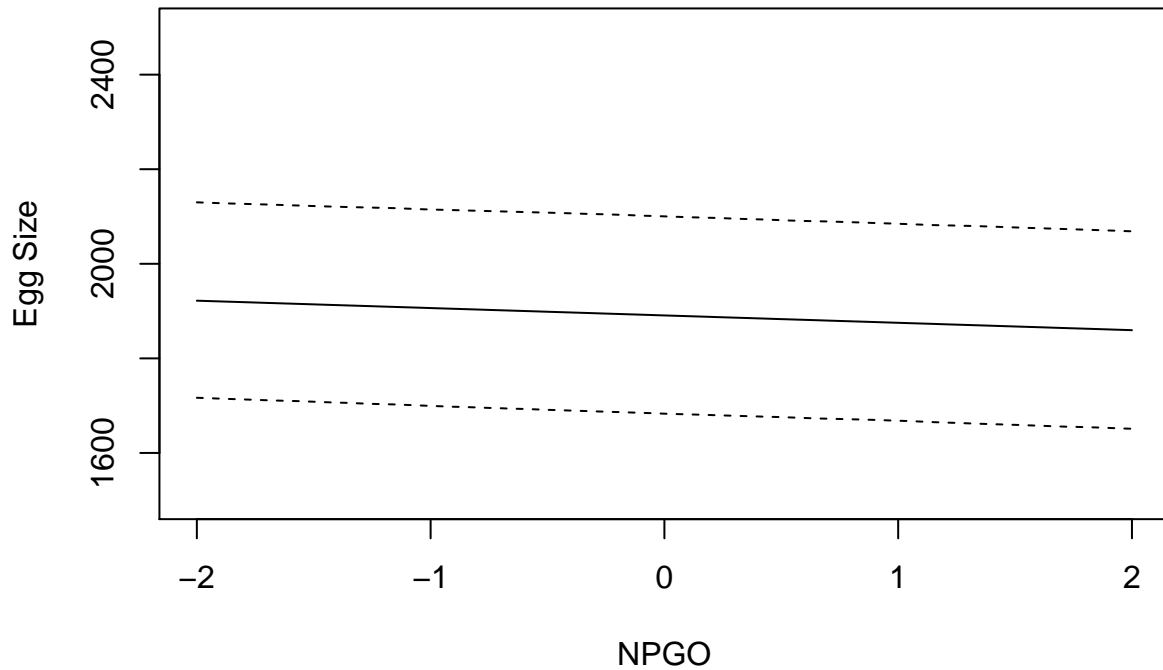
[illegible]

[illegible]

```

## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## check plots
plot(SCMU.NPGO$NPGO[1:100],y = apply(pv.NPGOv2,2,mean)[1:100],xlab = "NPGO",ylab = "Egg Size",type = "l")
lines(SCMU.NPGO$NPGO[1:100],y = apply(pv.NPGOv2,2,function(x)quantile(x,probs= 0.025))[1:100],lty=2)
lines(SCMU.NPGO$NPGO[1:100],y = apply(pv.NPGOv2,2,function(x)quantile(x,probs= 0.975))[1:100],lty=2)

```



ONI

```
# model prediction for one model only (top model)
pv.ONIv2 <- matrix(nrow = sims, ncol = nrow(SCMU.ONI))
for(i in 1:sims){
  #we simulate conditioning on no specific random effects levels
  y <- unlist(simulate(topmod1))
  bmod <- refit(topmod1,y)
  pv.ONIv2[i,] <- predict(bmod, re.form = ~0, newdata = SCMU.ONI) + rnorm(1,0,sd=plot.sd.1) + rnorm(1,0,sd=plot.sd.1)
}
```

```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

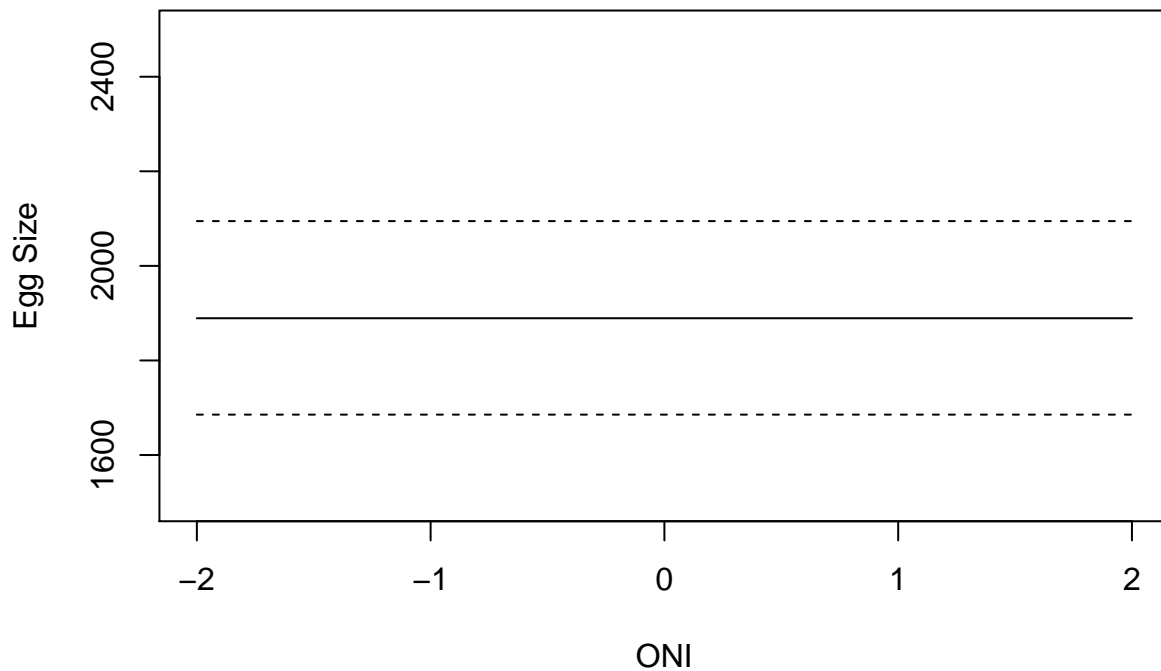
[illegible]

[illegible]

[illegible]

[illegible]


```
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## boundary (singular) fit: see ?isSingular
## check plots
plot(SCMU.ONI$ONI[1:100],y = apply(pv.ONIv2,2,mean)[1:100],xlab = "ONI",ylab = "Egg Size",type = "l",ylt=2)
lines(SCMU.ONI$ONI[1:100],y = apply(pv.ONIv2,2,function(x)quantile(x,probs= 0.025))[1:100],lty=2)
lines(SCMU.ONI$ONI[1:100],y = apply(pv.ONIv2,2,function(x)quantile(x,probs= 0.975))[1:100],lty=2)
```



Plots

```
## ggplot
color_pal <- c("#593d9cff", "#f68f46ff")

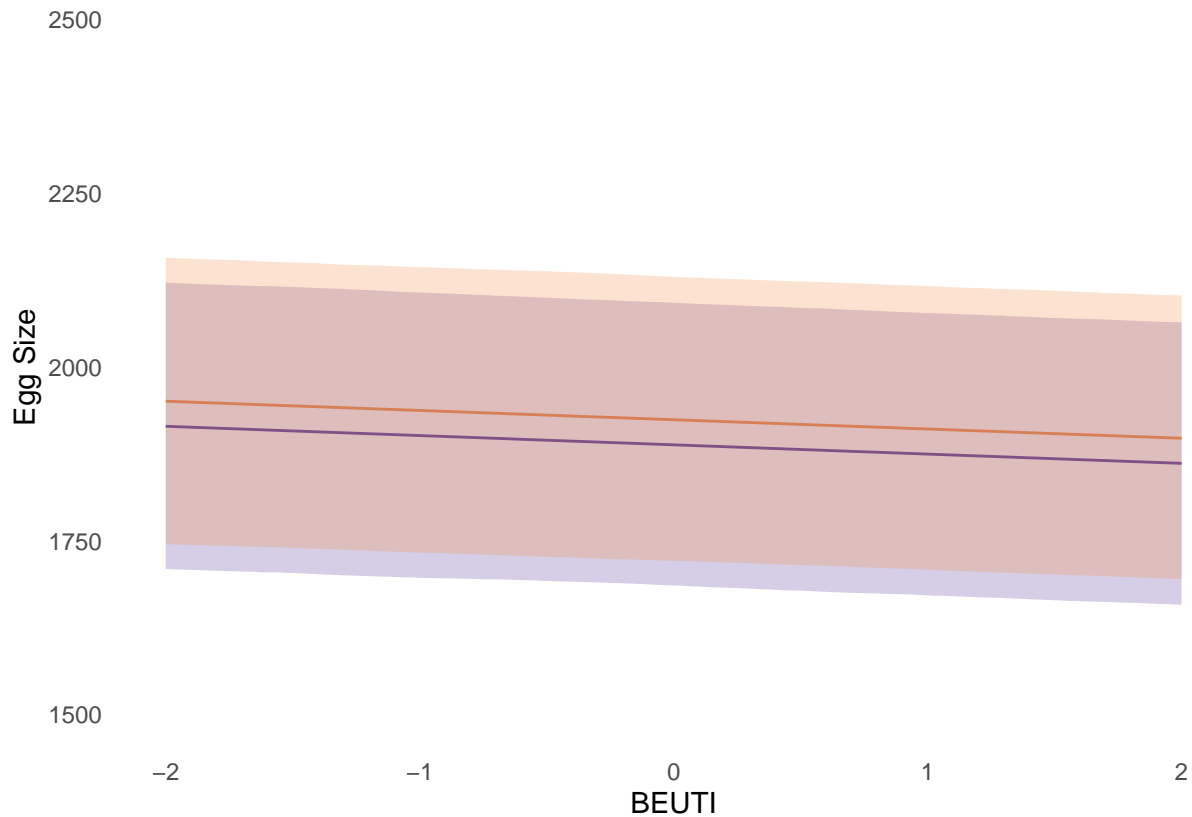
### BEUTI
y4 <- as.data.frame(apply(pv.BEUTIV2,2,mean))
```

```

quantile.low4 <- as.data.frame(apply(pv.BEUTlv2,2,function(x)quantile(x,probs= 0.025)))
quantile.high4 <- as.data.frame(apply(pv.BEUTlv2,2,function(x)quantile(x,probs= 0.975)))
BEUTI.pred4 <- cbind(y4, quantile.low4, quantile.high4, SCMU.BEUTI$BEUTI) %>%
  mutate(EggOrder = c(rep("Egg1",100),rep("Egg2",100)))
colnames(BEUTI.pred4) <- c("mean", "low", "high", "BEUTI", "egg_order")

p4 <- ggplot(data = BEUTI.pred4) +
  geom_line(aes(x = BEUTI, y = mean, color = egg_order)) +
  geom_ribbon(aes(ymin = low, ymax = high, x = BEUTI, fill = egg_order), alpha = 0.25, linetype = "dashed") +
  ylim(c(1500,2500)) +
  scale_color_manual(values = color_pal) +
  scale_fill_manual(values = color_pal) +
  ylab("Egg Size") +
  theme_minimal() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        legend.position = "none")
p4

```



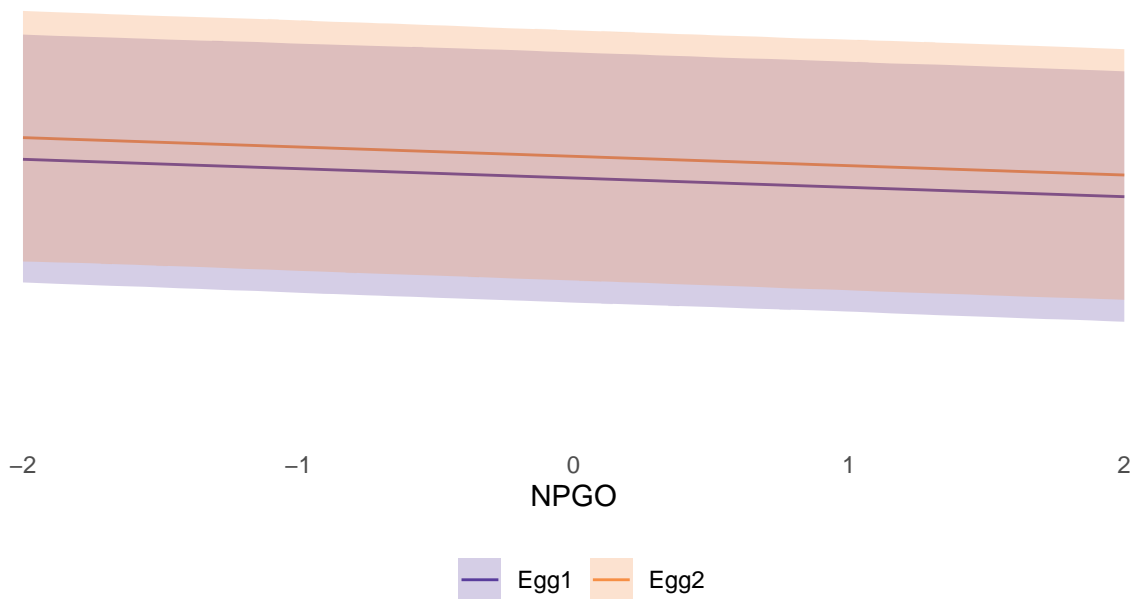
```

### NPGO
y5 <- as.data.frame(apply(pv.NPG0v2,2,mean))
quantile.low5 <- as.data.frame(apply(pv.NPG0v2,2,function(x)quantile(x,probs= 0.025)))
quantile.high5 <- as.data.frame(apply(pv.NPG0v2,2,function(x)quantile(x,probs= 0.975)))
NPG0.pred5 <- cbind(y5, quantile.low5, quantile.high5, SCMU.NPG0$NPG0) %>%
  mutate(EggOrder = c(rep("Egg1",100),rep("Egg2",100)))
colnames(NPG0.pred5) <- c("mean", "low", "high", "NPG0", "egg_order")

```

```
p5 <- ggplot(data = NPG0.pred5) +
  geom_line(aes(x = NPG0, y = mean, color = egg_order)) +
  geom_ribbon(aes(ymin = low, ymax = high, x = NPG0, fill = egg_order), alpha = 0.25, linetype = "dashed") +
  ylim(c(1500,2500)) +
  scale_color_manual(values = color_pal) +
  scale_fill_manual(values = color_pal) +
  ylab("Egg Size") +
  theme_minimal() +
  guides(size = FALSE) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.title.y = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks.y = element_blank(),
        legend.title = element_blank(),
        legend.position = "bottom")
```

p5

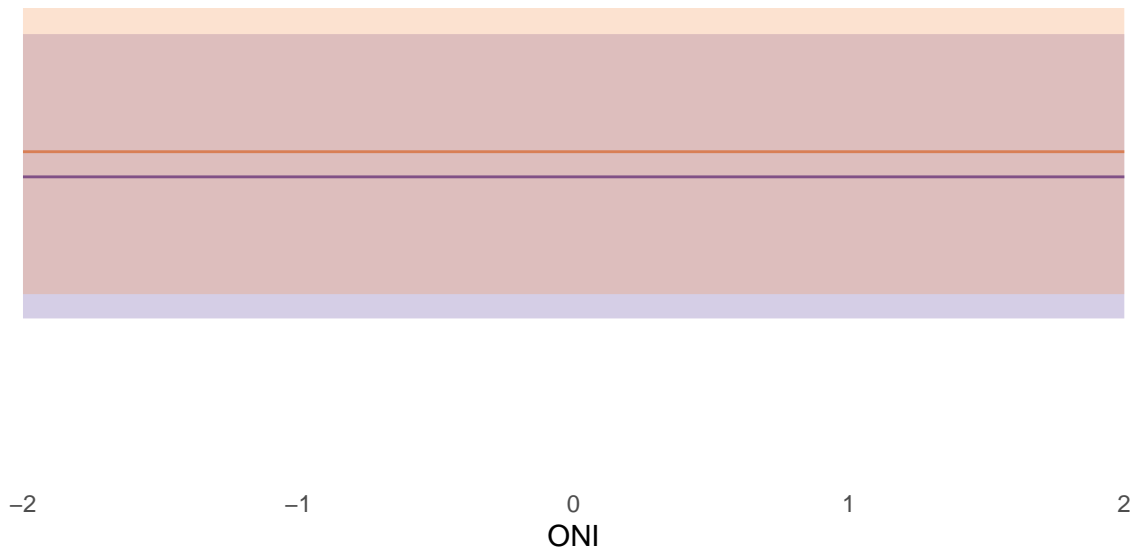


```
### ONI
y6 <- as.data.frame(apply(pv.ONIv2,2,mean))
quantile.low6 <- as.data.frame(apply(pv.ONIv2,2,function(x)quantile(x,probs= 0.025)))
quantile.high6 <- as.data.frame(apply(pv.ONIv2,2,function(x)quantile(x,probs= 0.975)))
ONI.pred6 <- cbind(y6, quantile.low6, quantile.high6, SCMU.ONI$ONI) %>%
  mutate(EggOrder = c(rep("Egg1",100),rep("Egg2",100)))
colnames(ONI.pred6) <- c("mean", "low", "high", "ONI", "egg_order")
```

```

p6 <- ggplot(data = ONI.pred6) +
  geom_line(aes(x = ONI, y = mean, color = egg_order)) +
  geom_ribbon(aes(ymin = low, ymax = high, x = ONI, fill = egg_order), alpha = 0.25, linetype = "dashed") +
  ylim(c(1500, 2500)) +
  scale_color_manual(values = color_pal) +
  scale_fill_manual(values = color_pal) +
  ylab("Egg Size") +
  theme_minimal() +
  guides(size = FALSE) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.title.y = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks.y = element_blank(),
        legend.position = "none")
p6

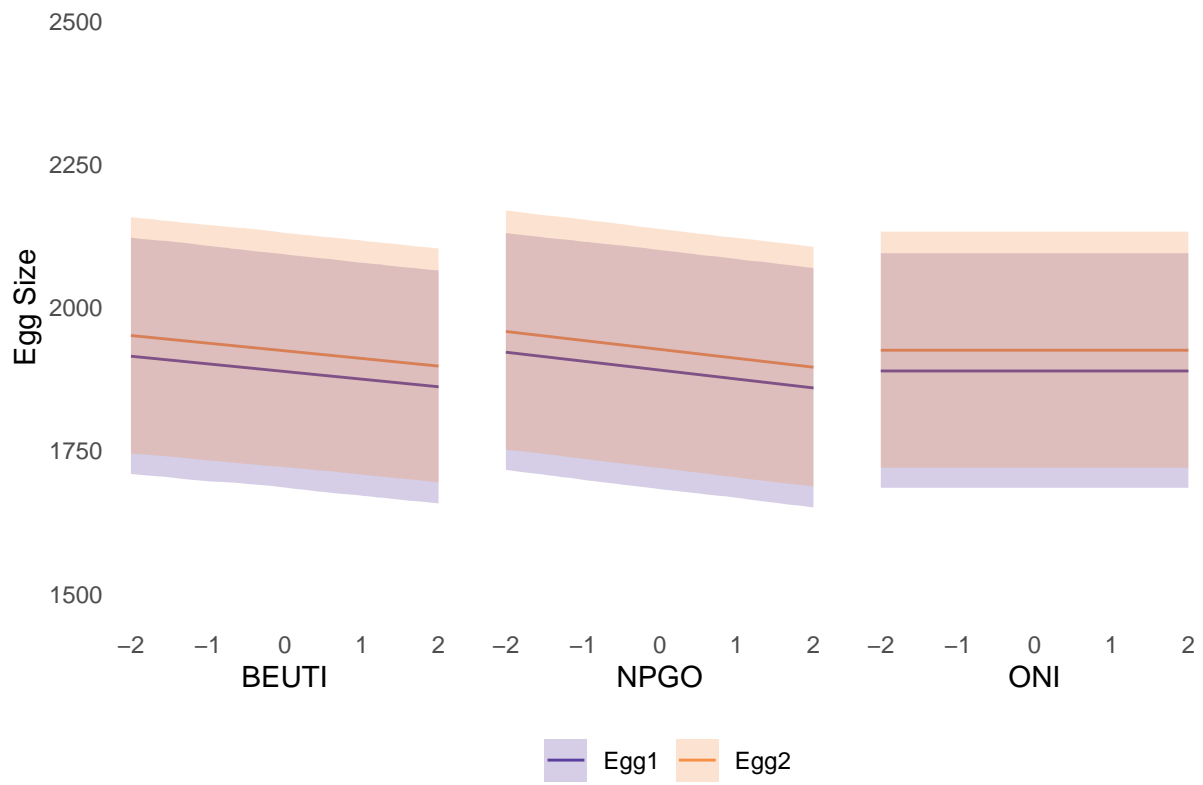
```



```

## plot together
library(patchwork)
all2 <- p4 + p5 + p6
all2

```



```
# ggsave(here("results", "pred_plots_v2.pdf"), width = 10, height = 7)
```