```
##############################################################################

# R code for Ivan Spelic for analysis of presence of P. parva

##############################################################################
#Load packages
library(arm)
library(plyr)
library(car)
library(ggbeeswarm)
library(ggplot2)
library(lattice)
library(lawstat)
library(outliers)
library(tidyverse)
library(lmerTest)

# New packages
# install.packages("sjlabelled")
# install.packages("sjmisc")
# install.packages("sjPlot")
library(sjlabelled)
library(sjmisc)
library(sjPlot)

#######################################

pp <- read.table(file = "ivanparva_final_dataset_klenpredator.txt",
                 header = TRUE,
                 dec = ".",
                 stringsAsFactors = TRUE)
```

```
# Inspect the dataframe
str(pp)



# Designate year as factor
pp$fYear <- factor(pp$year)



########################################
# DATA EXPLORATION

# Are there missing values?
colSums(is.na(pp))



# DATA EXPLORATION

# OUTLIERS

# Look at outliers in continuous covariates
Var <- c("alt", "distf", "depth", "veget", "cond", "ShWi", "SpRich")
dotplot(as.matrix(as.matrix(pp[,Var])),
      groups=FALSE,
      strip = strip.custom(bg = 'white',
                           par.strip.text = list(cex = 1.2)),
      scales = list(x = list(relation = "free", draw = TRUE),
                    y = list(relation = "free", draw = FALSE)),
      col = 1, cex  = 0.6, pch = 16,
      xlab = list(label = "Data range", cex = 1.2),
      ylab = list(label = "Data order", cex = 1.2))
```

```
# outliers with SpRich + uneven distribution of
#       alt, depth, distf

#Look at balance among months
table(pp$month)
# aug july  jun  mar  may  nov  oct sept
# 5   41   27    1   19    6   11    4
# Not good. Combine mar,may = spring; jun,july,aug = summer;
# and sept,oct,nov = autumn

# years
table(pp$year)



# river altered
table(pp$alter)
# nat reg
# 35  79
# OK

# presence of submerged vegetation
table(pp$veget)
# 0  1  2  3  4  5
# 23 15 23 13 11 29

# presence of gravel on stream bed
table(pp$gravel)



# presence of predators
```

```
table(pp$pred)



#NNS except parva
table(pp$NNS)
# good

#######################################
#CALCULATE NUMBER OF ZEROS IN THE RESPONSE VARIABLE

# How many zeros for parva?

Nzeros <- sum(pp$parva == 0)
Nzeros



Pzeros <- sum(pp$parva == 0) * 100 / nrow(pp)
round(Pzeros,0)



#######################################
#COLLINEARITY

#Plot covariates against each other and check collinearity

Coll <- c("alt", "alter", "distf", "gravel",
          "veget", "depth", "cond",
          "pred", "ShWi", "SpRich", "NNS")

panel.cor <- function(x, y, digits=1, prefix="", cex.cor = 6)
```

```
{usr <- par("usr"); on.exit(par(usr))
par(usr = c(0, 1, 0, 1))
r1=cor(x,y,use="pairwise.complete.obs")
r <- abs(cor(x, y,use="pairwise.complete.obs"))
txt <- format(c(r1, 0.123456789), digits=digits)[1]
txt <- paste(prefix, txt, sep="")
if(missing(cex.cor)) { cex <- 0.9/strwidth(txt) } else {
  cex = cex.cor}
text(0.5, 0.5, txt, cex = cex * r)}
pairs(pp[, Coll], lower.panel = panel.cor)

# Species richness and Shannon Wiener are collinear,
#     leave only Shannon Wiener, since species richness
#       also has outliers
# otherwise, nothing obvious



########################################
#PLOT RELATIONSHIPS

# Plot response variable against covariates

xyplot(parva ~ cond | alter,
      data = pp,
      layout = c(3,1),
      xlab = list(label = "cond", cex = 1.2),
      ylab = list(label = "Probability of parva",
                  cex = 1.2),
      strip = function(bg = 'white', ...)
        strip.default(bg = 'white', ...),
      scales = list(alternating = TRUE,
```

```
            x = list(relation = "free"),
            y = list(relation = "same")),
    panel = function(x,y){
     panel.grid(h=-1, v= 2)
     panel.points(x,y, col = 1, pch = 16, cex = 1.1)
     panel.abline(lm(y~x), col = 1, lwd = 3)
    })


xyplot(parva ~ distf | gravel,
    data = pp,
    layout = c(3,1),
    xlab = list(label = "distf", cex = 1.2),
    ylab = list(label = "Probability of parva",
          cex = 1.2),
    strip = function(bg = 'white', ...)
     strip.default(bg = 'white', ...),
    scales = list(alternating = TRUE,
            x = list(relation = "free"),
            y = list(relation = "same")),
    panel = function(x,y){
     panel.grid(h=-1, v= 2)
     panel.points(x,y, col = 1, pch = 16, cex = 1.1)
     panel.abline(lm(y~x), col = 1, lwd = 3)
    })

xyplot(parva ~ distf | veget,
    data = pp,
    layout = c(6,1),
    xlab = list(label = "distf", cex = 1.2),
    ylab = list(label = "Probability of parva",
```

```
            cex = 1.2),
      strip = function(bg = 'white', ...)
        strip.default(bg = 'white', ...),
      scales = list(alternating = TRUE,
                    x = list(relation = "free"),
                    y = list(relation = "same")),
      panel = function(x,y){
        panel.grid(h=-1, v= 2)
        panel.points(x,y, col = 1, pch = 16, cex = 1.1)
        panel.abline(lm(y~x), col = 1, lwd = 3)
      })

xyplot(parva ~ gravel | veget,
      data = pp,
      layout = c(6,1),
      xlab = list(label = "gravel", cex = 1.2),
      ylab = list(label = "Probability of parva",
                  cex = 1.2),
      strip = function(bg = 'white', ...)
        strip.default(bg = 'white', ...),
      scales = list(alternating = TRUE,
                    x = list(relation = "free"),
                    y = list(relation = "same")),
      panel = function(x,y){
        panel.grid(h=-1, v= 2)
        panel.points(x,y, col = 1, pch = 16, cex = 1.1)
        panel.abline(lm(y~x), col = 1, lwd = 3)
      })

xyplot(parva ~ ShWi | veget,
      data = pp,
```

```
      layout = c(6,1),
      xlab = list(label = "ShWi", cex = 1.2),
      ylab = list(label = "Probability of parva",
                  cex = 1.2),
      strip = function(bg = 'white', ...)
        strip.default(bg = 'white', ...),
      scales = list(alternating = TRUE,
                    x = list(relation = "free"),
                    y = list(relation = "same")),
      panel = function(x,y){
        panel.grid(h=-1, v= 2)
        panel.points(x,y, col = 1, pch = 16, cex = 1.1)
        panel.abline(lm(y~x), col = 1, lwd = 3)
      })

xyplot(parva ~ ShWi | gravel,
      data = pp,
      layout = c(3,1),
      xlab = list(label = "ShWi", cex = 1.2),
      ylab = list(label = "Probability of parva",
                  cex = 1.2),
      strip = function(bg = 'white', ...)
        strip.default(bg = 'white', ...),
      scales = list(alternating = TRUE,
                    x = list(relation = "free"),
                    y = list(relation = "same")),
      panel = function(x,y){
        panel.grid(h=-1, v= 2)
        panel.points(x,y, col = 1, pch = 16, cex = 1.1)
        panel.abline(lm(y~x), col = 1, lwd = 3)
      })
```

```
xyplot(parva ~ ShWi | NNS,
      data = pp,
      layout = c(3,1),
      xlab = list(label = "ShWi", cex = 1.2),
      ylab = list(label = "Probability of parva",
                  cex = 1.2),
      strip = function(bg = 'white', ...)
        strip.default(bg = 'white', ...),
      scales = list(alternating = TRUE,
                    x = list(relation = "free"),
                    y = list(relation = "same")),
      panel = function(x,y){
        panel.grid(h=-1, v= 2)
        panel.points(x,y, col = 1, pch = 16, cex = 1.1)
        panel.abline(lm(y~x), col = 1, lwd = 3)
      })



##############################################

# So, for the model
# 1. No NAs to remove
# 2. Outliers in distance farm
# 3. Lots of zeros - but that is expected
# 4. Imbalance of some categorical variables
# 5. No obvious collinearity
# 6. Possible interactions? (need more data plotting)
# 7. Lack of independence in the data due to month and year?

#####################################
```

```
#  Transform alt and distf data
#  Combine months into seasons
#  Start by fitting Bernoulli GLM

pp1 <- pp

# 2. Transform uneven covariates
pp1$logAlt <- log10(pp$alt)
pp1$logDis <- log10(pp$distf)

str(pp1)

# And plot
Var1 <- c("logAlt", "logDis")
dotplot(as.matrix(as.matrix(pp1[,Var1])),
      groups=FALSE,
      strip = strip.custom(bg = 'white',
                           par.strip.text = list(cex = 1.2)),
      scales = list(x = list(relation = "free", draw = TRUE),
                    y = list(relation = "free", draw = FALSE)),
      col = 1, cex  = 0.6, pch = 16,
      xlab = list(label = "Data range", cex = 1.2),
      ylab = list(label = "Data order", cex = 1.2))


# 3. Combine months into seasons
pp1$seas <- dplyr::recode(pp1$month, "mar" = "spring",
                                  "may" = "spring",
                                  "jun" = "summer",
                                  "july" = "summer",
```

```
                    "aug" = "summer",
                    "sept" = "autumn",
                    "oct" = "autumn",
                    "nov" = "autumn")


# Model formulation - Bernoulli GLM, intercept only
M1 <- glm(parva ~ 1,
          data = pp1,
          family = binomial(link = "logit"))


# Tested models
Litall <- glm(parva ~ gravel + pred + logDis + veget +
                + cond + alt + NNS + ShWi,
        data = pp1,
        family = binomial(link = "logit"))

Litoft <- glm(parva ~ gravel  + veget + pred + ShWi,
        data = pp1,
        family = binomial(link = "logit"))

Abiotic <- glm(parva ~ gravel + alt + cond,
        data = pp1,
        family = binomial(link = "logit"))

Biotic <- glm(parva ~ pred + ShWi + NNS + veget + logDis,
        data = pp1,
        family = binomial(link = "logit"))

Angler <- glm(parva ~ pred + ShWi + veget,
```

```
        data = pp1,
        family = binomial(link = "logit"))

Poorh <- glm(parva ~ gravel + alter + ShWi + cond + NNS,
        data = pp1,
        family = binomial(link = "logit"))

Lentic <- glm(parva ~ gravel + ShWi + veget + cond,
        data = pp1,
        family = binomial(link = "logit"))

Farvel <- glm(parva ~ logDis + gravel + logDis*gravel,
        data = pp1,
        family = binomial(link = "logit"))

Farveg <- glm(parva ~ logDis + veget + logDis*veget,
        data = pp1,
        family = binomial(link = "logit"))

Farcon <- glm(parva ~ logDis + cond + logDis*cond,
        data = pp1,
        family = binomial(link = "logit"))

Floveg <- glm(parva ~ gravel + veget + veget*gravel,
        data = pp1,
        family = binomial(link = "logit"))

# Compare the models using AIC (best fitting model has low score)
round (AIC(M1, Litall, Litoft, Abiotic, Biotic, Angler, Poorh,
       Lentic, Farvel, Farveg, Farcon, Floveg),0)
```

```
# Model averaging - take variables in both best-fitting models and create 'LenP'
LenP <- glm(parva ~ gravel + ShWi + cond,
            data = pp1,
            family = binomial(link = "logit"))

round (AIC(M1, Litall, Litoft, Abiotic, Biotic, Angler, Poorh,
      Lentic, Farvel, Farveg, Farcon, Floveg, LenP),0)

# Backward model selection with 'drop1'
round(drop1(LenP),0)

# Model:
#   parva ~ gravel + ShWi + cond
# Including 'cond' does not substantially improve model fit (AIC <3)
# Drop 'cond'?

LenP2 <- glm(parva ~ gravel + ShWi,
       data = pp1,
       family = binomial(link = "logit"))

round (AIC(M1, Litall, Litoft, Abiotic, Biotic, Angler, Poorh,
      Lentic, Farvel, Farveg, Farcon, Floveg, LenP, LenP2),0)

# MODEL VALIDATION

# Plot residuals versus fitted values
par(mfrow = c(1,1), mar = c(5,5,1,1), cex.lab = 1.2)

Fitted <- fitted(LenP2)
Resid  <- resid(LenP2, type = "pearson")
par(mfrow = c(1,1), mar = c(5,5,2,2), cex.lab = 1.2)
```

```
plot(x = Fitted, y = Resid,
    xlab = "Fitted values",
    ylab = "Pearson Residuals")
abline(h = 0, lty = 2)


# Plot model residuals against covariates in the model

# gravel
boxplot(Resid ~ gravel,
       xlab = "Presence of gravel",
       ylab = "Pearson residuals",
       pch = 16, cex = 1.2, col = "firebrick2",
       data = pp1)
abline(0,0, lty=2)

#Shannon Wiener
par(mfrow = c(1,2), mar = c(5,5,1,1))
plot(x = pp1$ShWi,
    y = Resid,
    pch = 16, cex = 1.2,
    xlab = "Shannon Wiener index",
    ylab = "Pearson residuals")
abline(h = 0, lty = 2)


par(mfrow = c(1, 1))
plot(cooks.distance(LenP2),
      xlab = "Observation",
      ylab = "Cook's distance",
```

```
      type = "h",
      ylim = c(0, 1.1),
      cex.lab = 1.5)
abline(h = 1, lty = 2)

##############################################

#We assume now that the model is a good fit to the data
# and we can examine the output and visualise the model

#Model summary
summary(LenP2)



# Get a neater table of results
tab_model(LenP2,
        show.zeroinf = F,
        dv.labels = c("Bernoulli GLM (P. parva)"),
        string.pred = "Coefficient",
        string.ci = "Conf. Int (95%)",
        string.p = "P-value",
        p.style = c("numeric"),
        emph.p = FALSE,
        transform = NULL)



#Create data on grid, and the matching X matrix
range(pp1$ShWi)
# 0 - 2.1

MyData <- expand.grid(
```

```
        gravel = levels(pp1$gravel),
          ShWi = seq(0, 2.085, length = 100))

X <- model.matrix( ~ gravel + ShWi, data = MyData)
head(MyData)

#Calculate predicted values for model LenP
MyData$Pred <- X %*% coef(LenP2)

#Calculate on the predictor scale
MyData$Pi  <- exp(MyData$Pred) / (1 + exp(MyData$Pred))

#Calculate standard erroparva (SE) for predicted values for model Lentic
MyData$se <- sqrt(  diag(X %*% vcov(LenP2) %*% t(X))  )

#Calculate the SEs on the scale of the predictor function
MyData$SeUp  <- exp(MyData$Pred + 1.96 *MyData$se) /
            (1 + exp(MyData$Pred  + 1.96 *MyData$se))
MyData$SeLo  <- exp(MyData$Pred - 1.96 *MyData$se) /
            (1 + exp(MyData$Pred  - 1.96 *MyData$se))

head(MyData)



# Define labels
levels(pp1$gravel)
label_tactic <- c(no = "No gravel", yes = "Gravel present")

p <- ggplot()
p <- p + geom_jitter(data = pp1, position = position_jitter
       (width = 0.05, height = 0.02), aes(y = parva, x = ShWi),
```

```
      shape = 19, size = 2.5, alpha = 0.7)
p <- p + labs(x = "Shannon-Wiener Index",
           y = expression(paste("Probability of ", italic("Pseudorasbora parva"))))
p <- p + theme(text = element_text(size=13))
p <- p + theme(panel.background = element_blank())
p <- p + theme(panel.border = element_rect(fill = NA, colour = "black", size = 1))
p <- p + theme(strip.background = element_rect
       (fill = "white", color = "white", size = 1))
p <- p + geom_line(data = MyData, aes(x = ShWi, y = Pi), colour = "black", size = 1)
p <- p + geom_ribbon(data = MyData, aes(x = ShWi,
                ymax = SeUp, ymin = SeLo), alpha = 0.4)
p <- p + facet_grid(.~ gravel,
                  scales = "fixed", space = "fixed", labeller=labeller
                  (gravel = label_tactic))

p

# Also plot both together with sjPlot

# Visualise the model as a figure
set_theme(
  axis.ticksize.x = 0.5,
  axis.ticksize.y = 0.5,
  axis.textsize.x = 0.5,
  axis.textsize.y = 0.5,
  axis.linecolor = "black",
  axis.textcolor = "black",
  axis.tickslen = 0.2,
  axis.title.size = 0.9,
  axis.title.color = "black",
  axis.textsize = 0.8,
```

```
 geom.outline.size = 1,
 legend.pos = "bottom",
 legend.title.face = "italic",
 legend.backgroundcol = "white",
 legend.item.backcol = "white",
 legend.item.bordercol = "white",
 base = theme_blank())

# Define preferred figure format
My_theme <- theme(panel.background = element_blank(),
                  panel.border = element_rect(fill = NA, size = 1),
                  strip.background = element_rect(fill = "white",
                                                  color = "white",
                                                  size = 1),
                  text = element_text(size = 14),
                  panel.grid.major = element_line(colour = "white",
                                                  size = 0.1),
                  panel.grid.minor = element_line(colour = "white",
                                                  size = 0.1))

# And plot
plot_model(LenP2,
           type = "pred",
           terms = c("ShWi", "gravel"),
           colors = c("red2", "blue2", "green2"),
           show.data = T,
           pred.type = c("fe"),
           title = "",
           show.legend = TRUE,
           jitter = 0.05,
           axis.title = c("Shannon-Wiener Index",
```

```
                    "Probability of Pseudorasbora parva"),
         show.values = F) + My_theme +
  scale_y_continuous(limits = c(-0.05, 1.05)) +
  scale_x_continuous(limits = c(-0.05, 2.25))



############### END ##################
```