R-code for "Dancing in the moonlight: evidence that Killdeer foraging behavior varies with lunar cycle"

Luke J. Eberhart-Phillips

In this document I provide all the necessary code for repeating the analyses presented in my paper. To access the dataset and Rmarkdown file, please download this GitHub repository. Simply follow the link and click on *Download ZIP* on the right-hand side of the page. An explanation of the files in the repository can be found in the Readme file. Please don't hesitate to contact me at `luke.eberhart[at]gmail.com` if you have any questions.

The structure of the code I present here follows the analyses presented in the *Results* section of the paper.

Prerequisites:

- For running the complete code you need a `files` subfolder with the raw data file found in the GitHub repository.
- The following packages are needed for analysis and can be easily installed from CRAN by uncommenting the `install.packages` functions:

```r
# install.packages("dplyr")
# install.packages("gridExtra")
# install.packages("ggplot2")
# install.packages("RColorBrewer")
# install.packages("grid")
# install.packages("extrafont")
# install.packages("lme4")
# install.packages("AICcmodavg")
# install.packages("VGAM")
# install.packages("tidyr")
# install.packages("stringr")
# install.packages("scales")
# install.packages("multcomp")
# install.packages("Rmisc")
# install.packages("arm")
library(dplyr)
library(gridExtra)
library(ggplot2)
library(RColorBrewer)
library(grid)
library(extrafont)
library(lme4)
library(AICcmodavg)
library(VGAM)
library(tidyr)
library(stringr)
library(scales)
library(multcomp)
library(Rmisc)
library(arm)
library(reshape)
```

Loading and wrangling data

Loading the

- **behavior data** in which each row is a single observation of the behavior (i.e. foraging, roosting, alert, or preening) of an individual on a given date, time, and scan. Within each behavior, a "1" signifies the behavior the individual was engaged in, whereas a "0" signifies the behavior the individual was not engaged in.
- **lunar data** of the proportion of the moon's disk that was illuminated the preceding night of the date specified in each row.
- **climate data** for the maximum precipitation (mm) and minimum temperature (°C) recorded during the preceding 12 hours of each survey session.

```
Behavior <- read.csv(".//Data_files//behavior_data_raw.csv")
Lunar <- read.csv(".//Data_files//lunar_data_raw.csv")
Climate <- read.csv(".//Data_files//climate_data_raw.csv")
```

Summarize the `Behavior` data such that each row reveals the total number of individuals engaged in a given behavior on each scan.

```
Behavior_totals <-
  Behavior %>%
  dplyr::group_by(Date, Time, Scan) %>%
  dplyr::summarise(Foraging = sum(Foraging),
                   Roosting = sum(Roosting),
                   Alert = sum(Alert),
                   Preening = sum(Preening))
```

Define the variables of the `Behavior_totals` correctly. `Date` needs to be recognized as a date, `Time` needs to be a factor with the order: `Sunrise`, `Midday`, and `Sunset`, and the behaviors need to be numeric.

```
Behavior_totals$Date <- as.Date(Behavior_totals$Date, "%Y-%m-%d")
Behavior_totals$Time <- factor(Behavior_totals$Time,
                               levels = c("Sunrise", "Midday", "Sunset"))
Behavior_totals[,c("Foraging", "Roosting", "Alert", "Preening")] <-
  lapply(Behavior_totals[,c("Foraging", "Roosting", "Alert", "Preening")], as.numeric)
```

Create a `Total` variable in the dataframe which sums the total number of birds observed on each scan

```
Behavior_totals$Total <- rowSums(Behavior_totals[,c(4:7)])
```

Define the variables of the `Climate` dataframe correctly. `Date` needs to be recognized as a date, `Time` needs to be a factor with the order: `Sunrise`, `Midday`, and `Sunset`, and `Min12Temp` and `Max12Precip` need to be numeric.

```
Climate$Date <- as.Date(Climate$Date, "%Y-%m-%d")
Climate[,c("Min12Temp", "Max12Precip")] <-
  lapply(Climate[,c("Min12Temp", "Max12Precip")], as.numeric)
Climate$Time <- factor(Climate$Time,
                       levels = c("Sunrise", "Midday", "Sunset"))
```

Define the variables of the `Lunar` dataframe correctly. `Date` needs to be recognized as a date and `PropMoon` need to be numeric.

```
Lunar$Date <- as.Date(Lunar$Date, "%Y-%m-%d")
Lunar[,c("PropMoon")] <- as.numeric(Lunar[,c("PropMoon")])
```

Join the three dataframes by `Date` and `Time` to create a collated dataframe henceforth referred to as `Killdeer`. Order the data according to `Date`, `Time`, and `Scan`.

```
Killdeer <- dplyr::left_join(Behavior_totals, Climate, by = c("Date", "Time"))
Killdeer <- dplyr::left_join(Killdeer, Lunar, by = c("Date"))
Killdeer <- Killdeer[with(Killdeer, order(Date, Time, Scan)), ]
head(Killdeer)
#> Source: local data frame [6 x 11]
#> Groups: Date, Time [1]
#>
#>         Date    Time  Scan Foraging Roosting Alert Preening Total
#>       (date)  (fctr) (int)    (dbl)    (dbl) (dbl)    (dbl) (dbl)
#> 1 2010-02-07 Sunrise     1       11       35     0        7    53
#> 2 2010-02-07 Sunrise     2       32       24     0        5    61
#> 3 2010-02-07 Sunrise     3       30       26     0        7    63
#> 4 2010-02-07 Sunrise     4       30       25     0        8    63
#> 5 2010-02-07 Sunrise     5       28       23     0        7    58
#> 6 2010-02-07 Sunrise     6       35       19     0        7    61
#> Variables not shown: Min12Temp (dbl), Max12Precip (dbl), PropMoon (dbl)
```

Subset dataset to include only scans that had at least one bird observed.

```
Killdeer <- dplyr::filter(Killdeer, Total != 0)
```

Create a `Session` variable that defines each unique visit to the study site

```
Killdeer$Session <- as.factor(paste(Killdeer$Date, Killdeer$Time, sep = "_"))
```

Create a `Unique_scan` variable that defines each unique scan made

```
Killdeer$Unique_scan <- as.factor(paste(Killdeer$Session, Killdeer$Scan, sep = "_"))
```

Add variables describing the number of individuals not engaged in the focal behavior

```
Killdeer$NonForaging <- Killdeer$Total - Killdeer$Foraging
Killdeer$NonRoosting <- Killdeer$Total - Killdeer$Roosting
Killdeer$NonPreening <- Killdeer$Total - Killdeer$Preening
Killdeer$NonAlert <- Killdeer$Total - Killdeer$Alert
```

Check sample sizes for `Date`, `Session`, and `Unique_scan`

```
length(levels(as.factor(Killdeer$Date)))
#> [1] 9
length(levels(as.factor(Killdeer$Session)))
#> [1] 22
length(levels(as.factor(Killdeer$Unique_scan)))
#> [1] 131
```

Temporal variation in killdeer abundance

Linear regression mixed model predicting variation in abundance across time of day (i.e. fixed effect: `Time`, random effects: `Unique_scan`, `Session`, `Date`).

```
Abundance_Time_model <-
  lme4::glmer(Total ~
              Time + (1| Unique_scan) + (1 | Session) + (1| Date),
              data = Killdeer, family = poisson)
```

Calculate the post hoc Tukey contrast statistics

```
summary(multcomp::glht(Abundance_Time_model, mcp(Time = "Tukey")))
#>
#>   Simultaneous Tests for General Linear Hypotheses
#>
#> Multiple Comparisons of Means: Tukey Contrasts
#>
#>
#> Fit: lme4::glmer(formula = Total ~ Time + (1 | Unique_scan) + (1 |
#>     Session) + (1 | Date), data = Killdeer, family = poisson)
#>
#> Linear Hypotheses:
#>                    Estimate Std. Error z value Pr(>|z|)
#> Midday - Sunrise == 0  -0.3385     0.4702  -0.720   0.7509
#> Sunset - Sunrise == 0  -1.2255     0.5191  -2.361   0.0476 *
#> Sunset - Midday == 0   -0.8870     0.5508  -1.610   0.2401
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> (Adjusted p values reported -- single-step method)
```

Calcluate summary statistics of abundance across the three times of day, then plot (Figure 2 in the paper).

```
Abundance <- Rmisc::summarySE(Killdeer, measurevar = "Total", groupvars = c("Time"))
ggplot2::ggplot(data = Abundance, aes(x = Time, y = Total)) +
        geom_bar(position = position_dodge(), stat = "identity", alpha = 0.5) +
        geom_errorbar(aes(ymin = Total - ci, ymax = Total + ci),
                      width = .2, position = position_dodge(.9)) +
        theme_bw() +
        theme(axis.title.x = element_text(size = 14, vjust = -0.5),
              axis.text.x  = element_text(size = 13),
              axis.title.y = element_text(size = 14, vjust = 1.5),
              axis.text.y  = element_text(size = 13),
              panel.grid.major = element_blank(),
              panel.grid.minor = element_blank()) +
        ylab("Abundance (± 95% CI)") +
        xlab("Time of survey")
```
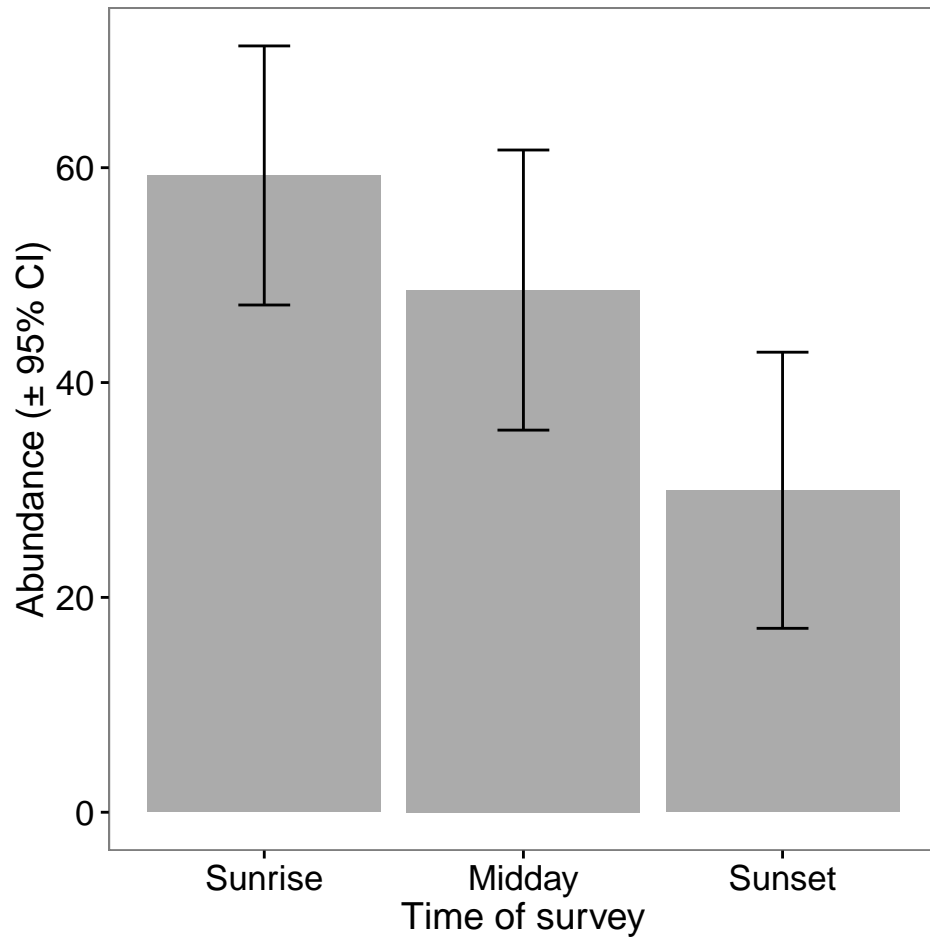
Foraging model selection, analysis, and plotting

Assign the *a priori* models to a list

```
Forage_models <- list()
```

- **Model 1: Null model**

```
Forage_models[[1]] <-
  lme4::glmer(cbind(Foraging, NonForaging) ~
                 1 +
                 (1| Unique_scan) + (1 | Session) + (1| Date),
              data = Killdeer, family = binomial)
```

- **Model 2: Univariate model with proportion of moon illuminated**

```
Forage_models[[2]] <-
  lme4::glmer(cbind(Foraging, NonForaging) ~
                 PropMoon +
```

```
                (1| Unique_scan) + (1 | Session) + (1| Date),
            data = Killdeer, family = binomial)
```

- **Model 3: Univariate model with minimum 12 hour temperature**

```
Forage_models[[3]] <-
  lme4::glmer(cbind(Foraging, NonForaging) ~
                Min12Temp +
                (1| Unique_scan) + (1 | Session) + (1| Date),
            data = Killdeer, family = binomial)
```

- **Model 4: Univariate model with max 12 hour precipitation**

```
Forage_models[[4]] <-
  lme4::glmer(cbind(Foraging, NonForaging) ~
                Max12Precip +
                (1| Unique_scan) + (1 | Session) + (1| Date),
            data = Killdeer, family = binomial)
```

- **Model 5: Additive model of both lunar and temperature covariates**

```
Forage_models[[5]] <-
  lme4::glmer(cbind(Foraging, NonForaging) ~
                Min12Temp + PropMoon +
                (1| Unique_scan) + (1 | Session) + (1| Date),
            data = Killdeer, family = binomial)
```

- **Model 6: Additive model of both lunar and precipitation covariates**

```
Forage_models[[6]] <-
  lme4::glmer(cbind(Foraging, NonForaging) ~
                Max12Precip + PropMoon +
                (1| Unique_scan) + (1 | Session) + (1| Date),
            data = Killdeer, family = binomial)
```

- **Model 7: Additive model of both lunar, temperature, and precipitation covariates**

```
Forage_models[[7]] <-
  lme4::glmer(cbind(Foraging, NonForaging) ~
                Min12Temp + Max12Precip + PropMoon +
                (1| Unique_scan) + (1 | Session) + (1| Date),
            data = Killdeer, family = binomial)
```

Name models and create a ranked AIC table

```
Modnames <- paste("Model", 1:length(Forage_models), sep = " ")
Forage_models_table <- aictab(cand.set = Forage_models, modnames = Modnames, sort = TRUE)
```

Calculate evidence ratios and add them to the AIC table

```
Evidence_Ratios <- function(AIC_table){
  ER <- vector( , nrow(AIC_table))
  for(i in 1:nrow(AIC_table)) {
    ER[i] <- AIC_table$AICcWt[1]/AIC_table$AICcWt[i]
  }
  ER[1] <- NA
  table <- cbind(AIC_table, ER)
  out <- cbind(table[, c(1:2)], round(table[, c(3:9)], 2))
}


Forage_models_table <- Evidence_Ratios(Forage_models_table)
```

Estimate the marginal R-squared of each model following Nakagawa and Schielzeth (Equ. 29 and 30 and Table 2; 2013) and add them to the AIC table to produce Table 1 presented in the paper.

```
r2_m_glmm <- function(mod){
  v <- var(as.vector(fixef(mod) %*% t(model.matrix(mod))))
  r2_m <- v/(v + VarCorr(mod)$Unique_scan[1] +
              VarCorr(mod)$Session[1] + VarCorr(mod)$Date[1] + pi^2/3)
  out <- unlist(list(r2_m = r2_m))
}
r2_m <- data.frame(round(t(as.data.frame(t(sapply(Forage_models, r2_m_glmm)))), 3))
r2_m$Modnames <-
  c("Model 1", "Model 2", "Model 3", "Model 4", "Model 5", "Model 6", "Model 7")
colnames(r2_m) <- c("r2_m", "Modnames")
Forage_models_table <- left_join(Forage_models_table, r2_m)[,c(1,2,4,6,9,10)]
Forage_models_table
#>    Modnames K Delta_AICc AICcWt     ER    r2_m
#> 1   Model 2 5       0.00   0.55     NA   0.431
#> 2   Model 5 6       2.19   0.18   2.99   0.432
#> 3   Model 6 6       2.19   0.18   2.99   0.431
#> 4   Model 7 7       4.41   0.06   9.08   0.433
#> 5   Model 1 4       7.56   0.01  43.73   0.000
#> 6   Model 3 5       8.82   0.01  82.39   0.021
#> 7   Model 4 5       9.42   0.00 111.03   0.006
```

Calculate the 95% confidence interval of the top model by creating a function which applies varying steps of the predictor (i.e., PropMoon) to the model and returns the coefficents of the intercept and beta.

```
CI_foraging <- function(offs) {
  model <- glmer(cbind(Foraging, NonForaging) ~
              I(PropMoon-offs) + (1| Unique_scan) + (1 | Session) + (1| Date),
              data = Killdeer, family = binomial)
  ests <- summary(model)$coefficients[1,1:2]
  # backlink the coefficients to the probability scale
  return(c(offs,ests,invlogit(ests[1]+c(-1,0,1)*1.96*ests[2])))
}
# specify the offs (i.e., vector of numbers from 0 to 1 stepped by 0.05)
offs_foraging <- seq(0,1,0.05)
# apply the offs vector to the function (retuning a matrix)
result_foraging <- sapply(offs_foraging,CI_foraging)
# transpose the matrix
result_foraging <- t(result_foraging)
```

```r
# convert the matrix to a data.frame
result_foraging <- data.frame(result_foraging)
# define the column names
colnames(result_foraging) <-
  c("PropMoon", "Coefficient", "Std. Error", "Upper", "Mean", "Lower")
# Create proportional variable for plotting
Killdeer$prop_Forage <- Killdeer$Foraging/Killdeer$Total
```

Plot the top model with confidence intervals

```r
ggplot2::ggplot() +
        geom_point(data = Killdeer,
                    aes(x = PropMoon, y = prop_Forage), size = 3, alpha = 0.5) +
        geom_ribbon(data = result_foraging,
                    aes(x = PropMoon, y = Mean, ymin = Lower, ymax = Upper),
                    fill = "#1B9E77", alpha = 0.25) +
        geom_line(data = result_foraging,
                aes(x = PropMoon, y = Mean), size = 1.5, colour = "#1B9E77") +
        theme_bw() +
        theme(legend.position = c(1, 1),
            legend.justification = c(1, 1),
            legend.text=element_text(size=11),
            legend.title=element_blank(),
            legend.key.height=unit(0.8,"line"),
            legend.key.width=unit(0.8,"line"),
            legend.background = element_rect(fill=NA),
            axis.title.x = element_text(size = 14, vjust = -0.5),
            axis.text.x  = element_text(size = 13),
            axis.title.y = element_text(size = 14, vjust = 1.5),
            axis.text.y  = element_text(size = 13),
            panel.grid.major = element_blank()) +
        scale_y_continuous(limits=c(0,1)) +
        ylab("Proportion of flock foraging in daylight") +
        xlab("Proportion of moon illuminated the previous night")
```
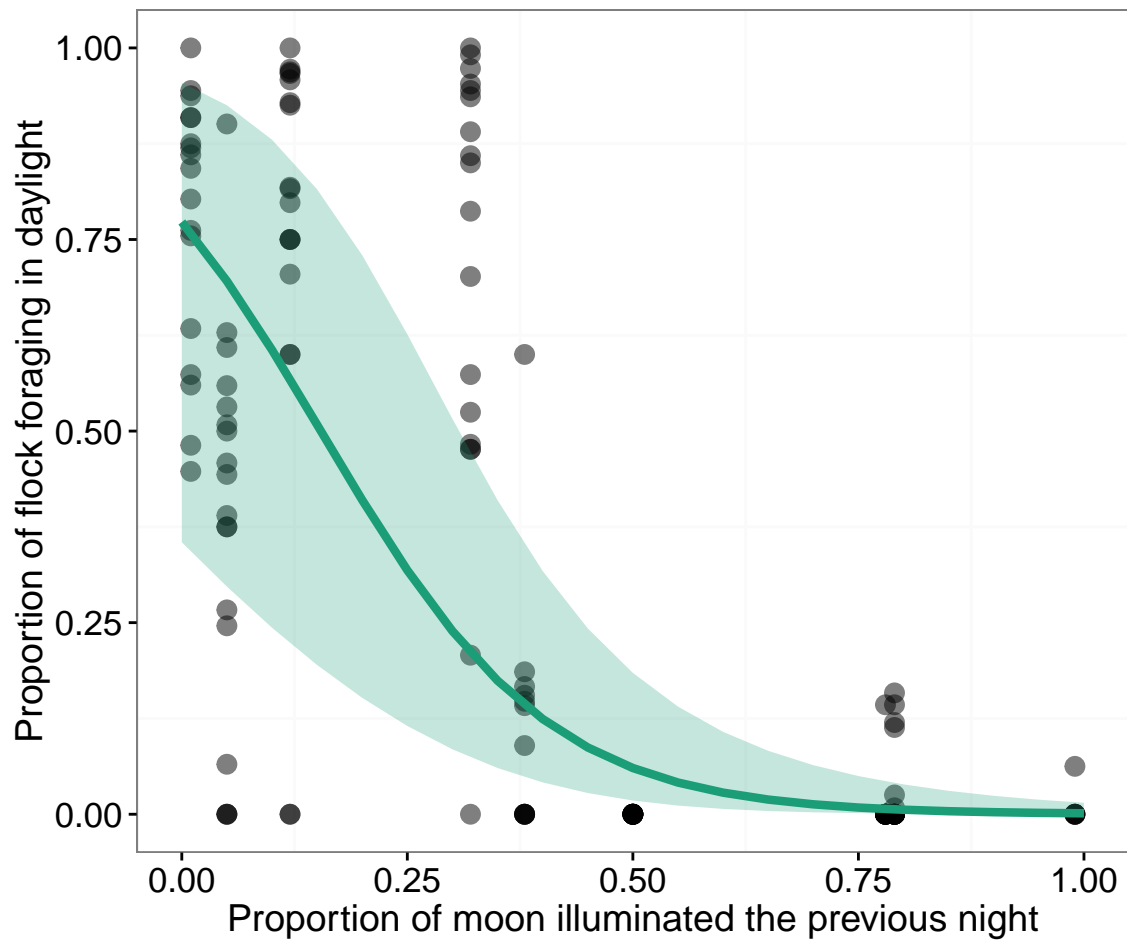
Roosting model selection, analysis, and plotting

Assign the *a priori* models to a list

```
Roosting_models <- list()
```

- **Model 1: Null model**

```
Roosting_models[[1]] <-
  lme4::glmer(cbind(Roosting, NonRoosting) ~
               1 +
               (1| Unique_scan) + (1 | Session) + (1| Date),
             data = Killdeer, family = binomial)
```

- **Model 2: Univariate model with proportion of moon illuminated**

```
Roosting_models[[2]] <-
       lme4::glmer(cbind(Roosting, NonRoosting) ~
               PropMoon +
```

```
                 (1| Unique_scan) + (1 | Session) + (1| Date),
              data = Killdeer, family = binomial)
```

- **Model 3: Univariate model with minimum 12 hour temperature**

```
Roosting_models[[3]] <-
  lme4::glmer(cbind(Roosting, NonRoosting) ~
               Min12Temp +
               (1| Unique_scan) + (1 | Session) + (1| Date),
             data = Killdeer, family = binomial)
```

- **Model 4: Univariate model with max 12 hour precipitation**

```
Roosting_models[[4]] <-
  lme4::glmer(cbind(Roosting, NonRoosting) ~
          Max12Precip +
          (1| Unique_scan) + (1 | Session) + (1| Date),
        data = Killdeer, family = binomial)
```

- **Model 5: Additive model of both lunar and temperature covariates**

```
Roosting_models[[5]] <-
  lme4::glmer(cbind(Roosting, NonRoosting) ~
               Min12Temp + PropMoon +
               (1| Unique_scan) + (1 | Session) + (1| Date),
             data = Killdeer, family = binomial)
```

- **Model 6: Additive model of both lunar and precipitation covariates**

```
Roosting_models[[6]] <-
  lme4::glmer(cbind(Roosting, NonRoosting) ~
               Max12Precip + PropMoon +
               (1| Unique_scan) + (1 | Session) + (1| Date),
             data = Killdeer, family = binomial)
```

- **Model 7: Additive model of both lunar, temperature, and precipitation covariates**

```
Roosting_models[[7]] <-
  lme4::glmer(cbind(Roosting, NonRoosting) ~
          Min12Temp + Max12Precip + PropMoon +
          (1| Unique_scan) + (1 | Session) + (1| Date),
        data = Killdeer, family = binomial)
```

Name models, create a ranked AIC table, calculate evidence ratios and marginal r-squared, and produce Table 2 presented in the paper.

```r
Modnames <- paste("Model", 1:length(Roosting_models), sep = " ")
Roost_models_table <- aictab(cand.set = Roosting_models, modnames = Modnames, sort = TRUE)
Roost_models_table <- Evidence_Ratios(Roost_models_table)
r2_m <- data.frame(round(t(as.data.frame(t(sapply(Roosting_models, r2_m_glmm)))), 3))
r2_m$Modnames <-
  c("Model 1", "Model 2", "Model 3", "Model 4", "Model 5", "Model 6", "Model 7")
colnames(r2_m) <- c("r2_m", "Modnames")
Roost_models_table <- left_join(Roost_models_table, r2_m)[,c(1,2,4,6,9,10)]
Roost_models_table
#>   Modnames K Delta_AICc AICcWt    ER  r2_m
#> 1  Model 2 5       0.00   0.49    NA 0.364
#> 2  Model 6 6       1.63   0.22  2.25 0.374
#> 3  Model 5 6       2.19   0.16  2.99 0.365
#> 4  Model 7 7       3.77   0.07  6.59 0.378
#> 5  Model 1 4       5.95   0.02 19.58 0.000
#> 6  Model 4 5       6.28   0.02 23.05 0.057
#> 7  Model 3 5       7.57   0.01 44.10 0.018
```

Plot the top model with confidence intervals

```r
CI_roosting <- function(offs) {
  model <- glmer(cbind(Roosting, NonRoosting) ~
               I(PropMoon-offs) + (1| Unique_scan) + (1 | Session) + (1| Date),
               data = Killdeer, family = binomial)
  ests <- summary(model)$coefficients[1,1:2]
  return(c(offs,ests,invlogit(ests[1]+c(-1,0,1)*1.96*ests[2])))
}
offs_roosting <- seq(0,1,0.05)
result_roosting <- sapply(offs_roosting,CI_roosting)
result_roosting <- t(result_roosting)
result_roosting <- data.frame(result_roosting)
colnames(result_roosting) <-
  c("PropMoon", "Coefficient", "Std. Error", "Upper", "Mean", "Lower")
Killdeer$prop_Roost <- Killdeer$Roosting/Killdeer$Total
ggplot2::ggplot() +
        geom_point(data = Killdeer,
                   aes(x = PropMoon, y = prop_Roost), size = 3, alpha = 0.5) +
        geom_ribbon(data = result_roosting,
                    aes(x = PropMoon, y = Mean, ymin = Lower, ymax = Upper),
                    fill = "#D95F02", alpha = 0.25) +
        geom_line(data = result_roosting,
                  aes(x = PropMoon, y = Mean), size = 1.5, colour = "#D95F02") +
        theme_bw() +
        theme(legend.position = c(1, 1),
              legend.justification = c(1, 1),
              legend.text=element_text(size=11),
              legend.title=element_blank(),
              legend.key.height=unit(0.8,"line"),
              legend.key.width=unit(0.8,"line"),
              legend.background = element_rect(fill=NA),
              axis.title.x = element_text(size = 14, vjust = -0.5),
              axis.text.x  = element_text(size = 13),
              axis.title.y = element_text(size = 14, vjust = 1.5),
```

```
            axis.text.y  = element_text(size = 13),
            panel.grid.major = element_blank()) +
     scale_y_continuous(limits=c(0,1)) +
     ylab("Proportion of flock roosting in daylight") +
     xlab("Proportion of moon illuminated the previous night")
```
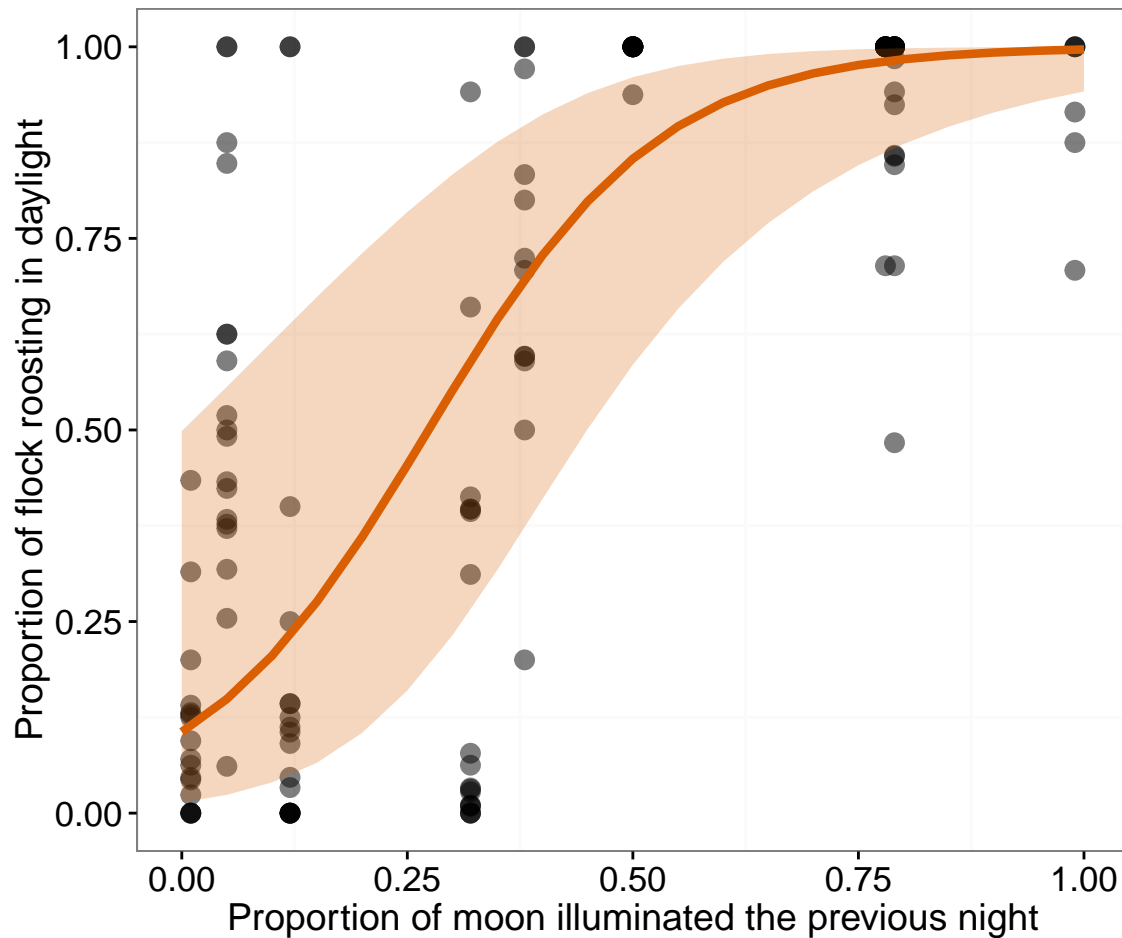

Preening model selection, analysis, and plotting

Assign the *a priori* models to a list

```
Preening_models <- list()
```

- **Model 1: Null model**

```
Preening_models[[1]] <-
  lme4::glmer(cbind(Preening, NonPreening) ~
                1 +
                (1| Unique_scan) + (1 | Session) + (1| Date),
            data = Killdeer, family = binomial)
```

- **Model 2: Univariate model with proportion of moon illuminated**

```
Preening_models[[2]] <-
  lme4::glmer(cbind(Preening, NonPreening) ~
                PropMoon +
                (1| Unique_scan) + (1 | Session) + (1| Date),
              data = Killdeer, family = binomial)
```

- **Model 3: Univariate model with minimum 12 hour temperature**

```
Preening_models[[3]] <-
  lme4::glmer(cbind(Preening, NonPreening) ~
          Min12Temp +
          (1| Unique_scan) + (1 | Session) + (1| Date),
        data = Killdeer, family = binomial)
```

- **Model 4: Univariate model with max 12 hour precipitation**

```
Preening_models[[4]] <-
  lme4::glmer(cbind(Preening, NonPreening) ~
                Max12Precip +
                (1| Unique_scan) + (1 | Session) + (1| Date),
              data = Killdeer, family = binomial)
```

- **Model 5: Additive model of both lunar and temperature covariates**

```
Preening_models[[5]] <-
  lme4::glmer(cbind(Preening, NonPreening) ~
                Min12Temp + PropMoon +
                (1| Unique_scan) + (1 | Session) + (1| Date),
              data = Killdeer, family = binomial)
```

- **Model 6: Additive model of both lunar and precipitation covariates**

```
Preening_models[[6]] <-
  lme4::glmer(cbind(Preening, NonPreening) ~
                Max12Precip + PropMoon +
                (1| Unique_scan) + (1 | Session) + (1| Date),
              data = Killdeer, family = binomial)
```

- **Model 7: Additive model of both lunar, temperature, and precipitation covariates**

```
Preening_models[[7]] <-
  lme4::glmer(cbind(Preening, NonPreening) ~
                Min12Temp + Max12Precip + PropMoon +
                (1| Unique_scan) + (1 | Session) + (1| Date),
              data = Killdeer, family = binomial)
```

Name models, create a ranked AIC table, calculate evidence ratios and marginal r-squared, and produce Table 3 presented in the paper.

```r
Modnames <- paste("Model", 1:length(Preening_models), sep = " ")
Preen_models_table <- aictab(cand.set = Preening_models, modnames = Modnames, sort = TRUE)
Preen_models_table <- Evidence_Ratios(Preen_models_table)
r2_m <- data.frame(round(t(as.data.frame(t(sapply(Preening_models, r2_m_glmm)))), 3))
r2_m$Modnames <-
  c("Model 1", "Model 2", "Model 3", "Model 4", "Model 5", "Model 6", "Model 7")
colnames(r2_m) <- c("r2_m", "Modnames")
Preen_models_table <- left_join(Preen_models_table, r2_m)[,c(1,2,4,6,9,10)]
Preen_models_table
#>   Modnames K Delta_AICc AICcWt   ER   r2_m
#> 1  Model 4 5       0.00   0.35   NA  0.040
#> 2  Model 2 5       1.25   0.19 1.86  0.039
#> 3  Model 6 6       1.35   0.18 1.96  0.051
#> 4  Model 1 4       2.27   0.11 3.11  0.000
#> 5  Model 7 7       3.22   0.07 4.99  0.053
#> 6  Model 5 6       3.42   0.06 5.52  0.040
#> 7  Model 3 5       4.08   0.05 7.67  0.005
```

This is a test