R-CODES

Supplementary material

ABSTRACT

The Moon as Semaphore: Occupancy Modeling Reveals Contrasting Lunar Responses in Elusive Terraranas of a Tropical Montane Forest

VICTOR ACOSTA CHAVES ET AL.

Contents

OCCUPANCY MODELING AND RELATIVE IMPORTANCE ANALYSIS FOR PRISTIMANTIS.....2
OCCUPANCY MODELING AND RELATIVE IMPORTANCE ANALYSIS FOR DIASPORUS.....25
#RELATIVE IMPORTANCE OF COVARIATES IN MODELS (FINAL VISUALIZATION).....42
#FINAL PLOTS FOR VISUALIZATION (OCCUPANCY) 55

OCCUPANCY MODELING AND RELATIVE IMPORTANCE ANALYSIS FOR PRISTIMANTIS

```
#============================================================================
#
# This script performs single-season occupancy modeling for two Pristimantis
# frog species to evaluate environmental and sampling covariates affecting
# detection and occupancy probabilities. The analysis follows the
# single-species, single-season occupancy modeling framework of MacKenzie et al. (2002).
#
# Species: Pristimantis caryophyllaceus and Pristimantis cruentus
# Study System: Tropical montane forest, Costa Rica
#
# Covariates:
# - Detection: absolute humidity, lunar cycle
# - Occupancy: tree diameter (DBH), tree richness, tree abundance
#
# Analysis includes:
# 1. Data preparation and correlation analysis
# 2. Model selection using AIC
# 3. Prediction plots with back-transformed scales
# 4. Relative importance of covariates
#
#============================================================================

# Load required libraries for ecological analysis
library(tidyverse)     # Data manipulation and visualization
library(readxl)        # Excel file reading
library(unmarked)      # Occupancy modeling framework
library(patchwork)     # Plot arrangement and composition
library(showtext)      # Custom fonts for publication-quality figures
library(corrplot)      # Correlation matrix visualization
library(conflicted)    # Conflict resolution for function names
library(correlation)   # Advanced correlation analysis
library(ggcorrplot)    # ggplot2-based correlation plots

# Resolve common function conflicts in ecological analysis
```

```
conflicts_prefer(
  dplyr::filter,    # Prefer dplyr filter over stats
  dplyr::lag,       # Prefer dplyr lag over stats
  dplyr::select,    # Prefer dplyr select over MASS
  dplyr::mutate,    # Ensure dplyr mutate is used
  dplyr::arrange,   # Ensure dplyr arrange is used
  dplyr::summarize  # Ensure dplyr summarize is used
)

# Configure publication-quality typography
showtext_auto()
font_add_google("Lato", "lato")

#==============================================================================
# ABSOLUTE HUMIDITY CALCULATION FUNCTION
#==============================================================================

# Function to calculate absolute humidity (g/m^3) from temperature (°C) and relative humidity (%)
calculate_absolute_humidity <- function(temp_c, rh_percent) {
  # Convert temperature to Kelvin
  temp_k <- temp_c + 273.15

  # Saturation vapor pressure (Tetens equation) in hPa
  es_hpa <- 6.112 * exp((17.67 * temp_c) / (temp_c + 243.5))

  # Actual vapor pressure in hPa
  ea_hpa <- (rh_percent / 100) * es_hpa

  # Absolute humidity in g/m^3
  absolute_humidity <- (216.7 * ea_hpa) / temp_k

  return(absolute_humidity)
}

#==============================================================================
# DATA IMPORT AND PREPARATION
#==============================================================================

# Read ecological survey data from original Excel file
file_path <- "C:/Users/Victor Acosta/Desktop/ecologyandevolution/especies_modelos_RM.xlsx"
data <- read_excel(file_path)

# Calculate absolute humidity for each temperature/humidity pair
data_with_ah <- data %>%
  # Convert to numeric and clean data
  mutate(across(starts_with("temp"), ~as.numeric(gsub(",", ".", .)))) %>%
  mutate(across(starts_with("hum"), ~as.numeric(gsub(",", ".", .)))) %>%
  # Calculate absolute humidity for each temp/hum pair
```

```
 mutate(across(starts_with("temp"),
          ~calculate_absolute_humidity(., get(paste0("hum", str_sub(cur_column(), 5)))),
          .names = "ah_{str_sub(.col, 5)}"))

# Use the data with absolute humidity for analysis
data <- data_with_ah

#------------------------------------------------------------------------------
# FUNCTION: PREPARE UNMARKED DATA FOR SINGLE-SPECIES OCCUPANCY ANALYSIS
#
# This function formats detection/non-detection data into the unmarked framework
# for occupancy modeling. It handles scaling of continuous covariates and
# preserves original values for back-transformation of predictions.
#
# Parameters:
#   species_name: Character string of target species name
#
# Returns:
#   List containing:
#    - umf: unmarkedFrame for occupancy modeling
#    - original_values: Means and SDs for back-transformation
#------------------------------------------------------------------------------

prepare_species_data <- function(species_name) {
 # Filter data for focal species
 species_data <- data %>% filter(especie == species_name)

 # Store original values before scaling for ecological interpretation
 original_values <- list()

 # Extract detection history (oc1 to oc10) - binary detection matrix
 y <- species_data %>%
  select(starts_with("oc")) %>%
  as.matrix()

 # Extract and clean site covariates (habitat characteristics)
 siteCovs <- species_data %>%
  select(area, dap, tree_rich, tree_abu) %>%
  mutate(across(c(dap, tree_rich, tree_abu), ~ as.numeric(gsub(",", ".", .x))))

 # Extract and scale observation covariates (absolute humidity and moon)
 ah_data <- species_data %>%
  select(starts_with("ah")) %>%
  mutate(across(everything(), ~as.numeric(gsub(",", ".", .)))) %>%
  as.matrix()

 moon_data <- species_data %>%
  select(starts_with("moon")) %>%
```

```
  mutate(across(everything(), ~as.numeric(gsub(",", ".", .)))) %>%
  as.matrix()

 # Store original ecological values for meaningful interpretation
 original_values$ah <- c(mean = mean(ah_data, na.rm = TRUE),
                         sd = sd(ah_data, na.rm = TRUE))
 original_values$moon <- c(mean = mean(moon_data, na.rm = TRUE),
                           sd = sd(moon_data, na.rm = TRUE))
 original_values$dap <- c(mean = mean(siteCovs$dap, na.rm = TRUE),
                          sd = sd(siteCovs$dap, na.rm = TRUE))

 obsCovs <- list(ah = ah_data, moon = moon_data)

 # Create unmarked frame for occupancy analysis
 umf <- unmarkedFrameOccu(y = y, siteCovs = siteCovs, obsCovs = obsCovs)

 # Scale covariates for model convergence and comparison
 umf@siteCovs$dap <- scale(umf@siteCovs$dap)
 umf@siteCovs$tree_rich <- scale(umf@siteCovs$tree_rich)
 umf@siteCovs$tree_abu <- scale(umf@siteCovs$tree_abu)

 umf@obsCovs$ah <- scale(umf@obsCovs$ah)
 umf@obsCovs$moon <- scale(umf@obsCovs$moon)

 return(list(umf = umf, original_values = original_values))
}

#==============================================================================
# CORRELATION ANALYSIS AMONG ECOLOGICAL COVARIATES
#==============================================================================

#------------------------------------------------------------------------------
# FUNCTION: CORRELATION ANALYSIS FOR MULTICOLLINEARITY ASSESSMENT
#
# Evaluates pairwise correlations among environmental covariates to identify
# potential multicollinearity issues before occupancy modeling. High correlations
# (>0.7) may require careful model selection to avoid overparameterization.
#
# Parameters:
#   species_name: Character string of target species name
#
# Returns:
#   List containing correlation matrix and complete cases data
#------------------------------------------------------------------------------

correlation_analysis <- function(species_name) {
 # Get the species data
 species_data <- data %>%
```

```
  filter(especie == species_name) %>%
  mutate(across(c(dap, tree_rich, tree_abu), ~ as.numeric(gsub(",", ".", .x))))

 # Select site-level habitat covariates
 site_covs <- species_data %>%
  select(dap, tree_rich, tree_abu)

 # Calculate mean values for observation covariates across surveys
 obs_covs <- species_data %>%
  rowwise() %>%
  mutate(
   ah_mean = mean(c_across(starts_with("ah")), na.rm = TRUE),
   moon_mean = mean(c_across(starts_with("moon")), na.rm = TRUE)
  ) %>%
  ungroup() %>%
  select(ah_mean, moon_mean) %>%
  mutate(across(everything(), ~ as.numeric(gsub(",", ".", .))))

 # Combine all covariates for correlation assessment
 all_covs <- bind_cols(site_covs, obs_covs)

 # Remove any rows with missing values for correlation analysis
 all_covs_complete <- all_covs %>% filter(complete.cases(.))

 # Run correlation analysis using robust methods
 cor_results <- correlation::correlation(all_covs_complete)

 return(list(
  correlation_matrix = cor_results,
  data = all_covs_complete
 ))
}

# Execute correlation analysis for both focal species
cat("=== CORRELATION ANALYSIS - P. caryophyllaceus ===\n")
cor_ca <- correlation_analysis("caryophyllaceus")
print(cor_ca$correlation_matrix)

cat("\n=== CORRELATION ANALYSIS - P. cruentus ===\n")
cor_cru <- correlation_analysis("cruentus")
print(cor_cru$correlation_matrix)

#------------------------------------------------------------------------------
# FUNCTION: CREATE CORRELATION VISUALIZATION FOR PUBLICATION
#
# Generates correlation matrices using hierarchical clustering to visualize
# relationships among ecological covariates. Helps identify correlated variable
# groups that may affect model selection.
```

```
#
# Parameters:
#   cor_data: Output from correlation_analysis function
#   species_name: Character string for plot title
#------------------------------------------------------------------------------

create_correlation_plot <- function(cor_data, species_name) {
 # Extract correlation matrix from the ecological data
 corr_matrix <- cor(cor_data$data, use = "complete.obs")

 # Create meaningful ecological variable names for visualization
 colnames(corr_matrix) <- rownames(corr_matrix) <- c(
  "DAP", "Tree Richness", "Tree Abundance",
  "Absolute Humidity", "Moon"
 )

 # Create correlation plot using hierarchical clustering
 corrplot(corr_matrix,
        method = "color",
        type = "lower",
        order = "hclust",
        tl.col = "black",
        tl.srt = 45,
        title = paste("Variable Correlations -", species_name),
        mar = c(0, 0, 2, 0))
}

# Generate correlation plots for both species
par(mfrow = c(1, 2))
create_correlation_plot(cor_ca, "P. caryophyllaceus")
create_correlation_plot(cor_cru, "P. cruentus")
par(mfrow = c(1, 1))

# Alternative ggplot2 version for publication flexibility
create_correlation_plot_gg <- function(cor_data, species_name) {
 corr_matrix <- cor(cor_data$data, use = "complete.obs")

 colnames(corr_matrix) <- rownames(corr_matrix) <- c(
  "DAP", "Tree Richness", "Tree Abundance",
  "Absolute Humidity", "Moon"
 )

 ggcorrplot(corr_matrix,
         method = "circle",
         type = "lower",
         lab = TRUE,
         lab_size = 3,
         colors = c("#6D9EC1", "white", "#E46726"),
```

```
        title = paste("Variable Correlations -", species_name)) +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"),
      axis.text.x = element_text(angle = 45, hjust = 1))
}

# Create and save ggplot2-style correlation plots
cor_plot_ca <- create_correlation_plot_gg(cor_ca, "P. caryophyllaceus")
cor_plot_cru <- create_correlation_plot_gg(cor_cru, "P. cruentus")

print(cor_plot_ca)
print(cor_plot_cru)

# Save correlation plots for publication
ggsave("correlation_caryophyllaceus.png", cor_plot_ca, width = 8, height = 6, dpi = 300)
ggsave("correlation_cruentus.png", cor_plot_cru, width = 8, height = 6, dpi = 300)

#==============================================================================
# OCCUPANCY MODEL IMPLEMENTATION AND SELECTION
#==============================================================================

# Prepare unmarked data for both focal species
caryophyllaceus_data <- prepare_species_data("caryophyllaceus")
cruentus_data <- prepare_species_data("cruentus")

c <- caryophyllaceus_data$umf
cruentus_umf <- cruentus_data$umf

# Store original values for ecological interpretation of predictions
c_original <- caryophyllaceus_data$original_values
cruentus_original <- cruentus_data$original_values

#------------------------------------------------------------------------------
# FUNCTION: AUTOMATED MODEL SELECTION USING AIC FRAMEWORK
#
# Implements a comprehensive model selection approach by fitting all combinations
# of detection and occupancy covariates. Uses Akaike Information Criterion (AIC)
# for model comparison and ranks models by parsimony and fit.
#
# Parameters:
#   umf: unmarkedFrame for occupancy modeling
#   max_models: Optional limit on number of top models to return
#
# Returns:
#   List containing ranked model results and fitted model objects
#------------------------------------------------------------------------------

model_selection_unmarked <- function(umf, max_models = NULL) {
```

```
# Define all biologically plausible combinations of covariates
det_forms <- c(
  "~1", "~ah", "~moon", "~ah + moon"
)

occ_forms <- c(
  "~1", "~dap", "~tree_rich", "~tree_abu",
  "~dap + tree_rich", "~dap + tree_abu", "~tree_rich + tree_abu",
  "~dap + tree_rich + tree_abu"
)

# Fit all models and store results
results <- list()
aic_values <- c()
formulas <- c()
k_values <- c()  # Number of parameters for AIC calculation

cat("Fitting", length(det_forms) * length(occ_forms), "model combinations...\n")

model_count <- 0
for(i in 1:length(det_forms)) {
  for(j in 1:length(occ_forms)) {
    formula_str <- paste(det_forms[i], occ_forms[j], sep = " ")
    model_count <- model_count + 1

    cat("Model", model_count, "of", length(det_forms) * length(occ_forms), ":", formula_str)

    tryCatch({
      mod <- occu(as.formula(formula_str), data = umf)

      # Extract AIC using unmarked's method
      aic_val <- mod@AIC

      results[[formula_str]] <- mod
      aic_values <- c(aic_values, aic_val)
      formulas <- c(formulas, formula_str)
      k_values <- c(k_values, length(coef(mod)))  # Store number of parameters

      cat(" - AIC:", round(aic_val, 2), "\n")

    }, error = function(e) {
      cat(" - ERROR:", e$message, "\n")
    })
  }
}

# Create results dataframe using base R for reliability
if (length(aic_values) > 0) {
```

```
  model_results <- data.frame(
   formula = formulas,
   k = k_values,
   AIC = aic_values,
   stringsAsFactors = FALSE
  )

  # Order by AIC (lowest AIC indicates best model)
  model_results <- model_results[order(model_results$AIC), ]

  # Calculate delta AIC and Akaike weights for model comparison
  model_results$delta_AIC <- model_results$AIC - min(model_results$AIC)
  model_results$weight <- exp(-0.5 * model_results$delta_AIC) / sum(exp(-0.5 *
model_results$delta_AIC))
  model_results$CumW <- cumsum(model_results$weight)

  # Reset row names for clean output
  rownames(model_results) <- NULL

 } else {
  stop("No models were successfully fitted. Check your data and formulas.")
 }

 # Limit to top models if specified
 if(!is.null(max_models)) {
  model_results <- model_results[1:min(max_models, nrow(model_results)), ]
 }

 return(list(
  results = model_results,
  models = results
 ))
}

# Execute model selection for both focal species
cat("=== RUNNING MODEL SELECTION FOR P. caryophyllaceus ===\n")
selection_ca <- model_selection_unmarked(c)

# Extract best model using AIC criterion
if (nrow(selection_ca$results) > 0) {
 best_formula_ca <- selection_ca$results$formula[1]
 best_ca <- selection_ca$models[[best_formula_ca]]
 cat("Best model for P. caryophyllaceus:", best_formula_ca, "\n")
} else {
 stop("No successful models for P. caryophyllaceus")
}
# Display top models with delta AIC ≤ 2 for P. caryophyllaceus
cat("\n=== TOP MODELS FOR P. caryophyllaceus (delta AIC ≤ 2) ===\n")
```

```
top_models_ca <- selection_ca$results %>%
  filter(delta_AIC <= 2) %>%
  select(formula, AIC, delta_AIC, weight)

print(top_models_ca)

# Cruentus model selection
cat("\n=== RUNNING MODEL SELECTION FOR P. cruentus ===\n")
selection_cru <- model_selection_unmarked(cruentus_umf)

if (nrow(selection_cru$results) > 0) {
  best_formula_cru <- selection_cru$results$formula[1]
  best_cru <- selection_cru$models[[best_formula_cru]]
  cat("Best model for P. cruentus:", best_formula_cru, "\n")
} else {
  stop("No successful models for P. cruentus")
}
# Display top models with delta AIC ≤ 2 for P. cruentus
cat("\n=== TOP MODELS FOR P. cruentus (delta AIC ≤ 2) ===\n")
top_models_cru <- selection_cru$results %>%
  filter(delta_AIC <= 2) %>%
  select(formula, AIC, delta_AIC, weight)

print(top_models_cru)


#============================================================================
# ECOLOGICAL PREDICTION AND VISUALIZATION
#============================================================================

#----------------------------------------------------------------------------
# FUNCTION: CREATE PREDICTION PLOTS WITH ECOLOGICAL SCALES
#
# Generates partial dependence plots showing how detection and occupancy
# probabilities vary with environmental covariates. Back-transforms scaled
# covariates to original ecological units for meaningful interpretation.
#
# Parameters:
#   best_model: Top-ranked occupancy model from AIC selection
#   best_formula: Formula of the best model
#   umf: unmarkedFrame used for modeling
#   original_values: Means and SDs for back-transformation
#   species_name: Character string for plot labeling
#
# Returns:
#   List of ggplot objects showing covariate effects
#----------------------------------------------------------------------------
```

```
create_prediction_plots_original_scale <- function(best_model, best_formula, umf, original_values,
species_name) {

 # Convert scaled ranges back to original ecological scales
 ah_range_scaled <- seq(min(umf@obsCovs$ah, na.rm = TRUE),
                        max(umf@obsCovs$ah, na.rm = TRUE),
                        length.out = 100)
 ah_range_original <- ah_range_scaled * original_values$ah["sd"] + original_values$ah["mean"]

 moon_range_scaled <- seq(min(umf@obsCovs$moon, na.rm = TRUE),
                          max(umf@obsCovs$moon, na.rm = TRUE),
                          length.out = 100)
 moon_range_original <- moon_range_scaled * original_values$moon["sd"] +
original_values$moon["mean"]

 # For DAP (tree diameter at breast height)
 dap_range_scaled <- seq(min(umf@siteCovs$dap, na.rm = TRUE),
                         max(umf@siteCovs$dap, na.rm = TRUE),
                         length.out = 100)
 dap_range_original <- dap_range_scaled * original_values$dap["sd"] +
original_values$dap["mean"]

 plots_list <- list()

 # ABSOLUTE HUMIDITY EFFECT ON DETECTION PROBABILITY
 if(grepl("ah", best_formula)) {
  new_data_ah <- data.frame(
   ah = ah_range_scaled,
   moon = ifelse(grepl("moon", best_formula), mean(umf@obsCovs$moon, na.rm = TRUE), 0)
  )

  pred_ah <- predict(best_model, type = 'det', newdata = new_data_ah, appendData = TRUE)
  pred_ah$ah_original <- ah_range_original

  p_ah <- ggplot(pred_ah, aes(ah_original, Predicted)) +
   geom_ribbon(aes(ymin = lower, ymax = upper), fill = "#1B9E77", alpha = 0.3) +
   geom_line(color = "#1B9E77", linewidth = 1.5) +
   scale_y_continuous(limits = c(0, 1)) +
   labs(
    x = "Absolute Humidity (g/m^3)",
    y = "Detection Probability"
   ) +
   theme_minimal() +
   theme(
    axis.title.x = element_text(size = 16, face = "bold", margin = margin(t = 10)),
    axis.title.y = element_text(size = 16, face = "bold", margin = margin(r = 10)),
    axis.text.x = element_text(size = 14, color = "black"),
    axis.text.y = element_text(size = 14, color = "black"),
```

```
      panel.grid.minor = element_blank(),
      panel.grid.major = element_line(color = "grey90", linewidth = 0.5),
      plot.background = element_rect(fill = "white", color = NA)
    )

  plots_list[["absolute_humidity"]] <- p_ah
 }

 # LUNAR CYCLE EFFECT ON DETECTION PROBABILITY
 if(grepl("moon", best_formula)) {
  new_data_moon <- data.frame(
   ah = ifelse(grepl("ah", best_formula), mean(umf@obsCovs$ah, na.rm = TRUE), 0),
   moon = moon_range_scaled
  )

  pred_moon <- predict(best_model, type = 'det', newdata = new_data_moon, appendData = TRUE)
  pred_moon$moon_original <- moon_range_original

  # Create ecologically meaningful moon phase labels
  moon_breaks <- seq(min(pred_moon$moon_original), max(pred_moon$moon_original),
length.out = 5)
  moon_labels <- c("New Moon", "First Quarter", "Waxing Gibbous", "Waning Gibbous", "Full Moon")

  p_moon <- ggplot(pred_moon, aes(moon_original, Predicted)) +
   geom_ribbon(aes(ymin = lower, ymax = upper), fill = "#7570B3", alpha = 0.3) +
   geom_line(color = "#7570B3", linewidth = 1.5) +
   scale_y_continuous(limits = c(0, 1)) +
   scale_x_continuous(
    name = "Moon Phase",
    breaks = moon_breaks,
    labels = moon_labels
   ) +
   labs(y = "Detection Probability") +
   theme_minimal() +
   theme(
    axis.title.x = element_text(size = 16, face = "bold", margin = margin(t = 10)),
    axis.title.y = element_text(size = 16, face = "bold", margin = margin(r = 10)),
    axis.text.x = element_text(size = 12, color = "black", angle = 45, hjust = 1),
    axis.text.y = element_text(size = 14, color = "black"),
    panel.grid.minor = element_blank(),
    panel.grid.major = element_line(color = "grey90", linewidth = 0.5),
    plot.background = element_rect(fill = "white", color = NA)
   )

  plots_list[["moon"]] <- p_moon
 }

 # TREE DIAMETER (DAP) EFFECT ON OCCUPANCY PROBABILITY
```

```
  if(grepl("dap", best_formula)) {
    new_data_dap <- data.frame(
      dap = dap_range_scaled,
      tree_rich = ifelse(grepl("tree_rich", best_formula), mean(umf@siteCovs$tree_rich, na.rm =
TRUE), 0),
      tree_abu = ifelse(grepl("tree_abu", best_formula), mean(umf@siteCovs$tree_abu, na.rm =
TRUE), 0)
    )

    pred_dap <- predict(best_model, type = 'state', newdata = new_data_dap, appendData = TRUE)
    pred_dap$dap_original <- dap_range_original

    p_dap <- ggplot(pred_dap, aes(dap_original, Predicted)) +
      geom_ribbon(aes(ymin = lower, ymax = upper), fill = "#33a02c", alpha = 0.3) +
      geom_line(color = "#33a02c", linewidth = 1.5) +
      scale_y_continuous(limits = c(0, 1)) +
      labs(
        x = "Tree Diameter at Breast Height (DBH, cm)",
        y = "Occupancy Probability"
      ) +
      theme_minimal() +
      theme(
        axis.title.x = element_text(size = 16, face = "bold", margin = margin(t = 10)),
        axis.title.y = element_text(size = 16, face = "bold", margin = margin(r = 10)),
        axis.text.x = element_text(size = 14, color = "black"),
        axis.text.y = element_text(size = 14, color = "black"),
        panel.grid.minor = element_blank(),
        panel.grid.major = element_line(color = "grey90", linewidth = 0.5),
        plot.background = element_rect(fill = "white", color = NA)
      )

    plots_list[["dap"]] <- p_dap
  }

  return(plots_list)
}

# Generate ecological prediction plots for both species
plots_ca <- create_prediction_plots_original_scale(best_ca, best_formula_ca, c, c_original,
"Pristimantis caryophyllaceus")
plots_cru <- create_prediction_plots_original_scale(best_cru, best_formula_cru, cruentus_umf,
cruentus_original, "Pristimantis cruentus")

#------------------------------------------------------------------------------
# FUNCTION: CREATE INTERSPECIFIC COMPARISON FIGURES
#
# Generates side-by-side comparison plots to visualize differences in ecological
# responses between the two Pristimantis species. Facilitates comparative
```

```
# ecological inference.
#
# Parameters:
#   plots_ca: Prediction plots for P. caryophyllaceus
#   plots_cru: Prediction plots for P. cruentus
#   covariate_name: Specific covariate to compare
#   title: Plot title
#------------------------------------------------------------------------------

create_comparison_figure <- function(plots_ca, plots_cru, covariate_name, title) {
  if(covariate_name %in% names(plots_ca) && covariate_name %in% names(plots_cru)) {

    p_ca <- plots_ca[[covariate_name]] +
      labs(subtitle = expression(italic("Pristimantis caryophyllaceus"))) +
      theme(plot.subtitle = element_text(face = "italic", hjust = 0.5, size = 14))

    p_cru <- plots_cru[[covariate_name]] +
      labs(subtitle = expression(italic("Pristimantis cruentus"))) +
      theme(plot.subtitle = element_text(face = "italic", hjust = 0.5, size = 14))

    combined <- p_ca + p_cru +
      plot_layout(ncol = 2) +
      plot_annotation(
        title = title,
        theme = theme(
          plot.title = element_text(hjust = 0.5, face = "bold", size = 18)
        )
      )

    return(combined)
  }
  return(NULL)
}

# Create and save interspecific comparison figures
if("moon" %in% names(plots_ca) && "moon" %in% names(plots_cru)) {
  moon_comparison <- create_comparison_figure(plots_ca, plots_cru, "moon", "Effect of Moon
Phase on Detection Probability")
  ggsave("moon_comparison.png", moon_comparison, width = 16, height = 8, dpi = 300, bg =
"white")
}

if("absolute_humidity" %in% names(plots_ca) && "absolute_humidity" %in% names(plots_cru)) {
  ah_comparison <- create_comparison_figure(plots_ca, plots_cru, "absolute_humidity", "Effect of
Absolute Humidity on Detection Probability")
  ggsave("absolute_humidity_comparison.png", ah_comparison, width = 16, height = 8, dpi = 300, bg
= "white")
}
```

```
if("dap" %in% names(plots_ca) && "dap" %in% names(plots_cru)) {
 dap_comparison <- create_comparison_figure(plots_ca, plots_cru, "dap", "Effect of Tree Diameter
on Occupancy Probability")
 ggsave("dap_comparison.png", dap_comparison, width = 16, height = 8, dpi = 300, bg = "white")
}

# Save individual species plots for publication
for(plot_name in names(plots_ca)) {
 filename <- paste0("caryophyllaceus_", plot_name, ".png")
 ggsave(filename, plots_ca[[plot_name]], width = 10, height = 8, dpi = 300, bg = "white")
}

for(plot_name in names(plots_cru)) {
 filename <- paste0("cruentus_", plot_name, ".png")
 ggsave(filename, plots_cru[[plot_name]], width = 10, height = 8, dpi = 300, bg = "white")
}

#==============================================================================
# RELATIVE IMPORTANCE ANALYSIS
#==============================================================================

#------------------------------------------------------------------------------
# FUNCTION: CALCULATE RELATIVE IMPORTANCE OF ECOLOGICAL COVARIATES
#
# Computes relative importance values by summing Akaike weights across all
# models containing each covariate. Provides inference about which ecological
# factors most strongly influence occupancy and detection.
#
# Parameters:
#   selection_results: Output from model_selection_unmarked function
#
# Returns:
#   Dataframe with covariates ranked by relative importance
#------------------------------------------------------------------------------

calculate_relative_importance <- function(selection_results) {
 # Extract model results
 model_results <- selection_results$results

 # Define all ecological covariates considered
 all_covariates <- c("ah", "moon", "dap", "tree_rich", "tree_abu")

 # Calculate importance for each covariate
 importance_list <- list()

 for(covariate in all_covariates) {
  # Find models that contain this covariate
```

```
  if(covariate %in% c("ah", "moon")) {
    # Detection covariates
    models_with_covariate <- grepl(covariate, model_results$formula)
  } else {
    # Occupancy covariates
    models_with_covariate <- grepl(covariate, model_results$formula)
  }

  # Sum Akaike weights of models containing this covariate
  total_weight <- sum(model_results$weight[models_with_covariate])

  importance_list[[covariate]] <- data.frame(
    covariate = covariate,
    importance = total_weight
  )
 }

 # Combine all importance values and rank by importance
 importance_df <- bind_rows(importance_list) %>%
  arrange(desc(importance))

 return(importance_df)
}

# Calculate relative importance for both species
importance_ca <- calculate_relative_importance(selection_ca)
importance_cru <- calculate_relative_importance(selection_cru)

cat("✓ Relative importance data calculated\n")

#------------------------------------------------------------------------------
# FUNCTION: CLUSTERED RELATIVE IMPORTANCE PLOT
#
# Creates a grouped bar chart showing relative importance values for both
# species together, facilitating direct comparison of ecological drivers
# between species.
#
# Parameters:
#   importance_ca: Importance data for P. caryophyllaceus
#   importance_cru: Importance data for P. cruentus
#------------------------------------------------------------------------------

create_clustered_relimp_plot <- function(importance_ca, importance_cru) {
 # Combine both species' data
 combined_data <- bind_rows(
  importance_ca %>% mutate(species = "Pristimantis caryophyllaceus"),
  importance_cru %>% mutate(species = "Pristimantis cruentus")
 )
```

```
 # Define ecological color scheme
 colors <- c("ah" = "#35978f", "moon" = "#b0b0b0",
         "dap" = "#bf812d", "tree_rich" = "#66A61E", "tree_abu" = "#01665e")

 # Create meaningful ecological labels
 pretty_labels <- c(
  "ah" = "Absolute Humidity",
  "moon" = "Lunar cycle",
  "dap" = "Tree diameter (DBH)",
  "tree_rich" = "Tree richness",
  "tree_abu" = "Tree abundance"
 )

 combined_data <- combined_data %>%
  mutate(
   label = pretty_labels[covariate],
   species = factor(species, levels = c("Pristimantis caryophyllaceus", "Pristimantis cruentus")),
   # Short names for clustering
   species_short = ifelse(species == "Pristimantis caryophyllaceus", "P. caryophyllaceus", "P.
cruentus")
  )

 ggplot(combined_data, aes(x = importance, y = reorder(label, importance), fill = species_short)) +
  geom_col(position = position_dodge(0.8), width = 0.7, alpha = 0.8) +
  geom_text(aes(label = sprintf("%.2f", importance)),
         position = position_dodge(0.8),
         hjust = -0.2, size = 5, color = "black") +
  scale_fill_manual(values = c("P. caryophyllaceus" = "#1f78b4", "P. cruentus" = "#33a02c")) +
  scale_x_continuous(
   limits = c(0, 1.1),
   expand = expansion(mult = c(0, 0.05)),
   breaks = seq(0, 1, 0.2)
  ) +
  labs(
   x = "Relative Importance",
   y = NULL,
   title = "Relative Importance of Covariates in Occupancy Models",
   subtitle = "Based on model weights across all candidate models",
   fill = "Species"
  ) +
  theme_minimal() +
  theme(
   legend.position = "top",
   legend.title = element_text(face = "bold", size = 14),
   legend.text = element_text(face = "italic", size = 12),
   plot.title = element_text(hjust = 0.5, face = "bold", size = 20, margin = margin(b = 15)),
   plot.subtitle = element_text(hjust = 0.5, size = 16, color = "grey40", margin = margin(b = 20)),
```

```
    axis.title.x = element_text(face = "bold", size = 18, margin = margin(t = 15)),
    axis.text.y = element_text(face = "bold", size = 16, color = "black"),
    axis.text.x = element_text(size = 14),
    # Add black axis lines
    axis.line = element_line(color = "black", linewidth = 0.5),
    axis.ticks = element_line(color = "black"),
    panel.grid.major.y = element_blank(),
    panel.grid.minor.y = element_blank(),
    panel.grid.major.x = element_line(color = "grey90"),
    panel.grid.minor.x = element_blank(),
    panel.border = element_blank(),
    plot.background = element_rect(fill = "white", color = NA)
  ) +
  coord_cartesian(clip = "off")
}

# Create and save clustered relative importance plot
clustered_relimp <- create_clustered_relimp_plot(importance_ca, importance_cru)
ggsave("relative_importance_clustered.png", clustered_relimp,
     width = 18, height = 10, dpi = 300, bg = "white")

cat("✓ Clustered relative importance plot saved\n")

#------------------------------------------------------------------------------
# FUNCTION: COMBINED RELATIVE IMPORTANCE PLOT
#
# Creates side-by-side facet plot showing relative importance for both species
# separately but in a single figure for comparative assessment.
#
# Parameters:
#   importance_ca: Importance data for P. caryophyllaceus
#   importance_cru: Importance data for P. cruentus
#------------------------------------------------------------------------------

create_combined_relimp_plot <- function(importance_ca, importance_cru) {
 # Combine both species' data
 combined_data <- bind_rows(
  importance_ca %>% mutate(species = "Pristimantis caryophyllaceus"),
  importance_cru %>% mutate(species = "Pristimantis cruentus")
 )

 # Define ecological color scheme
 colors <- c("ah" = "#35978f", "moon" = "#b0b0b0",
         "dap" = "#bf812d", "tree_rich" = "#66A61E", "tree_abu" = "#01665e")

 # Create meaningful ecological labels
 pretty_labels <- c(
  "ah" = "Absolute Humidity",
```

```
  "moon" = "Lunar cycle",
  "dap" = "Tree diameter (DBH)",
  "tree_rich" = "Tree richness",
  "tree_abu" = "Tree abundance"
)

combined_data <- combined_data %>%
  mutate(
    label = pretty_labels[covariate],
    species = factor(species, levels = c("Pristimantis caryophyllaceus", "Pristimantis cruentus"))
  )

ggplot(combined_data, aes(x = importance, y = reorder(label, importance))) +
  geom_col(aes(fill = covariate), width = 0.7, alpha = 0.8, position = "dodge") +
  geom_text(aes(label = sprintf("%.2f", importance)),
            position = position_dodge(width = 0.7),
            hjust = -0.2, size = 5, color = "black") +
  scale_fill_manual(values = colors) +
  scale_x_continuous(
    limits = c(0, 1.1),
    expand = expansion(mult = c(0, 0.05)),
    breaks = seq(0, 1, 0.2)
  ) +
  facet_wrap(~ species, ncol = 2) +
  labs(
    x = "Relative Importance",
    y = NULL,
    title = "Relative Importance of Covariates in Occupancy Models",
    subtitle = "Based on model weights across all candidate models"
  ) +
  theme_minimal() +
  theme(
    legend.position = "none",
    plot.title = element_text(hjust = 0.5, face = "bold", size = 20, margin = margin(b = 15)),
    plot.subtitle = element_text(hjust = 0.5, size = 16, color = "grey40", margin = margin(b = 20)),
    axis.title.x = element_text(face = "bold", size = 18, margin = margin(t = 15)),
    axis.text.y = element_text(face = "bold", size = 16, color = "black"),
    axis.text.x = element_text(size = 14),
    strip.text = element_text(face = "italic", size = 16, color = "black"),
    # Add black axis lines
    axis.line = element_line(color = "black", linewidth = 0.5),
    axis.ticks = element_line(color = "black"),
    panel.grid.major.y = element_blank(),
    panel.grid.minor.y = element_blank(),
    panel.grid.major.x = element_line(color = "grey90"),
    panel.grid.minor.x = element_blank(),
    panel.border = element_blank(),
    plot.background = element_rect(fill = "white", color = NA)
```

```
  ) +
  coord_cartesian(clip = "off")
}

# Create and save combined relative importance plot
combined_relimp_single <- create_combined_relimp_plot(importance_ca, importance_cru)
ggsave("relative_importance_both_species.png", combined_relimp_single,
     width = 18, height = 10, dpi = 300, bg = "white")

cat("✓ Combined relative importance plot saved with proper font sizes\n")

#------------------------------------------------------------------------------
# FUNCTION: INDIVIDUAL SPECIES RELATIVE IMPORTANCE PLOTS
#
# Creates separate relative importance plots for each species with consistent
# color scheme for individual species assessment.
#
# Parameters:
#   importance_df: Importance data for a single species
#   species_name: Character string for plot title
#------------------------------------------------------------------------------

create_single_relimp_plot <- function(importance_df, species_name) {
 # Use consistent ecological color scheme
 colors <- c("ah" = "#35978f", "moon" = "#b0b0b0",
         "dap" = "#bf812d", "tree_rich" = "#66A61E", "tree_abu" = "#01665e")

 # Create meaningful ecological labels
 pretty_labels <- c(
  "ah" = "Absolute Humidity",
  "moon" = "Lunar cycle",
  "dap" = "Tree diameter (DBH)",
  "tree_rich" = "Tree richness",
  "tree_abu" = "Tree abundance"
 )

 importance_df <- importance_df %>%
  mutate(label = pretty_labels[covariate])

 ggplot(importance_df, aes(x = importance, y = reorder(label, importance))) +
  geom_col(aes(fill = covariate), width = 0.7, alpha = 0.8) +
  geom_text(aes(label = sprintf("%.2f", importance)),
         hjust = -0.2, size = 5, color = "black") +
  scale_fill_manual(values = colors) +
  scale_x_continuous(
   limits = c(0, 1.1),
   expand = expansion(mult = c(0, 0.05)),
   breaks = seq(0, 1, 0.2)
```

```
  ) +
  labs(
    x = "Relative Importance",
    y = NULL,
    title = species_name,
    subtitle = "Relative importance of covariates"
  ) +
  theme_minimal() +
  theme(
    legend.position = "none",
    plot.title = element_text(face = "italic", hjust = 0.5, size = 18, margin = margin(b = 10)),
    plot.subtitle = element_text(hjust = 0.5, size = 14, color = "grey40", margin = margin(b = 15)),
    axis.title.x = element_text(face = "bold", size = 16, margin = margin(t = 10)),
    axis.text.y = element_text(face = "bold", size = 14, color = "black"),
    axis.text.x = element_text(size = 12),
    # Add black axis lines
    axis.line = element_line(color = "black", linewidth = 0.5),
    axis.ticks = element_line(color = "black"),
    panel.grid.major.y = element_blank(),
    panel.grid.minor.y = element_blank(),
    panel.grid.major.x = element_line(color = "grey90"),
    panel.grid.minor.x = element_blank(),
    panel.border = element_blank(),
    plot.background = element_rect(fill = "white", color = NA)
  ) +
  coord_cartesian(clip = "off")
}

# Create individual species relative importance plots
relimp_ca <- create_single_relimp_plot(importance_ca, "Pristimantis caryophyllaceus")
relimp_cru <- create_single_relimp_plot(importance_cru, "Pristimantis cruentus")

# Save individual plots
ggsave("relative_importance_caryophyllaceus.png", relimp_ca,
       width = 10, height = 8, dpi = 300, bg = "white")
ggsave("relative_importance_cruentus.png", relimp_cru,
       width = 10, height = 8, dpi = 300, bg = "white")

cat("✓ Individual relative importance plots saved\n")

#==============================================================================
# RESULTS DISPLAY AND SUMMARY
#==============================================================================

# Display all generated plots in R graphics device
cat("\n=== QUICK DISPLAY OF ALL ECOLOGICAL PLOTS ===\n")

# Display individual species prediction plots
```

```
cat("\n--- Pristimantis caryophyllaceus ---\n")
for(plot_name in names(plots_ca)) {
  cat("Displaying:", plot_name, "\n")
  print(plots_ca[[plot_name]])
}

cat("\n--- Pristimantis cruentus ---\n")
for(plot_name in names(plots_cru)) {
  cat("Displaying:", plot_name, "\n")
  print(plots_cru[[plot_name]])
}

# Display comparison plots
cat("\n--- Interspecific Comparison Plots ---\n")
comparison_plots <- ls(pattern = "_comparison$")
for(plot_name in comparison_plots) {
  cat("Displaying:", plot_name, "\n")
  print(get(plot_name))
}

# Display relative importance plots
cat("\n--- Relative Importance Analysis ---\n")
if(exists("relimp_ca")) {
  cat("Displaying: P. caryophyllaceus Relative Importance\n")
  print(relimp_ca)
}
if(exists("relimp_cru")) {
  cat("Displaying: P. cruentus Relative Importance\n")
  print(relimp_cru)
}
if(exists("combined_relimp_single")) {
  cat("Displaying: Combined Relative Importance (Side by Side)\n")
  print(combined_relimp_single)
}
if(exists("clustered_relimp")) {
  cat("Displaying: Clustered Relative Importance (Both Species Together)\n")
  print(clustered_relimp)
}

#------------------------------------------------------------------------------
# ECOLOGICAL ANALYSIS SUMMARY
#
# Provides concise summary of key findings for both focal species, including
# best models, AIC values, and ecological interpretation of results.
#------------------------------------------------------------------------------

cat("\n=== ECOLOGICAL ANALYSIS SUMMARY ===\n")
cat("Pristimantis caryophyllaceus:\n")
```

```
cat("  Best model:", best_formula_ca, "\n")
cat("  AIC:", round(best_ca@AIC, 2), "\n")
cat("  Ecological drivers identified:", names(plots_ca), "\n\n")

cat("Pristimantis cruentus:\n")
cat("  Best model:", best_formula_cru, "\n")
cat("  AIC:", round(best_cru@AIC, 2), "\n")
cat("  Ecological drivers identified:", names(plots_cru), "\n\n")

cat("Comparative figures generated:\n")
if(!is.null(moon_comparison)) cat("- Lunar cycle effects on detection\n")
if(!is.null(ah_comparison)) cat("- Absolute humidity effects on detection\n")
if(!is.null(dap_comparison)) cat("- Tree diameter effects on occupancy\n")

cat("\nRelative importance analyses completed:\n")
cat("- Clustered bar chart (both species)\n")
cat("- Combined facet plot\n")
cat("- Individual species plots\n")

cat("\n✅ OCCUPANCY MODELING ANALYSIS COMPLETE\n")
cat("All ecological plots and analyses saved for publication.\n")
```

OCCUPANCY MODELING AND RELATIVE IMPORTANCE ANALYSIS FOR DIASPORUS

```
#==============================================================================
#
# This script performs single-season occupancy modeling for Diasporus diastema
# to evaluate environmental and sampling covariates affecting detection and
# occupancy probabilities. The analysis follows the single-species, single-season
# occupancy modeling framework of MacKenzie et al. (2002).
#
# Species: Diasporus diastema
# Study System: Tropical montane forest, Costa Rica
#
# Covariates:
# - Detection: absolute humidity, lunar cycle
# - Occupancy: tree diameter (DBH), tree richness, tree abundance
#
# Analysis includes:
# 1. Data preparation and correlation analysis
# 2. Model selection using AIC
# 3. Prediction plots with back-transformed scales
# 4. Relative importance of covariates
#
#==============================================================================

# Load required libraries for ecological analysis
library(tidyverse)     # Data manipulation and visualization
library(readxl)        # Excel file reading
library(unmarked)      # Occupancy modeling framework
library(patchwork)     # Plot arrangement and composition
library(showtext)      # Custom fonts for publication-quality figures
library(corrplot)      # Correlation matrix visualization
library(conflicted)    # Conflict resolution for function names
library(correlation)   # Advanced correlation analysis
library(ggcorrplot)    # ggplot2-based correlation plots

# Resolve common function conflicts in ecological analysis
conflicts_prefer(
  dplyr::filter,    # Prefer dplyr filter over stats
  dplyr::lag,       # Prefer dplyr lag over stats
  dplyr::select,    # Prefer dplyr select over MASS
  dplyr::mutate,    # Ensure dplyr mutate is used
  dplyr::arrange,   # Ensure dplyr arrange is used
  dplyr::summarize  # Ensure dplyr summarize is used
)

# Configure publication-quality typography
```

```
showtext_auto()
font_add_google("Lato", "lato")

#==============================================================================
# ABSOLUTE HUMIDITY CALCULATION FUNCTION
#==============================================================================

# Function to calculate absolute humidity (g/m^3) from temperature (°C) and relative humidity (%)
calculate_absolute_humidity <- function(temp_c, rh_percent) {
 # Convert temperature to Kelvin
 temp_k <- temp_c + 273.15

 # Saturation vapor pressure (Tetens equation) in hPa
 es_hpa <- 6.112 * exp((17.67 * temp_c) / (temp_c + 243.5))

 # Actual vapor pressure in hPa
 ea_hpa <- (rh_percent / 100) * es_hpa

 # Absolute humidity in g/m^3
 absolute_humidity <- (216.7 * ea_hpa) / temp_k

 return(absolute_humidity)
}

#==============================================================================
# DATA IMPORT AND PREPARATION
#==============================================================================

# Read ecological survey data from Excel file
file_path <- "C:/Users/Victor Acosta/Desktop/ecologyandevolution/especies_modelos_RM.xlsx"
data <- read_excel(file_path)

# Calculate absolute humidity for each temperature/humidity pair
data_with_ah <- data %>%
 # Convert to numeric and clean data
 mutate(across(starts_with("temp"), ~as.numeric(gsub(",", ".", .)))) %>%
 mutate(across(starts_with("hum"), ~as.numeric(gsub(",", ".", .)))) %>%
 # Calculate absolute humidity for each temp/hum pair
 mutate(across(starts_with("temp"),
          ~calculate_absolute_humidity(., get(paste0("hum", str_sub(cur_column(), 5)))),
          .names = "ah_{str_sub(.col, 5)}"))

# Use the data with absolute humidity for analysis
data <- data_with_ah

#------------------------------------------------------------------------------
# FUNCTION: PREPARE UNMARKED DATA FOR SINGLE-SPECIES OCCUPANCY ANALYSIS
#
```

```
# This function formats detection/non-detection data into the unmarked framework
# for occupancy modeling. It handles scaling of continuous covariates and
# preserves original values for back-transformation of predictions.
#
# Parameters:
#   species_name: Character string of target species name
#
# Returns:
#   List containing:
#     - umf: unmarkedFrame for occupancy modeling
#     - original_values: Means and SDs for back-transformation
#------------------------------------------------------------------------------

prepare_species_data <- function(species_name) {
 # Filter data for focal species
 species_data <- data %>% filter(especie == species_name)

 # Store original values before scaling for ecological interpretation
 original_values <- list()

 # Extract detection history (oc1 to oc10) - binary detection matrix
 y <- species_data %>%
  select(starts_with("oc")) %>%
  as.matrix()

 # Extract and clean site covariates (habitat characteristics)
 siteCovs <- species_data %>%
  select(area, dap, tree_rich, tree_abu) %>%
  mutate(across(c(dap, tree_rich, tree_abu), ~ as.numeric(gsub(",", ".", .x))))

 # Extract and scale observation covariates (absolute humidity and moon)
 ah_data <- species_data %>%
  select(starts_with("ah")) %>%
  mutate(across(everything(), ~as.numeric(gsub(",", ".", .)))) %>%
  as.matrix()

 moon_data <- species_data %>%
  select(starts_with("moon")) %>%
  mutate(across(everything(), ~as.numeric(gsub(",", ".", .)))) %>%
  as.matrix()

 # Store original ecological values for meaningful interpretation
 original_values$ah <- c(mean = mean(ah_data, na.rm = TRUE),
                         sd = sd(ah_data, na.rm = TRUE))
 original_values$moon <- c(mean = mean(moon_data, na.rm = TRUE),
                           sd = sd(moon_data, na.rm = TRUE))
 original_values$dap <- c(mean = mean(siteCovs$dap, na.rm = TRUE),
                          sd = sd(siteCovs$dap, na.rm = TRUE))
```

```
 obsCovs <- list(ah = ah_data, moon = moon_data)

 # Create unmarked frame for occupancy analysis
 umf <- unmarkedFrameOccu(y = y, siteCovs = siteCovs, obsCovs = obsCovs)

 # Scale covariates for model convergence and comparison
 umf@siteCovs$dap <- scale(umf@siteCovs$dap)
 umf@siteCovs$tree_rich <- scale(umf@siteCovs$tree_rich)
 umf@siteCovs$tree_abu <- scale(umf@siteCovs$tree_abu)

 umf@obsCovs$ah <- scale(umf@obsCovs$ah)
 umf@obsCovs$moon <- scale(umf@obsCovs$moon)

 return(list(umf = umf, original_values = original_values))
}

#==============================================================================
# CORRELATION ANALYSIS AMONG ECOLOGICAL COVARIATES
#==============================================================================

#------------------------------------------------------------------------------
# FUNCTION: CORRELATION ANALYSIS FOR MULTICOLLINEARITY ASSESSMENT
#
# Evaluates pairwise correlations among environmental covariates to identify
# potential multicollinearity issues before occupancy modeling. High correlations
# (>0.7) may require careful model selection to avoid overparameterization.
#
# Parameters:
#   species_name: Character string of target species name
#
# Returns:
#   List containing correlation matrix and complete cases data
#------------------------------------------------------------------------------

correlation_analysis <- function(species_name) {
 # Get the species data
 species_data <- data %>%
  filter(especie == species_name) %>%
  mutate(across(c(dap, tree_rich, tree_abu), ~ as.numeric(gsub(",", ".", .x))))

 # Select site-level habitat covariates
 site_covs <- species_data %>%
  select(dap, tree_rich, tree_abu)

 # Calculate mean values for observation covariates across surveys
 obs_covs <- species_data %>%
  rowwise() %>%
```

```
    mutate(
      ah_mean = mean(c_across(starts_with("ah")), na.rm = TRUE),
      moon_mean = mean(c_across(starts_with("moon")), na.rm = TRUE)
    ) %>%
    ungroup() %>%
    select(ah_mean, moon_mean) %>%
    mutate(across(everything(), ~ as.numeric(gsub(",", ".", .))))

  # Combine all covariates for correlation assessment
  all_covs <- bind_cols(site_covs, obs_covs)

  # Remove any rows with missing values for correlation analysis
  all_covs_complete <- all_covs %>% filter(complete.cases(.))

  # Run correlation analysis using robust methods
  cor_results <- correlation::correlation(all_covs_complete)

  return(list(
    correlation_matrix = cor_results,
    data = all_covs_complete
  ))
}

# Execute correlation analysis for Diasporus diastema
cat("=== CORRELATION ANALYSIS - Diasporus diastema ===\n")
cor_diastema <- correlation_analysis("diastema")
print(cor_diastema$correlation_matrix)

#------------------------------------------------------------------------------
# FUNCTION: CREATE CORRELATION VISUALIZATION FOR PUBLICATION
#
# Generates correlation matrices using hierarchical clustering to visualize
# relationships among ecological covariates. Helps identify correlated variable
# groups that may affect model selection.
#
# Parameters:
#   cor_data: Output from correlation_analysis function
#   species_name: Character string for plot title
#------------------------------------------------------------------------------

create_correlation_plot <- function(cor_data, species_name) {
  # Extract correlation matrix from the ecological data
  corr_matrix <- cor(cor_data$data, use = "complete.obs")

  # Create meaningful ecological variable names for visualization
  colnames(corr_matrix) <- rownames(corr_matrix) <- c(
    "DAP", "Tree Richness", "Tree Abundance",
    "Absolute Humidity", "Moon"
```

```
  )

  # Create correlation plot using hierarchical clustering
  corrplot(corr_matrix,
           method = "color",
           type = "lower",
           order = "hclust",
           tl.col = "black",
           tl.srt = 45,
           title = paste("Variable Correlations -", species_name),
           mar = c(0, 0, 2, 0))
}

# Generate correlation plot for Diasporus diastema
create_correlation_plot(cor_diastema, "Diasporus diastema")

# Alternative ggplot2 version for publication flexibility
create_correlation_plot_gg <- function(cor_data, species_name) {
  corr_matrix <- cor(cor_data$data, use = "complete.obs")

  colnames(corr_matrix) <- rownames(corr_matrix) <- c(
    "DAP", "Tree Richness", "Tree Abundance",
    "Absolute Humidity", "Moon"
  )

  ggcorrplot(corr_matrix,
             method = "circle",
             type = "lower",
             lab = TRUE,
             lab_size = 3,
             colors = c("#6D9EC1", "white", "#E46726"),
             title = paste("Variable Correlations -", species_name)) +
    theme(plot.title = element_text(hjust = 0.5, face = "bold"),
          axis.text.x = element_text(angle = 45, hjust = 1))
}

# Create and save ggplot2-style correlation plot
cor_plot_diastema <- create_correlation_plot_gg(cor_diastema, "Diasporus diastema")
print(cor_plot_diastema)

# Save correlation plot for publication
ggsave("correlation_diastema.png", cor_plot_diastema, width = 8, height = 6, dpi = 300)

#==============================================================================
# OCCUPANCY MODEL IMPLEMENTATION AND SELECTION
#==============================================================================

# Prepare unmarked data for Diasporus diastema
```

```
diastema_data <- prepare_species_data("diastema")
d <- diastema_data$umf

# Store original values for ecological interpretation of predictions
diastema_original <- diastema_data$original_values

#------------------------------------------------------------------------------
# FUNCTION: AUTOMATED MODEL SELECTION USING AIC FRAMEWORK
#
# Implements a comprehensive model selection approach by fitting all combinations
# of detection and occupancy covariates. Uses Akaike Information Criterion (AIC)
# for model comparison and ranks models by parsimony and fit.
#
# Parameters:
#   umf: unmarkedFrame for occupancy modeling
#   max_models: Optional limit on number of top models to return
#
# Returns:
#   List containing ranked model results and fitted model objects
#------------------------------------------------------------------------------

model_selection_unmarked <- function(umf, max_models = NULL) {

 # Define all biologically plausible combinations of covariates
 det_forms <- c(
  "~1", "~ah", "~moon", "~ah + moon"
 )

 occ_forms <- c(
  "~1", "~dap", "~tree_rich", "~tree_abu",
  "~dap + tree_rich", "~dap + tree_abu", "~tree_rich + tree_abu",
  "~dap + tree_rich + tree_abu"
 )

 # Fit all models and store results
 results <- list()
 aic_values <- c()
 formulas <- c()
 k_values <- c()  # Number of parameters for AIC calculation

 cat("Fitting", length(det_forms) * length(occ_forms), "model combinations...\n")

 model_count <- 0
 for(i in 1:length(det_forms)) {
  for(j in 1:length(occ_forms)) {
   formula_str <- paste(det_forms[i], occ_forms[j], sep = " ")
   model_count <- model_count + 1
```

```
    cat("Model", model_count, "of", length(det_forms) * length(occ_forms), ":", formula_str)

    tryCatch({
      mod <- occu(as.formula(formula_str), data = umf)

      # Extract AIC using unmarked's method
      aic_val <- mod@AIC

      results[[formula_str]] <- mod
      aic_values <- c(aic_values, aic_val)
      formulas <- c(formulas, formula_str)
      k_values <- c(k_values, length(coef(mod)))  # Store number of parameters

      cat(" - AIC:", round(aic_val, 2), "\n")

    }, error = function(e) {
      cat(" - ERROR:", e$message, "\n")
    })
  }
 }

 # Create results dataframe using base R for reliability
 if (length(aic_values) > 0) {
  model_results <- data.frame(
   formula = formulas,
   k = k_values,
   AIC = aic_values,
   stringsAsFactors = FALSE
  )

  # Order by AIC (lowest AIC indicates best model)
  model_results <- model_results[order(model_results$AIC), ]

  # Calculate delta AIC and Akaike weights for model comparison
  model_results$delta_AIC <- model_results$AIC - min(model_results$AIC)
  model_results$weight <- exp(-0.5 * model_results$delta_AIC) / sum(exp(-0.5 *
model_results$delta_AIC))
  model_results$CumW <- cumsum(model_results$weight)

  # Reset row names for clean output
  rownames(model_results) <- NULL

 } else {
  stop("No models were successfully fitted. Check your data and formulas.")
 }

 # Limit to top models if specified
 if(!is.null(max_models)) {
```

```
  model_results <- model_results[1:min(max_models, nrow(model_results)), ]
 }

 return(list(
  results = model_results,
  models = results
 ))
}

# Execute model selection for Diasporus diastema
cat("=== RUNNING MODEL SELECTION FOR Diasporus diastema ===\n")
selection_diastema <- model_selection_unmarked(d)

# Extract best model using AIC criterion
if (nrow(selection_diastema$results) > 0) {
 best_formula_diastema <- selection_diastema$results$formula[1]
 best_diastema <- selection_diastema$models[[best_formula_diastema]]
 cat("Best model for Diasporus diastema:", best_formula_diastema, "\n")
 cat("AIC:", round(best_diastema@AIC, 2), "\n")
} else {
 stop("No successful models for Diasporus diastema")
}

# Display top models
cat("\n=== TOP MODELS FOR Diasporus diastema ===\n")
print(head(selection_diastema$results, 10))

#-------------------------------------------------------------------------------
# FUNCTION: CREATE PREDICTION PLOTS WITH ECOLOGICAL SCALES
#
# Generates partial dependence plots showing how detection and occupancy
# probabilities vary with environmental covariates. Back-transforms scaled
# covariates to original ecological units for meaningful interpretation.
# Only creates plots for covariates that are actually in the best model.
#
# Parameters:
#   best_model: Top-ranked occupancy model from AIC selection
#   best_formula: Formula of the best model
#   umf: unmarkedFrame used for modeling
#   original_values: Means and SDs for back-transformation
#   species_name: Character string for plot labeling
#
# Returns:
#   List of ggplot objects showing covariate effects
#-------------------------------------------------------------------------------

create_prediction_plots_original_scale <- function(best_model, best_formula, umf, original_values,
species_name) {
```

```
plots_list <- list()

# ABSOLUTE HUMIDITY EFFECT ON DETECTION PROBABILITY
if(grepl("ah", best_formula)) {
 # Convert scaled ranges back to original ecological scales
 ah_range_scaled <- seq(min(umf@obsCovs$ah, na.rm = TRUE),
                        max(umf@obsCovs$ah, na.rm = TRUE),
                        length.out = 100)
 ah_range_original <- ah_range_scaled * original_values$ah["sd"] + original_values$ah["mean"]

 new_data_ah <- data.frame(
  ah = ah_range_scaled,
  moon = ifelse(grepl("moon", best_formula), mean(umf@obsCovs$moon, na.rm = TRUE), 0)
 )

 pred_ah <- predict(best_model, type = 'det', newdata = new_data_ah, appendData = TRUE)
 pred_ah$ah_original <- ah_range_original

 p_ah <- ggplot(pred_ah, aes(ah_original, Predicted)) +
  geom_ribbon(aes(ymin = lower, ymax = upper), fill = "#1B9E77", alpha = 0.3) +
  geom_line(color = "#1B9E77", linewidth = 1.5) +
  scale_y_continuous(limits = c(0, 1)) +
  labs(
   x = "Absolute Humidity (g/m^3)",
   y = "Detection Probability",
   title = expression(paste(italic("Diasporus diastema"), ": Absolute Humidity Effect"))
  ) +
  theme_minimal() +
  theme(
   plot.title = element_text(size = 16, face = "bold", hjust = 0.5, margin = margin(b = 10)),
   axis.title.x = element_text(size = 14, face = "bold", margin = margin(t = 10)),
   axis.title.y = element_text(size = 14, face = "bold", margin = margin(r = 10)),
   axis.text.x = element_text(size = 12, color = "black"),
   axis.text.y = element_text(size = 12, color = "black"),
   panel.grid.minor = element_blank(),
   panel.grid.major = element_line(color = "grey90", linewidth = 0.5),
   plot.background = element_rect(fill = "white", color = NA)
  )

 plots_list[["absolute_humidity"]] <- p_ah
}

# LUNAR CYCLE EFFECT ON DETECTION PROBABILITY
if(grepl("moon", best_formula)) {
 # Convert scaled ranges back to original ecological scales
 moon_range_scaled <- seq(min(umf@obsCovs$moon, na.rm = TRUE),
                          max(umf@obsCovs$moon, na.rm = TRUE),
```

```
                    length.out = 100)
  moon_range_original <- moon_range_scaled * original_values$moon["sd"] +
original_values$moon["mean"]

  new_data_moon <- data.frame(
   ah = ifelse(grepl("ah", best_formula), mean(umf@obsCovs$ah, na.rm = TRUE), 0),
   moon = moon_range_scaled
  )

  pred_moon <- predict(best_model, type = 'det', newdata = new_data_moon, appendData = TRUE)
  pred_moon$moon_original <- moon_range_original

  # Create ecologically meaningful moon phase labels (using your original labels)
  moon_breaks <- seq(min(pred_moon$moon_original), max(pred_moon$moon_original),
length.out = 5)
  moon_labels <- c("New Moon", "First Quarter", "Waxing Gibbous", "Waning Gibbous", "Full Moon")

  p_moon <- ggplot(pred_moon, aes(moon_original, Predicted)) +
   geom_ribbon(aes(ymin = lower, ymax = upper), fill = "#7570B3", alpha = 0.3) +
   geom_line(color = "#7570B3", linewidth = 1.5) +
   scale_y_continuous(limits = c(0, 1)) +
   scale_x_continuous(
    name = "Moon Phase",
    breaks = moon_breaks,
    labels = moon_labels
   ) +
   labs(
    y = "Detection Probability",
    title = expression(paste(italic("Diasporus diastema"), ": Lunar Cycle Effect"))
   ) +
   theme_minimal() +
   theme(
    plot.title = element_text(size = 16, face = "bold", hjust = 0.5, margin = margin(b = 10)),
    axis.title.x = element_text(size = 14, face = "bold", margin = margin(t = 10)),
    axis.title.y = element_text(size = 14, face = "bold", margin = margin(r = 10)),
    axis.text.x = element_text(size = 11, color = "black", angle = 45, hjust = 1),
    axis.text.y = element_text(size = 12, color = "black"),
    panel.grid.minor = element_blank(),
    panel.grid.major = element_line(color = "grey90", linewidth = 0.5),
    plot.background = element_rect(fill = "white", color = NA)
   )

  plots_list[["moon"]] <- p_moon
 }

 return(plots_list)
}
```

```
# Generate ecological prediction plots for Diasporus diastema
diastema_plots <- create_prediction_plots_original_scale(best_diastema, best_formula_diastema,
d, diastema_original, "Diasporus diastema")

# Display all generated plots
cat("\n=== GENERATED PLOTS FOR Diasporus diastema ===\n")
for(plot_name in names(diastema_plots)) {
 cat("Displaying:", plot_name, "\n")
 print(diastema_plots[[plot_name]])
}

# Save individual species plots for publication
for(plot_name in names(diastema_plots)) {
 filename <- paste0("diastema_", plot_name, ".png")
 ggsave(filename, diastema_plots[[plot_name]], width = 10, height = 8, dpi = 300, bg = "white")
 cat("Saved:", filename, "\n")
}

#============================================================================
# RELATIVE IMPORTANCE ANALYSIS
#============================================================================

#----------------------------------------------------------------------------
# FUNCTION: CALCULATE RELATIVE IMPORTANCE OF ECOLOGICAL COVARIATES
#
# Computes relative importance values by summing Akaike weights across all
# models containing each covariate. Provides inference about which ecological
# factors most strongly influence occupancy and detection.
#
# Parameters:
#   selection_results: Output from model_selection_unmarked function
#
# Returns:
#   Dataframe with covariates ranked by relative importance
#----------------------------------------------------------------------------

calculate_relative_importance <- function(selection_results) {
 # Extract model results
 model_results <- selection_results$results

 # Define all ecological covariates considered
 all_covariates <- c("ah", "moon", "dap", "tree_rich", "tree_abu")

 # Calculate importance for each covariate
 importance_list <- list()

 for(covariate in all_covariates) {
  # Find models that contain this covariate
```

```
  if(covariate %in% c("ah", "moon")) {
   # Detection covariates
   models_with_covariate <- grepl(covariate, model_results$formula)
  } else {
   # Occupancy covariates
   models_with_covariate <- grepl(covariate, model_results$formula)
  }

  # Sum Akaike weights of models containing this covariate
  total_weight <- sum(model_results$weight[models_with_covariate])

  importance_list[[covariate]] <- data.frame(
   covariate = covariate,
   importance = total_weight
  )
 }

 # Combine all importance values and rank by importance
 importance_df <- bind_rows(importance_list) %>%
  arrange(desc(importance))

 return(importance_df)
}

# Calculate relative importance for Diasporus diastema
importance_diastema <- calculate_relative_importance(selection_diastema)

cat("\n=== RELATIVE IMPORTANCE FOR Diasporus diastema ===\n")
print(importance_diastema)

#------------------------------------------------------------------------------
# FUNCTION: INDIVIDUAL SPECIES RELATIVE IMPORTANCE PLOTS
#
# Creates separate relative importance plots with consistent color scheme
# for individual species assessment.
#
# Parameters:
#   importance_df: Importance data for a single species
#   species_name: Character string for plot title
#------------------------------------------------------------------------------

create_single_relimp_plot <- function(importance_df, species_name) {
 # Use consistent ecological color scheme
 colors <- c("ah" = "#35978f", "moon" = "#b0b0b0",
         "dap" = "#bf812d", "tree_rich" = "#66A61E", "tree_abu" = "#01665e")

 # Create meaningful ecological labels
 pretty_labels <- c(
```

```
  "ah" = "Absolute Humidity",
  "moon" = "Lunar cycle",
  "dap" = "Tree diameter (DBH)",
  "tree_rich" = "Tree richness",
  "tree_abu" = "Tree abundance"
 )

 importance_df <- importance_df %>%
  mutate(label = pretty_labels[covariate])

 ggplot(importance_df, aes(x = importance, y = reorder(label, importance))) +
  geom_col(aes(fill = covariate), width = 0.7, alpha = 0.8) +
  geom_text(aes(label = sprintf("%.2f", importance)),
        hjust = -0.2, size = 5, color = "black") +
  scale_fill_manual(values = colors) +
  scale_x_continuous(
   limits = c(0, 1.1),
   expand = expansion(mult = c(0, 0.05)),
   breaks = seq(0, 1, 0.2)
  ) +
  labs(
   x = "Relative Importance",
   y = NULL,
   title = species_name,
   subtitle = "Relative importance of covariates"
  ) +
  theme_minimal() +
  theme(
   legend.position = "none",
   plot.title = element_text(face = "italic", hjust = 0.5, size = 18, margin = margin(b = 10)),
   plot.subtitle = element_text(hjust = 0.5, size = 14, color = "grey40", margin = margin(b = 15)),
   axis.title.x = element_text(face = "bold", size = 16, margin = margin(t = 10)),
   axis.text.y = element_text(face = "bold", size = 14, color = "black"),
   axis.text.x = element_text(size = 12),
   # Add black axis lines
   axis.line = element_line(color = "black", linewidth = 0.5),
   axis.ticks = element_line(color = "black"),
   panel.grid.major.y = element_blank(),
   panel.grid.minor.y = element_blank(),
   panel.grid.major.x = element_line(color = "grey90"),
   panel.grid.minor.x = element_blank(),
   panel.border = element_blank(),
   plot.background = element_rect(fill = "white", color = NA)
  ) +
  coord_cartesian(clip = "off")
}

# Create individual species relative importance plot
```

```
relimp_diastema <- create_single_relimp_plot(importance_diastema, "Diasporus diastema")

# Display and save relative importance plot
print(relimp_diastema)
ggsave("relative_importance_diastema.png", relimp_diastema,
     width = 10, height = 8, dpi = 300, bg = "white")

#==============================================================================
# RESULTS DISPLAY AND SUMMARY
#==============================================================================

# Display all generated plots in R graphics device
cat("\n=== FINAL DISPLAY OF ALL ECOLOGICAL PLOTS ===\n")

# Display individual species prediction plots
cat("\n--- Diasporus diastema ---\n")
for(plot_name in names(diastema_plots)) {
  cat("Displaying:", plot_name, "\n")
  print(diastema_plots[[plot_name]])
}

# Display relative importance plot
cat("\n--- Relative Importance Analysis ---\n")
print(relimp_diastema)

#------------------------------------------------------------------------------
# ECOLOGICAL ANALYSIS SUMMARY
#
# Provides concise summary of key findings for Diasporus diastema, including
# best models, AIC values, and ecological interpretation of results.
#------------------------------------------------------------------------------

cat("\n=== ECOLOGICAL ANALYSIS SUMMARY ===\n")
cat("Diasporus diastema:\n")
cat("  Best model:", best_formula_diastema, "\n")
cat("  AIC:", round(best_diastema@AIC, 2), "\n")
cat("  Ecological drivers identified:", names(diastema_plots), "\n\n")

cat("Relative importance of covariates:\n")
for(i in 1:nrow(importance_diastema)) {
  cat("  ", importance_diastema$covariate[i], ":", round(importance_diastema$importance[i], 3),
"\n")
}

cat("\nFiles generated:\n")
cat("- correlation_diastema.png\n")
for(plot_name in names(diastema_plots)) {
  cat("- diastema_", plot_name, ".png\n", sep = "")
```

```
}
cat("- relative_importance_diastema.png\n")

#------------------------------------------------------------------------------
# FUNCTION: CREATE COMBINED FIGURE FOR DIASPORUS DIASTEMA
#
# Generates a side-by-side figure showing both absolute humidity and moon effects
# for clean figure organization in publications.
#
# Parameters:
#   plots_list: List of ggplot objects from create_prediction_plots_original_scale
#   species_name: Character string for plot title
#------------------------------------------------------------------------------

create_combined_diastema_figure <- function(plots_list, species_name) {
 # Check if we have both absolute humidity and moon plots
 has_ah <- "absolute_humidity" %in% names(plots_list)
 has_moon <- "moon" %in% names(plots_list)

 if(has_ah && has_moon) {
  # Get the individual plots
  p_ah <- plots_list[["absolute_humidity"]]
  p_moon <- plots_list[["moon"]]

  # Remove individual titles since we'll have a combined title
  p_ah <- p_ah + labs(title = NULL)
  p_moon <- p_moon + labs(title = NULL)

  # Combine using patchwork
  combined <- p_ah + p_moon +
   plot_layout(ncol = 2) +
   plot_annotation(
    title = paste("Environmental Effects on Detection Probability -", species_name),
    theme = theme(
     plot.title = element_text(hjust = 0.5, face = "bold", size = 18)
    )
   )

  return(combined)
 } else {
  cat("Cannot create combined figure: missing absolute humidity or moon plot\n")
  return(NULL)
 }
}

# Create and save combined figure for Diasporus diastema
combined_diastema <- create_combined_diastema_figure(diastema_plots, "Diasporus diastema")
```

```
if(!is.null(combined_diastema)) {
  print(combined_diastema)
  ggsave("diastema_combined_effects.png", combined_diastema,
       width = 16, height = 8, dpi = 300, bg = "white")
  cat("Saved: diastema_combined_effects.png\n")
}

cat("\n✅ OCCUPANCY MODELING ANALYSIS FOR DIASPORUS DIASTEMA COMPLETE\n")
cat("All ecological plots and analyses saved for publication.\n")
```

#RELATIVE IMPORTANCE OF COVARIATES IN MODELS (FINAL VISUALIZATION)

```
#============================================================================


# LIBRARIES
library(ggplot2)
library(dplyr)
library(tidyr)
library(tibble)
library(scales)
library(fmsb)
library(patchwork)  # For combining plots

# ============================================================================
# DATA PREPARATION - Combine all three species
# ============================================================================

# Assuming you have these data frames from your modeling:
# importance_ca, importance_cru, importance_diastema

# Combine all species into one data frame
all_importance <- bind_rows(
  importance_ca %>% mutate(species = "Pristimantis caryophyllaceus"),
  importance_cru %>% mutate(species = "Pristimantis cruentus"),
  importance_diastema %>% mutate(species = "Diasporus diastema")
)

# Define consistent ecological color scheme
covariate_colors <- c(
  "ah" = "#35978f",       # Teal for absolute humidity
  "moon" = "#b0b0b0",     # Gray for moon
  "dap" = "#bf812d",      # Brown for tree diameter
  "tree_rich" = "#66A61E", # Green for tree richness
  "tree_abu" = "#01665e"   # Dark green for tree abundance
)

# Create meaningful ecological labels
pretty_labels <- c(
  "ah" = "Absolute Humidity",
  "moon" = "Lunar cycle",
  "dap" = "Tree diameter (DBH)",
  "tree_rich" = "Tree richness",
  "tree_abu" = "Tree abundance"
)

# Apply labels to the combined data
all_importance <- all_importance %>%
  mutate(
```

```
  label = pretty_labels[covariate],
  species = factor(species, levels = c("Pristimantis caryophyllaceus",
                                       "Pristimantis cruentus",
                                       "Diasporus diastema"))
 )

# ============================================================================
# SMALL MULTIPLES PLOT - Three species together
# ============================================================================

p_small_multiples <- ggplot(all_importance, aes(x = importance, y = reorder(label, importance))) +
 geom_col(aes(fill = covariate), width = 0.7, alpha = 0.8) +
 geom_text(aes(label = sprintf("%.2f", importance)),
           hjust = -0.2, size = 3.5, color = "black") +
 facet_wrap(~ species, ncol = 3, scales = "free_y") +
 scale_fill_manual(values = covariate_colors) +
 scale_x_continuous(
  limits = c(0, 1.1),
  expand = expansion(mult = c(0, 0.05)),
  breaks = seq(0, 1, 0.2)
 ) +
 labs(
  x = "Relative Importance",
  y = NULL,
  title = "Relative Importance of Ecological Covariates",
  subtitle = "Comparison across three amphibian species"
 ) +
 theme_minimal() +
 theme(
  legend.position = "none",
  plot.title = element_text(face = "bold", hjust = 0.5, size = 16, margin = margin(b = 10)),
  plot.subtitle = element_text(hjust = 0.5, size = 12, color = "grey40", margin = margin(b = 15)),
  axis.title.x = element_text(face = "bold", size = 12, margin = margin(t = 10)),
  axis.text.y = element_text(face = "bold", size = 10, color = "black"),
  axis.text.x = element_text(size = 9),
  axis.line = element_line(color = "black", linewidth = 0.5),
  axis.ticks = element_line(color = "black"),
  panel.grid.major.y = element_blank(),
  panel.grid.minor.y = element_blank(),
  panel.grid.major.x = element_line(color = "grey90"),
  panel.grid.minor.x = element_blank(),
  panel.border = element_blank(),
  strip.text = element_text(face = "italic", size = 10, color = "black"),
  strip.background = element_rect(fill = "grey90", color = NA),
  plot.background = element_rect(fill = "white", color = NA)
 ) +
 coord_cartesian(clip = "off")
```

```
# Save small multiples at 300 DPI
ggsave("SmallMultiples_RelativeImportance_300dpi.png",
     plot = p_small_multiples,
     width = 14,
     height = 6,
     dpi = 300,
     bg = "white")

# ============================================================================
# RADAR CHART - Three species comparison
# ============================================================================

# Prepare data for radar chart
radar_data <- all_importance %>%
  select(covariate, species, importance) %>%
  pivot_wider(names_from = covariate, values_from = importance) %>%
  as.data.frame() %>%
  column_to_rownames("species")

# Normalize data for radar chart (0-1 scale) - optional but good for comparison
normalize <- function(x) {
  (x - min(x, na.rm = TRUE)) / (max(x, na.rm = TRUE) - min(x, na.rm = TRUE))
}

radar_normalized <- as.data.frame(lapply(radar_data, normalize))
radar_normalized$species <- rownames(radar_data)

# Prepare data for fmsb (first row max, second row min, then actual data)
radar_plot_data <- rbind(rep(1, ncol(radar_normalized)-1),
                         rep(0, ncol(radar_normalized)-1),
                         radar_normalized[,-ncol(radar_normalized)])

# Define species colors for radar chart
species_colors <- c(
  "Pristimantis caryophyllaceus" = "#E41A1C",  # Red
  "Pristimantis cruentus" = "#377EB8",       # Blue
  "Diasporus diastema" = "#4DAF4A"           # Green
)

species_colors_alpha <- c(
  alpha("#E41A1C", 0.3),
  alpha("#377EB8", 0.3),
  alpha("#4DAF4A", 0.3)
)

# Create and save radar chart at 300 DPI
png("RadarChart_RelativeImportance_300dpi.png", width = 10, height = 8, units = "in", res = 300)
```

```
# Set margins and create radar chart
par(mar = c(2, 2, 3, 2))
radarchart(radar_plot_data,
        axistype = 1,
        pcol = species_colors,
        pfcol = species_colors_alpha,
        plwd = 3,
        plty = 1,
        cglcol = "darkgray",
        cglty = 1,
        axislabcol = "black",
        caxislabels = seq(0, 1, 0.2),
        cglwd = 1.2,
        vlcex = 1.1,
        calcex = 1.0,
        title = "Relative Importance of Covariates\nThree Amphibian Species Comparison")

# Add legend
legend("topright",
    legend = rownames(radar_data),
    bty = "o",
    bg = "white",
    pch = 20,
    col = species_colors,
    text.col = "black",
    cex = 1.1,
    pt.cex = 2.5)

dev.off()

# ============================================================================
# DISPLAY PLOTS
# ============================================================================

cat("✓ Small multiples plot saved: SmallMultiples_RelativeImportance_300dpi.png\n")
cat("✓ Radar chart saved: RadarChart_RelativeImportance_300dpi.png\n")

# Display plots in R
print(p_small_multiples)

# For radar chart display (will create in plot window)
par(mar = c(2, 2, 3, 2))
radarchart(radar_plot_data,
        axistype = 1,
        pcol = species_colors,
        pfcol = species_colors_alpha,
        plwd = 3,
        plty = 1,
```

```
        cglcol = "darkgray",
        cglty = 1,
        axislabcol = "black",
        caxislabels = seq(0, 1, 0.2),
        cglwd = 1.2,
        vlcex = 1.1,
        calcex = 1.0,
        title = "Relative Importance of Covariates\nThree Amphibian Species Comparison")

legend("topright",
     legend = rownames(radar_data),
     bty = "o",
     bg = "white",
     pch = 20,
     col = species_colors,
     text.col = "black",
     cex = 1.1,
     pt.cex = 2.5)

#FINALVERSIONFORPAPERSINGLEMULTIPLES
# LIBRARIES
library(ggplot2)
library(dplyr)
library(tidyr)

# ============================================================================
# DATA PREPARATION - Combine all three species
# ============================================================================

# Combine all species into one data frame
all_importance <- bind_rows(
  importance_ca %>% mutate(species = "Pristimantis caryophyllaceus"),
  importance_cru %>% mutate(species = "Pristimantis cruentus"),
  importance_diastema %>% mutate(species = "Diasporus diastema")
)

# Create meaningful ecological labels (changed "Lunar cycle" to "moon")
pretty_labels <- c(
  "ah" = "Absolute Humidity",
  "moon" = "Moon illumination",
  "dap" = "Tree diameter (dbh)",
  "tree_rich" = "Tree richness",
  "tree_abu" = "Tree abundance"
)

# Apply labels to the combined data and set species order
all_importance <- all_importance %>%
  mutate(
```

```
   label = pretty_labels[covariate],
   species = factor(species, levels = c("Pristimantis caryophyllaceus",
                                        "Pristimantis cruentus",
                                        "Diasporus diastema"))
  ) %>%
  # Reorder the labels by their mean importance across all species
  mutate(label = factor(label, levels = unique(label[order(importance)])))

# Define species colors as requested
species_colors <- c(
  "Pristimantis caryophyllaceus" = "#dfc27d",  # Tan
  "Pristimantis cruentus" = "#bf812d",       # Brownish
  "Diasporus diastema" = "#377EB8"           # Blue
)

# ==============================================================================
# SMALL MULTIPLES PLOT - Revised version
# ==============================================================================

p_small_multiples <- ggplot(all_importance, aes(x = importance, y = label)) +
  geom_col(aes(fill = species), width = 0.7, alpha = 0.8, position = position_dodge(0.8)) +
  geom_text(aes(label = sprintf("%.2f", importance)),
            position = position_dodge(0.8),
            hjust = -0.2, size = 3.5, color = "black") +
  facet_wrap(~ species, ncol = 3) +
  scale_fill_manual(values = species_colors) +
  scale_x_continuous(
    limits = c(0, 1.1),
    expand = expansion(mult = c(0, 0.05)),
    breaks = seq(0, 1, 0.2)
  ) +
  labs(
    x = "Relative Importance",
    y = NULL,
    title = "",
    subtitle = ""
  ) +
  theme_minimal() +
  theme(
    legend.position = "none",
    plot.title = element_text(face = "bold", hjust = 0.5, size = 16, margin = margin(b = 10)),
    plot.subtitle = element_text(hjust = 0.5, size = 12, color = "grey40", margin = margin(b = 15)),
    axis.title.x = element_text(face = "bold", size = 12, margin = margin(t = 10)),
    axis.text.y = element_text(face = "bold", size = 10, color = "black"),
    axis.text.x = element_text(size = 9),
    axis.line = element_line(color = "black", linewidth = 0.5),
    axis.ticks = element_line(color = "black"),
    panel.grid.major.y = element_blank(),
```

```
    panel.grid.minor.y = element_blank(),
    panel.grid.major.x = element_line(color = "grey90"),
    panel.grid.minor.x = element_blank(),
    panel.border = element_blank(),
    strip.text = element_text(face = "italic", size = 10, color = "black"),
    strip.background = element_rect(fill = "grey90", color = NA),
    plot.background = element_rect(fill = "white", color = NA)
  ) +
  coord_cartesian(clip = "off")

# Save small multiples at 300 DPI
ggsave("SmallMultiples_RelativeImportance_300dpi.png",
       plot = p_small_multiples,
       width = 14,
       height = 6,
       dpi = 300,
       bg = "white")

# Display the plot
print(p_small_multiples)

cat("✓ Small multiples plot saved: SmallMultiples_RelativeImportance_300dpi.png\n")

#DOTPLOT
# LIBRARIES
library(ggplot2)
library(dplyr)
library(tidyr)

# ============================================================================
# DATA PREPARATION - Combine all three species
# ============================================================================

# Combine all species into one data frame
all_importance <- bind_rows(
  importance_ca %>% mutate(species = "Pristimantis caryophyllaceus"),
  importance_cru %>% mutate(species = "Pristimantis cruentus"),
  importance_diastema %>% mutate(species = "Diasporus diastema")
)

# Create meaningful ecological labels (changed "Lunar cycle" to "moon")
pretty_labels <- c(
  "ah" = "Absolute Humidity",
  "moon" = "Moon illumination",
  "dap" = "Tree diameter (DBH)",
  "tree_rich" = "Tree richness",
  "tree_abu" = "Tree abundance"
)
```

```
# Apply labels to the combined data and set species order
all_importance <- all_importance %>%
 mutate(
  label = pretty_labels[covariate],
  species = factor(species, levels = c("Pristimantis caryophyllaceus",
                                       "Pristimantis cruentus",
                                       "Diasporus diastema"))
 ) %>%
 # Reorder the labels by their mean importance across all species
 mutate(label = factor(label, levels = unique(label[order(importance)])))

# Define species colors as requested
species_colors <- c(
 "Pristimantis caryophyllaceus" = "#dfc27d",  # Tan
 "Pristimantis cruentus" = "#bf812d",       # Brownish
 "Diasporus diastema" = "#377EB8"          # Blue
)

# ==============================================================================
# DOT PLOT - Publication Quality
# ==============================================================================

p_dot <- ggplot(all_importance, aes(x = importance, y = label, color = species)) +
 geom_point(size = 3.5, position = position_dodge(width = 0.5)) +
 geom_linerange(aes(xmin = 0, xmax = importance),
                position = position_dodge(width = 0.5), linewidth = 1.2) +
 geom_text(aes(label = sprintf("%.2f", importance)),
           position = position_dodge(width = 0.5),
           hjust = -0.3, size = 3.2, color = "black", fontface = "bold") +
 scale_color_manual(values = species_colors, name = "Species") +
 scale_x_continuous(
  limits = c(0, 1.2),
  breaks = seq(0, 1, 0.2),
  expand = expansion(mult = c(0, 0.1))
 ) +
 labs(
  x = "Relative Importance",
  y = NULL,
  title = "",
  subtitle = ""
 ) +
 theme_minimal() +
 theme(
  legend.position = "bottom",
  plot.title = element_text(face = "bold", hjust = 0.5, size = 16, margin = margin(b = 8)),
  plot.subtitle = element_text(hjust = 0.5, size = 12, color = "grey40", margin = margin(b = 12)),
  axis.title.x = element_text(face = "bold", size = 12, margin = margin(t = 10)),
```

```
    axis.text.y = element_text(face = "bold", size = 11, color = "black"),
    axis.text.x = element_text(size = 10),
    axis.line = element_line(color = "black", linewidth = 0.5),
    axis.ticks = element_line(color = "black"),
    panel.grid.major.y = element_line(color = "grey90"),
    panel.grid.minor.y = element_blank(),
    panel.grid.major.x = element_line(color = "grey90"),
    panel.grid.minor.x = element_blank(),
    legend.text = element_text(face = "italic", size = 10),
    legend.title = element_text(face = "bold", size = 11),
    plot.background = element_rect(fill = "white", color = NA)
  )

# Save dot plot at 300 DPI
ggsave("DotPlot_RelativeImportance_300dpi.png",
       plot = p_dot,
       width = 10,
       height = 6,
       dpi = 300,
       bg = "white")

# Display the plot
print(p_dot)

cat("✓ Dot plot saved: DotPlot_RelativeImportance_300dpi.png\n")
cat("✓ File dimensions: 10 x 6 inches at 300 DPI\n")
cat("✓ Species colors: Tan (P. caryophyllaceus), Brown (P. cruentus), Blue (D. diastema)\n")

# LIBRARIES
library(ggplot2)
library(dplyr)
library(tidyr)

# ============================================================================
# DATA PREPARATION - Combine all three species
# ============================================================================

# Combine all species into one data frame
all_importance <- bind_rows(
  importance_ca %>% mutate(species = "Pristimantis caryophyllaceus"),
  importance_cru %>% mutate(species = "Pristimantis cruentus"),
  importance_diastema %>% mutate(species = "Diasporus diastema")
)

# Create meaningful ecological labels (changed "Lunar cycle" to "moon")
pretty_labels <- c(
  "ah" = "Absolute Humidity",
  "moon" = "Moon",
```

```
  "dap" = "Tree diameter (DBH)",
  "tree_rich" = "Tree richness",
  "tree_abu" = "Tree abundance"
)

# Apply labels to the combined data and set species order
all_importance <- all_importance %>%
  mutate(
    label = pretty_labels[covariate],
    species = factor(species, levels = c("Pristimantis caryophyllaceus",
                                         "Pristimantis cruentus",
                                         "Diasporus diastema"))
  )

# ============================================================================
# SLOPE GRAPH - Publication Quality
# ============================================================================

p_slope <- ggplot(all_importance, aes(x = species, y = importance, group = label)) +
  geom_line(aes(color = label), alpha = 0.7, linewidth = 1.2) +
  geom_point(aes(color = label), size = 3) +
  geom_text(data = filter(all_importance, species == "Pristimantis caryophyllaceus"),
            aes(label = label, x = 0.9), hjust = 1, size = 3.5, check_overlap = TRUE) +
  scale_color_brewer(palette = "Dark2", name = "Covariates") +
  scale_y_continuous(
    limits = c(0, 1.1),
    breaks = seq(0, 1, 0.2)
  ) +
  labs(
    x = NULL,
    y = "Relative Importance",
    title = "Relative Importance Patterns Across Species",
    subtitle = "Slope graph showing covariate importance trends"
  ) +
  theme_minimal() +
  theme(
    axis.text.x = element_text(face = "italic", size = 11, color = "black"),
    axis.text.y = element_text(size = 10, color = "black"),
    axis.title.y = element_text(face = "bold", size = 12, margin = margin(r = 10)),
    axis.line = element_line(color = "black", linewidth = 0.5),
    axis.ticks = element_line(color = "black"),
    panel.grid.major.y = element_line(color = "grey90"),
    panel.grid.minor.y = element_blank(),
    panel.grid.major.x = element_blank(),
    panel.grid.minor.x = element_blank(),
    plot.title = element_text(face = "bold", hjust = 0.5, size = 16, margin = margin(b = 8)),
    plot.subtitle = element_text(hjust = 0.5, size = 12, color = "grey40", margin = margin(b = 15)),
    legend.position = "none",
```

```
   plot.background = element_rect(fill = "white", color = NA)
  )

# Save slope graph at 300 DPI
ggsave("SlopePlot_RelativeImportance_300dpi.png",
       plot = p_slope,
       width = 10,
       height = 6,
       dpi = 300,
       bg = "white")

# Display the plot
print(p_slope)

cat("✓ Slope graph saved: SlopePlot_RelativeImportance_300dpi.png\n")
cat("✓ File dimensions: 10 x 6 inches at 300 DPI\n")
cat("✓ Features: Shows trends in covariate importance across species\n")

# LIBRARIES
library(ggplot2)
library(dplyr)
library(tidyr)

# ============================================================================
# DATA PREPARATION - Combine all three species
# ============================================================================

# Combine all species into one data frame
all_importance <- bind_rows(
  importance_ca %>% mutate(species = "Pristimantis caryophyllaceus"),
  importance_cru %>% mutate(species = "Pristimantis cruentus"),
  importance_diastema %>% mutate(species = "Diasporus diastema")
)

# Create meaningful ecological labels (changed "Lunar cycle" to "moon")
pretty_labels <- c(
  "ah" = "Absolute Humidity",
  "moon" = "Moon",
  "dap" = "Tree diameter (DBH)",
  "tree_rich" = "Tree richness",
  "tree_abu" = "Tree abundance"
)

# Apply labels to the combined data and set species order
all_importance <- all_importance %>%
  mutate(
    label = pretty_labels[covariate],
    species = factor(species, levels = c("Pristimantis caryophyllaceus",
```

```
                        "Pristimantis cruentus",
                        "Diasporus diastema"))
 ) %>%
 # Order covariates by their mean importance for better visualization
 mutate(label = factor(label, levels = unique(label[order(-importance)])))

# Define species colors as requested
species_colors <- c(
 "Pristimantis caryophyllaceus" = "#dfc27d",  # Tan
 "Pristimantis cruentus" = "#bf812d",       # Brownish
 "Diasporus diastema" = "#377EB8"           # Blue
)

# ============================================================================
# GROUPED BARS - Publication Quality
# ============================================================================

p_grouped <- ggplot(all_importance, aes(x = label, y = importance, fill = species)) +
 geom_col(position = position_dodge(0.8), width = 0.7, alpha = 0.9) +
 geom_text(aes(label = sprintf("%.2f", importance)),
         position = position_dodge(0.8),
         vjust = -0.5, size = 3.2, fontface = "bold") +
 scale_fill_manual(values = species_colors, name = "Species") +
 scale_y_continuous(
  limits = c(0, 1.1),
  breaks = seq(0, 1, 0.2),
  expand = expansion(mult = c(0, 0.05))
 ) +
 labs(
  x = NULL,
  y = "Relative Importance",
  title = "",
  subtitle = ""
 ) +
 theme_minimal() +
 theme(
  axis.text.x = element_text(face = "bold", size = 11, angle = 45, hjust = 1, color = "black"),
  axis.text.y = element_text(size = 10, color = "black"),
  axis.title.y = element_text(face = "bold", size = 12, margin = margin(r = 10)),
  axis.line = element_line(color = "black", linewidth = 0.5),
  axis.ticks = element_line(color = "black"),
  panel.grid.major.y = element_line(color = "grey90"),
  panel.grid.minor.y = element_blank(),
  panel.grid.major.x = element_blank(),
  panel.grid.minor.x = element_blank(),
  plot.title = element_text(face = "bold", hjust = 0.5, size = 16, margin = margin(b = 8)),
  plot.subtitle = element_text(hjust = 0.5, size = 12, color = "grey40", margin = margin(b = 15)),
  legend.position = "bottom",
```

```
  legend.text = element_text(face = "italic", size = 10),
  legend.title = element_text(face = "bold", size = 11),
  plot.background = element_rect(fill = "white", color = NA)
 )

# Save grouped bars at 300 DPI
ggsave("GroupedBars_RelativeImportance_300dpi.png",
     plot = p_grouped,
     width = 11,
     height = 7,
     dpi = 300,
     bg = "white")

# Display the plot
print(p_grouped)

cat("✓ Grouped bars saved: GroupedBars_RelativeImportance_300dpi.png\n")
cat("✓ File dimensions: 11 x 7 inches at 300 DPI\n")
cat("✓ Species colors: Tan (P. caryophyllaceus), Brown (P. cruentus), Blue (D. diastema)\n")
cat("✓ Features: Classic grouped bar chart for clear comparison\n")
```

#FINAL PLOTS FOR VISUALIZATION (OCCUPANCY)

```
#==========================================================================


Packages
library(ggplot2)
library(dplyr)
library(tidyr)
library(cowplot)

# --- DATA (means and SE as provided) ---
data <- tibble(
  Covariate = c("Tree abundance", "Tree diameter (cm)", "Tree richness",
                "Air temperature (°C)", "Relative humidity (%)"),
  Old_Mean = c(12.00, 51.34, 7.29, 18.34, 83.34),
  Old_SE   = c(5.29, 20.61, 2.93, 1.63, 27.36),
  Sec_Mean = c(38.14, 11.48, 4.86, 17.90, 72.63),
  Sec_SE   = c(13.25, 9.22, 2.04, 2.13, 26.68)
)

# --- FUNCTIONS FOR ABSOLUTE HUMIDITY AND DERIVATIVES ---
# AH (g/m^3) based on Tetens formula:
# AH = C * es(T) * RH_frac / (273.15 + T)
# where es(T) = 6.112 * exp(17.67*T/(T+243.5)) (hPa),
# C = 2.1674 (conversion factor to g/m3 when es in hPa and T in °C)
AH_from_T_RH <- function(T_degC, RH_percent) {
  RH_frac <- RH_percent / 100
  es <- 6.112 * exp((17.67 * T_degC) / (T_degC + 243.5))
  C <- 2.1674
  AH <- C * es * RH_frac / (273.15 + T_degC)
  return(AH)
}

# Partial derivatives for delta-method variance propagation
AH_partials <- function(T_degC, RH_percent) {
  # Constants
  b <- 243.5
  C <- 2.1674
  RH_frac <- RH_percent / 100

  # es and derivative df/dT
  es <- 6.112 * exp((17.67 * T_degC) / (T_degC + b))
  a_prime <- 17.67 * b / (T_degC + b)^2        # derivative of exponent a(T)
  des_dT <- es * a_prime                       # derivative of es w.r.t. T

  # AH = C * es * RH_frac / (273.15 + T)
  denom <- 273.15 + T_degC
```

```
  # derivative wrt T (°C)
  dAH_dT <- C * RH_frac * (des_dT / denom - es / denom^2)

  # derivative wrt RH percent (because input RH is in %)
  # d(AH)/d(RH_percent) = C * es / denom * d(RH_frac)/d(RH_percent)
  # d(RH_frac)/d(RH_percent) = 1/100
  dAH_dRHpct <- C * es / denom * (1/100)

  return(list(dAH_dT = dAH_dT, dAH_dRHpct = dAH_dRHpct))
}

# --- CALCULATE MEAN AND SE OF ABSOLUTE HUMIDITY ---
# Take means and SEs of T and RH (SE of RH in percentage points)
T_old <- data$Old_Mean[data$Covariate == "Air temperature (°C)"]
SE_T_old <- data$Old_SE[data$Covariate == "Air temperature (°C)"]
RH_old <- data$Old_Mean[data$Covariate == "Relative humidity (%)"]
SE_RH_old <- data$Old_SE[data$Covariate == "Relative humidity (%)"]

T_sec <- data$Sec_Mean[data$Covariate == "Air temperature (°C)"]
SE_T_sec <- data$Sec_SE[data$Covariate == "Air temperature (°C)"]
RH_sec <- data$Sec_Mean[data$Covariate == "Relative humidity (%)"]
SE_RH_sec <- data$Sec_SE[data$Covariate == "Relative humidity (%)"]

# Compute AH means
AH_old_mean <- AH_from_T_RH(T_old, RH_old)
AH_sec_mean <- AH_from_T_RH(T_sec, RH_sec)

# Compute partials and use delta method for variance (Var ≈ (d/dT)^2 Var(T) + (d/dRH%)^2
Var(RH%))
partials_old <- AH_partials(T_old, RH_old)
partials_sec  <- AH_partials(T_sec, RH_sec)

var_AH_old <- (partials_old$dAH_dT^2) * (SE_T_old^2) + (partials_old$dAH_dRHpct^2) *
(SE_RH_old^2)
var_AH_sec  <- (partials_sec$dAH_dT^2) * (SE_T_sec^2) + (partials_sec$dAH_dRHpct^2) *
(SE_RH_sec^2)

AH_old_se <- sqrt(var_AH_old)
AH_sec_se  <- sqrt(var_AH_sec)

# Print results for check
cat("Estimated Absolute Humidity (Old Forest):", round(AH_old_mean, 3), "g/m^3  ±",
round(AH_old_se, 3), " (SE)\n")
cat("Estimated Absolute Humidity (Secondary):", round(AH_sec_mean, 3), "g/m^3  ±",
round(AH_sec_se, 3), " (SE)\n")

# --- ADD AH TO ORIGINAL DATAFRAME (for plotting) ---
data2 <- data %>%
```

```
  add_row(
    Covariate = "Absolute humidity",
    Old_Mean = AH_old_mean,
    Old_SE = AH_old_se,
    Sec_Mean = AH_sec_mean,
    Sec_SE = AH_sec_se
  )

# Reshape to long format for ggplot
data_long <- data2 %>%
  pivot_longer(cols = c(Old_Mean, Sec_Mean, Old_SE, Sec_SE),
               names_to = c("Forest", ".value"),
               names_pattern = "(Old|Sec)_(.*)") %>%
  mutate(Forest = ifelse(Forest == "Old", "Old Forest", "Secondary Forest"))

# --- GRAPH 1: Tree Abundance and Richness (dark colors for contrast) ---
tree_covs <- c("Tree abundance", "Tree richness")
tree_df <- filter(data_long, Covariate %in% tree_covs)

p_tree <- ggplot(tree_df, aes(x = Covariate, y = Mean, fill = Forest)) +
  geom_col(position = position_dodge(width = 0.8), width = 0.7) +
  geom_errorbar(aes(ymin = Mean - SE, ymax = Mean + SE),
                position = position_dodge(width = 0.8), width = 0.15, size = 0.7) +
  scale_fill_manual(values = c("Old Forest" = "#1b5e20", "Secondary Forest" = "#81c784")) +
  labs(title = "", y = "Quantity", x = "") +
  theme_minimal(base_size = 13) +
  theme(axis.text.x = element_text(angle = 0, hjust = 0.5),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.line = element_line(color = "darkgrey", size = 0.5),
        axis.ticks = element_line(color = "darkgrey"),
        legend.position = "bottom",
        legend.title = element_blank())

# --- GRAPH 2: Tree Diameter (dark colors for contrast) ---
dbh_covs <- c("Tree diameter (cm)")
dbh_df <- filter(data_long, Covariate %in% dbh_covs)

p_dbh <- ggplot(dbh_df, aes(x = Covariate, y = Mean, fill = Forest)) +
  geom_col(position = position_dodge(width = 0.8), width = 0.7) +
  geom_errorbar(aes(ymin = Mean - SE, ymax = Mean + SE),
                position = position_dodge(width = 0.8), width = 0.15, size = 0.7) +
  scale_fill_manual(values = c("Old Forest" = "#004d40", "Secondary Forest" = "#4db6ac")) +
  labs(title = "", y = "Diameter (cm)", x = "") +
  theme_minimal(base_size = 13) +
  theme(axis.text.x = element_text(angle = 0, hjust = 0.5),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
```

```
      axis.line = element_line(color = "darkgrey", size = 0.5),
      axis.ticks = element_line(color = "darkgrey"),
      legend.position = "bottom",
      legend.title = element_blank())

# --- GRAPH 3: Temperature (dark colors for contrast) ---
temp_covs <- c("Air temperature (°C)")
temp_df <- filter(data_long, Covariate %in% temp_covs)

p_temp <- ggplot(temp_df, aes(x = Covariate, y = Mean, fill = Forest)) +
  geom_col(position = position_dodge(width = 0.8), width = 0.7) +
  geom_errorbar(aes(ymin = Mean - SE, ymax = Mean + SE),
                position = position_dodge(width = 0.8), width = 0.15, size = 0.7) +
  scale_fill_manual(values = c("Old Forest" = "#01579b", "Secondary Forest" = "#4fc3f7")) +
  labs(title = "", y = "Temperature (°C)", x = "") +
  theme_minimal(base_size = 13) +
  theme(axis.text.x = element_text(angle = 0, hjust = 0.5),
      panel.grid.major = element_blank(),
      panel.grid.minor = element_blank(),
      axis.line = element_line(color = "darkgrey", size = 0.5),
      axis.ticks = element_line(color = "darkgrey"),
      legend.position = "bottom",
      legend.title = element_blank())

# --- GRAPH 4: Relative Humidity (dark colors for contrast) ---
rh_covs <- c("Relative humidity (%)")
rh_df <- filter(data_long, Covariate %in% rh_covs)

p_rh <- ggplot(rh_df, aes(x = Covariate, y = Mean, fill = Forest)) +
  geom_col(position = position_dodge(width = 0.8), width = 0.7) +
  geom_errorbar(aes(ymin = Mean - SE, ymax = Mean + SE),
                position = position_dodge(width = 0.8), width = 0.15, size = 0.7) +
  scale_fill_manual(values = c("Old Forest" = "#006064", "Secondary Forest" = "#26c6da")) +
  labs(title = "", y = "Relative humidity (%)", x = "") +
  theme_minimal(base_size = 13) +
  theme(axis.text.x = element_text(angle = 0, hjust = 0.5),
      panel.grid.major = element_blank(),
      panel.grid.minor = element_blank(),
      axis.line = element_line(color = "darkgrey", size = 0.5),
      axis.ticks = element_line(color = "darkgrey"),
      legend.position = "bottom",
      legend.title = element_blank())

# --- GRAPH 5: Absolute Humidity (dark colors for contrast) ---
ah_covs <- c("Absolute humidity")
ah_df <- filter(data_long, Covariate %in% ah_covs)

p_ah <- ggplot(ah_df, aes(x = Covariate, y = Mean, fill = Forest)) +
```

```
  geom_col(position = position_dodge(width = 0.8), width = 0.7) +
  geom_errorbar(aes(ymin = Mean - SE, ymax = Mean + SE),
                position = position_dodge(width = 0.8), width = 0.15, size = 0.7) +
  scale_fill_manual(values = c("Old Forest" = "#4a148c", "Secondary Forest" = "#ba68c8")) +
  labs(title = "", y = "Absolute humidity (g/m^3)", x = "") +
  theme_minimal(base_size = 13) +
  theme(axis.text.x = element_text(angle = 0, hjust = 0.5),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.line = element_line(color = "darkgrey", size = 0.5),
        axis.ticks = element_line(color = "darkgrey"),
        legend.position = "bottom",
        legend.title = element_blank())

# Display graphs in graphics device
print(p_tree)
print(p_dbh)
print(p_temp)
print(p_rh)
print(p_ah)

# Save graphs
ggsave("FigureA_TreeAbundanceRichness.png", plot = p_tree, width = 6, height = 5, dpi = 300)
ggsave("FigureB_TreeDiameter.png", plot = p_dbh, width = 6, height = 5, dpi = 300)
ggsave("FigureC_Temperature.png", plot = p_temp, width = 6, height = 5, dpi = 300)
ggsave("FigureD_RelativeHumidity.png", plot = p_rh, width = 6, height = 5, dpi = 300)
ggsave("FigureE_AbsoluteHumidity.png", plot = p_ah, width = 6, height = 5, dpi = 300)


# Small multiples graph
data_long_updated <- data_long %>%
  mutate(Covariate = case_when(
    Covariate == "Tree abundance" ~ "Tree abundance (n)",
    Covariate == "Tree richness" ~ "Tree richness (S)",
    Covariate == "Tree diameter (cm)" ~ "Tree diameter (cm)",
    Covariate == "Air temperature (°C)" ~ "Air temperature (°C)",
    Covariate == "Relative humidity (%)" ~ "Relative humidity (%)",
    Covariate == "Absolute humidity" ~ "Absolute humidity (g/m^3)",
    TRUE ~ Covariate
  )) %>%
  # Create factor with specified order
  mutate(Covariate = factor(Covariate,
                            levels = c("Air temperature (°C)",
                                       "Relative humidity (%)",
                                       "Absolute humidity (g/m^3)",
                                       "Tree richness (S)",
                                       "Tree abundance (n)",
                                       "Tree diameter (cm)")))
```

```
# Now create the plot with updated units and correct order
p_facets <- ggplot(data_long_updated, aes(x = Forest, y = Mean, fill = Forest)) +
 geom_col(width = 0.7) +
 geom_errorbar(aes(ymin = Mean - SE, ymax = Mean + SE), width = 0.2) +
 facet_wrap(~ Covariate, scales = "free_y", ncol = 3) +
 scale_fill_manual(values = c("Old Forest" = "#006400",  # Dark green
                              "Secondary Forest" = "#90EE90")) +  # Light green
 labs(title = "",
    y = "Mean ± SE",
    x = "") +
 theme_minimal() +
 theme(legend.position = "bottom",
      axis.text.x = element_blank(),  # Remove x-axis text
      axis.ticks.x = element_blank(), # Remove x-axis ticks
      strip.background = element_rect(fill = "grey90"),
      panel.grid.major = element_blank(),
      panel.grid.minor = element_blank(),
      legend.title = element_blank())  # Remove legend title

# Save at 400 DPI
ggsave("SmallMultiples_ForestCovariates_400dpi.tiff",
     plot = p_facets,
     width = 10,
     height = 8,
     dpi = 300)

# Also save as high-quality PDF (vector format)
ggsave("SmallMultiples_ForestCovariates.pdf",
     plot = p_facets,
     width = 10,
     height = 8)

print(p_facets)


# RADAR CHART
library(tibble)  # Make sure tibble is loaded for column_to_rownames

radar_data <- data_long %>%
 select(Covariate, Forest, Mean) %>%
 pivot_wider(names_from = Covariate, values_from = Mean) %>%
 as.data.frame() %>%  # Convert to data.frame first
 column_to_rownames("Forest")

# Normalize data for radar chart (0-1 scale)
normalize <- function(x) {
 (x - min(x, na.rm = TRUE)) / (max(x, na.rm = TRUE) - min(x, na.rm = TRUE))
```

```
}

radar_normalized <- as.data.frame(lapply(radar_data, normalize))
radar_normalized$Forest <- rownames(radar_data)

# Create radar chart
library(fmsb)

# Prepare data for fmsb (first row max, second row min, then actual data)
radar_plot_data <- rbind(rep(1, ncol(radar_normalized)-1),
                         rep(0, ncol(radar_normalized)-1),
                         radar_normalized[,-ncol(radar_normalized)])

# Color vector - using dark green for OF and light green for SF
colors_border <- c("#006400", "#90EE90")  # Dark green and light green
colors_in <- c(alpha("#006400", 0.3), alpha("#90EE90", 0.3))

# Create radar chart
radarchart(radar_plot_data,
        axistype = 1,
        pcol = colors_border,
        pfcol = colors_in,
        plwd = 2,
        cglcol = "grey",
        cglty = 1,
        axislabcol = "grey",
        caxislabels = seq(0, 1, 0.2),
        cglwd = 0.8,
        vlcex = 0.8)

legend("topright",
     legend = radar_normalized$Forest,
     bty = "n",
     pch = 20,
     col = colors_border,
     text.col = "black",  # Changed to black for better visibility
     cex = 1,
     pt.cex = 2)

# Save radar chart at 400 DPI
png("RadarChart_ForestCovariates_400dpi.png", width = 8, height = 8, units = "in", res = 400)
radarchart(radar_plot_data,
        axistype = 1,
        pcol = colors_border,
        pfcol = colors_in,
        plwd = 2,
        cglcol = "grey",
        cglty = 1,
```

```
        axislabcol = "grey",
        caxislabels = seq(0, 1, 0.2),
        cglwd = 0.8,
        vlcex = 0.8)

legend("topright",
     legend = radar_normalized$Forest,
     bty = "n",
     pch = 20,
     col = colors_border,
     text.col = "black",
     cex = 1,
     pt.cex = 2)
dev.off()

# RADAR CHART - IMPROVED VERSION
library(tibble)
library(tidyr)
library(dplyr)
library(scales)

radar_data <- data_long %>%
  select(Covariate, Forest, Mean) %>%
  pivot_wider(names_from = Covariate, values_from = Mean) %>%
  as.data.frame() %>%
  column_to_rownames("Forest")

# Normalize data for radar chart (0-1 scale)
normalize <- function(x) {
  (x - min(x, na.rm = TRUE)) / (max(x, na.rm = TRUE) - min(x, na.rm = TRUE))
}

radar_normalized <- as.data.frame(lapply(radar_data, normalize))
radar_normalized$Forest <- rownames(radar_data)

# Create radar chart
library(fmsb)

# Prepare data for fmsb (first row max, second row min, then actual data)
radar_plot_data <- rbind(rep(1, ncol(radar_normalized)-1),
                         rep(0, ncol(radar_normalized)-1),
                         radar_normalized[,-ncol(radar_normalized)])

# IMPROVED COLOR VECTOR - More contrasting colors
colors_border <- c("#006400", "#FF6B00")  # Dark green and orange for high contrast
colors_in <- c(alpha("#006400", 0.4), alpha("#FF6B00", 0.4))

# Create radar chart with larger size and better readability
```

```
par(mar = c(2, 2, 2, 2))  # Adjust margins
radarchart(radar_plot_data,
        axistype = 1,
        pcol = colors_border,
        pfcol = colors_in,
        plwd = 3,  # Thicker lines
        plty = 1,
        cglcol = "darkgray",  # Darker grid lines
        cglty = 1,
        axislabcol = "black",  # Black axis labels for readability
        caxislabels = seq(0, 1, 0.2),
        cglwd = 1.2,  # Thicker grid lines
        vlcex = 1.2,  # Larger variable labels
        calcex = 1.1, # Larger axis labels
        title = "Forest Covariates Comparison")  # Add title

legend("topright",
     legend = radar_normalized$Forest,
     bty = "o",  # Box around legend
     bg = "white",  # White background for legend
     pch = 20,
     col = colors_border,
     text.col = "black",
     cex = 1.3,  # Larger legend text
     pt.cex = 3)  # Larger legend points

# Save radar chart at 400 DPI with larger dimensions
png("RadarChart_ForestCovariates_400dpi.png", width = 12, height = 10, units = "in", res = 400)

# Set larger margins and recreate plot
par(mar = c(2, 2, 3, 2))
radarchart(radar_plot_data,
        axistype = 1,
        pcol = colors_border,
        pfcol = colors_in,
        plwd = 4,  # Even thicker lines for high resolution
        plty = 1,
        cglcol = "darkgray",
        cglty = 1,
        axislabcol = "black",
        caxislabels = seq(0, 1, 0.2),
        cglwd = 1.5,
        vlcex = 1.5,  # Larger variable labels
        calcex = 1.3, # Larger axis labels
        title = "Forest Covariates Comparison")

legend("topright",
     legend = radar_normalized$Forest,
```

```
    bty = "o",
    bg = "white",
    pch = 20,
    col = colors_border,
    text.col = "black",
    cex = 1.5,  # Larger legend text
    pt.cex = 3.5)  # Larger legend points

dev.off()
```