Daily-Level GAM Analysis of Monarch Butterfly Abundance

Kyle Nessen

2025-09-30

Table of contents

Introduction

This analysis investigates daily-level patterns in overwintering monarch butterfly abundance using Generalized Additive Models (GAMs). Unlike the 30-minute interval analysis, this approach aggregates data to daily summaries, examining how previous day's weather conditions affect butterfly abundance. The response variable is the 95th percentile of butterfly counts, providing a robust measure of daily peak abundance while being less sensitive to outliers than the maximum.

Setup

Load libraries and data:

```r
library(tidyverse)
library(mgcv)
library(lubridate)
library(plotly)
library(knitr)
library(DT)
library(here)
library(gratia)
library(patchwork)
library(corrplot)

# Load the daily lag analysis data
daily_data <- read_csv(here("data", "monarch_daily_lag_analysis.csv"))

# Create the square root transformed response variable early for use throughout
daily_data <- daily_data %>%
    mutate(
        butterfly_diff_95th_sqrt = ifelse(butterfly_diff_95th >= 0,
            sqrt(butterfly_diff_95th),
            -sqrt(-butterfly_diff_95th)
        )
    )
```

Data Exploration

Data Structure and Summary

```r
# Basic summary statistics
cat("Dataset dimensions:", nrow(daily_data), "rows x", ncol(daily_data), "columns\n")
```

```
Dataset dimensions: 103 rows x 46 columns
```

```
cat("Number of deployments:", n_distinct(daily_data$deployment_id), "\n")
```

```
Number of deployments: 7
```

```
cat("Date range:", min(daily_data$date_t), "to", max(daily_data$date_t), "\n\n")
```

```
Date range: 19680 to 19756
```

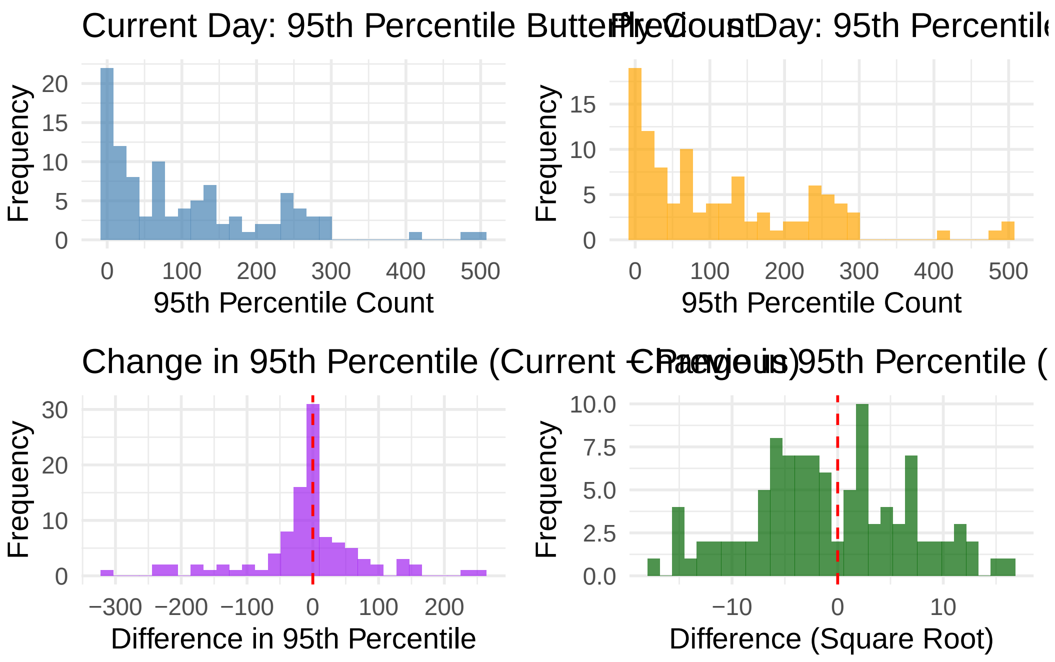
```
# Summary of key variables
summary_vars <- daily_data %>%
    select(
        butterflies_95th_percentile_t,
        butterflies_95th_percentile_t_1,
        butterfly_diff_95th,
        temp_max_t_1,
        temp_min_t_1,
        temp_at_max_count_t_1,
        wind_max_gust_t_1,
        sum_butterflies_direct_sun_t_1
    )

summary(summary_vars)
```

```
 butterflies_95th_percentile_t butterflies_95th_percentile_t_1
 Min.   :  0.00                Min.   :  0.0
 1st Qu.: 14.85                1st Qu.: 17.5
 Median : 70.05                Median : 77.0
 Mean   :107.41                Mean   :116.3
 3rd Qu.:166.95                3rd Qu.:199.5
 Max.   :499.00                Max.   :499.0

 butterfly_diff_95th  temp_max_t_1    temp_min_t_1    temp_at_max_count_t_1
 Min.   :-310.000    Min.   :14.00   Min.   : 3.000   Min.   : 5.00
 1st Qu.: -31.000    1st Qu.:16.00   1st Qu.: 7.000   1st Qu.:11.50
 Median :  -2.950    Median :18.00   Median :10.000   Median :14.00
 Mean   :  -8.919    Mean   :19.43   Mean   : 9.573   Mean   :13.37
 3rd Qu.:  18.000    3rd Qu.:22.00   3rd Qu.:12.000   3rd Qu.:15.50
 Max.   : 256.600    Max.   :37.00   Max.   :16.000   Max.   :25.00

 wind_max_gust_t_1 sum_butterflies_direct_sun_t_1
 Min.   :0.000     Min.   :   0.00  
 1st Qu.:2.750     1st Qu.:   2.00  
 Median :3.750     Median :  19.00  
 Mean   :3.718     Mean   :  94.77  
 3rd Qu.:4.500     3rd Qu.: 104.00  
 Max.   :7.200     Max.   :1122.00  
 NA's   :3                          
```

Response Variable Distribution

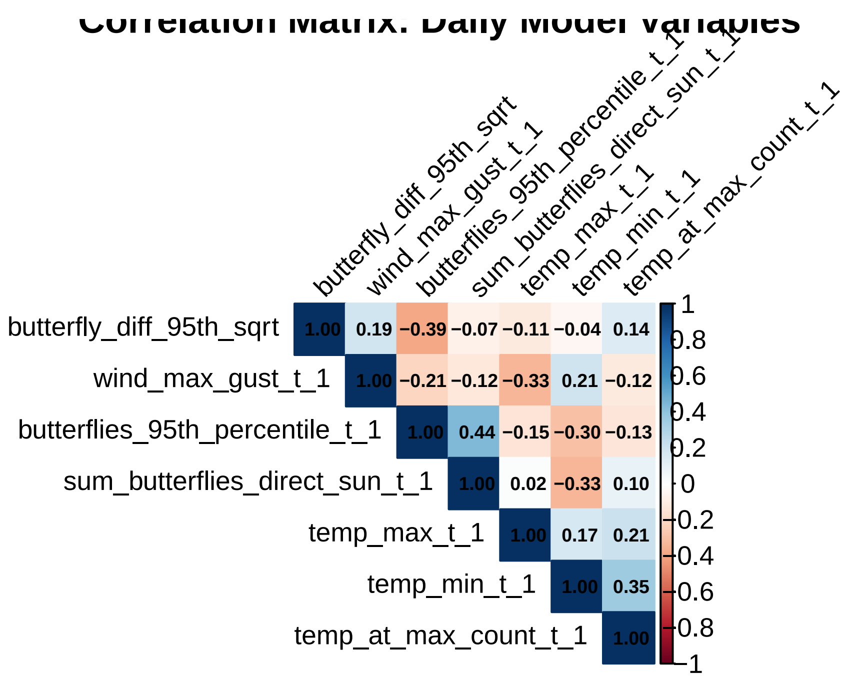
```r
library(gridExtra)

# Current day's 95th percentile
p1 <- ggplot(daily_data, aes(x = butterflies_95th_percentile_t)) +
    geom_histogram(bins = 30, fill = "steelblue", alpha = 0.7) +
    labs(
        title = "Current Day: 95th Percentile Butterfly Count",
        x = "95th Percentile Count", y = "Frequency"
    ) +
    theme_minimal()

# Previous day's 95th percentile
p2 <- ggplot(daily_data, aes(x = butterflies_95th_percentile_t_1)) +
    geom_histogram(bins = 30, fill = "orange", alpha = 0.7) +
    labs(
        title = "Previous Day: 95th Percentile Butterfly Count",
        x = "95th Percentile Count", y = "Frequency"
    ) +
    theme_minimal()

# Difference in 95th percentile
p3 <- ggplot(daily_data, aes(x = butterfly_diff_95th)) +
    geom_histogram(bins = 30, fill = "purple", alpha = 0.7) +
    geom_vline(xintercept = 0, linetype = "dashed", color = "red") +
    labs(
        title = "Change in 95th Percentile (Current - Previous)",
        x = "Difference in 95th Percentile", y = "Frequency"
    ) +
    theme_minimal()

# Square root transformed difference
```

```r
p4 <- ggplot(daily_data, aes(x = butterfly_diff_95th_sqrt)) +
    geom_histogram(bins = 30, fill = "darkgreen", alpha = 0.7) +
    geom_vline(xintercept = 0, linetype = "dashed", color = "red") +
    labs(
        title = "Change in 95th Percentile (Square Root Transformed)",
        x = "Difference (Square Root)", y = "Frequency"
    ) +
    theme_minimal()

grid.arrange(p1, p2, p3, p4, ncol = 2)
```



Correlation Analysis

```r
# Select model variables
model_vars <- daily_data %>%
    select(
        butterfly_diff_95th_sqrt,
        butterflies_95th_percentile_t_1,
        temp_max_t_1,
        temp_min_t_1,
        temp_at_max_count_t_1,
        wind_max_gust_t_1,
        sum_butterflies_direct_sun_t_1
    ) %>%
    na.omit()

# Correlation matrix
cor_matrix <- cor(model_vars)

# Create correlation plot
corrplot(cor_matrix,
    method = "color",
    type = "upper",
    order = "hclust",
    tl.cex = 0.8,
    tl.col = "black",
    tl.srt = 45,
    addCoef.col = "black",
    number.cex = 0.6,
    title = "Correlation Matrix: Daily Model Variables"
)
```



	butterfly_diff_95th_sqrt	wind_max_gust_t_1	butterflies_95th_percentile_t_1	sum_butterflies_direct_sun_t_1	temp_max_t_1	temp_min_t_1	temp_at_max_count_t_1
butterfly_diff_95th_sqrt	1.00	0.19	−0.39	−0.07	−0.11	−0.04	0.14
wind_max_gust_t_1		1.00	−0.21	−0.12	−0.33	0.21	−0.12
butterflies_95th_percentile_t_1			1.00	0.44	−0.15	−0.30	−0.13
sum_butterflies_direct_sun_t_1				1.00	0.02	−0.33	0.10
temp_max_t_1					1.00	0.17	0.21
temp_min_t_1						1.00	0.35
temp_at_max_count_t_1							1.00

```r
# Print correlation table
kable(round(cor_matrix, 3),
    caption = "Correlation Matrix for Daily Model Variables"
)
```

Table 1: Correlation Matrix for Daily Model Variables

	butterfly_diff_95th_sqrt	butterflies_95th_percentile_t_1	temp_max_t_1	temp_min_t_1	temp_at_max_count_t_1	wind_max_gust_t_1	sum_butterflies_direct_sun_
butterfly_diff_95th_sqrt	1.000	-0.389	-0.112	-0.042	0.145	0.193	-0.072
butterflies_95th_percentile_t_1	-0.389	1.000	-0.146	-0.299	-0.132	-0.211	0.442
temp_max_t_1	-0.112	-0.146	1.000	0.173	0.215	-0.334	0.016
temp_min_t_1	-0.042	-0.299	0.173	1.000	0.351	0.210	-0.331
temp_at_max_count_t_1	0.145	-0.132	0.215	0.351	1.000	-0.116	0.098
wind_max_gust_t_1	0.193	-0.211	-0.334	0.210	-0.116	1.000	-0.122
sum_butterflies_direct_sun_t_1	-0.072	0.442	0.016	-0.331	0.098	-0.122	1.000

Response Variable Normality Assessment

```r
library(nortest)

# First, identify all potential response variables in the dataset
# Exclude already-transformed variables to prevent double-transformation
response_candidates <- daily_data %>%
    select(contains("diff"), contains("butterfly")) %>%
    select(-contains("direct_sun"), -contains("sqrt"), -contains("cbrt"), -contains("log")) %
    names()

cat("Available response variable candidates:\n")
```

```
Available response variable candidates:
```

```r
print(response_candidates)
```

```
[1] "butterfly_diff"      "butterfly_diff_95th" "butterfly_diff_top3"
```

```r
# Define transformations to test
transformations <- list(
    "original" = function(x) x,
    "sqrt" = function(x) ifelse(x >= 0, sqrt(x), -sqrt(-x)) # Signed square root
)

# Function to calculate normality statistics
assess_normality <- function(x, var_name, transform_name) {
    # Remove NA values
    x_clean <- x[!is.na(x)]

    if (length(x_clean) < 10) {
        return(data.frame(
            Variable = var_name,
            Transformation = transform_name,
            N = length(x_clean),
            Mean = NA,
            SD = NA,
            Skewness = NA,
            Kurtosis = NA,
            Shapiro_p = NA,
            Anderson_p = NA,
            Normality_Score = 0
        ))
    }

    # Calculate statistics
    mean_val <- mean(x_clean)
    sd_val <- sd(x_clean)
    skew_val <- moments::skewness(x_clean)
    kurt_val <- moments::kurtosis(x_clean) - 3 # Excess kurtosis

    # Normality tests
    shapiro_p <- if (length(x_clean) <= 5000) shapiro.test(x_clean)$p.value else NA
    anderson_p <- tryCatch(nortest::ad.test(x_clean)$p.value, error = function(e) NA)

    # Create composite normality score (higher = more normal)
    # Based on: low absolute skewness, low absolute kurtosis, high p-values
    skew_score <- max(0, 1 - abs(skew_val) / 2) # Penalize skewness > 2
    kurt_score <- max(0, 1 - abs(kurt_val) / 4) # Penalize excess kurtosis > 4
    shapiro_score <- ifelse(is.na(shapiro_p), 0.5, shapiro_p)
    anderson_score <- ifelse(is.na(anderson_p), 0.5, anderson_p)
    # Weighted composite score
    normality_score <- (skew_score * 0.3 + kurt_score * 0.3 +
        shapiro_score * 0.2 + anderson_score * 0.2)

    return(data.frame(
        Variable = var_name,
        Transformation = transform_name,
        N = length(x_clean),
        Mean = round(mean_val, 3),
        SD = round(sd_val, 3),
        Skewness = round(skew_val, 3),
        Kurtosis = round(kurt_val, 3),
        Shapiro_p = ifelse(is.na(shapiro_p), NA, round(shapiro_p, 4)),
        Anderson_p = ifelse(is.na(anderson_p), NA, round(anderson_p, 4)),
        Normality_Score = round(normality_score, 4)
    ))
}

# Load required library for moments
library(moments)

# Apply transformations and assess normality for each response variable
normality_results <- list()

for (var_name in response_candidates) {
    if (var_name %in% names(daily_data)) {
        var_data <- daily_data[[var_name]]

        for (trans_name in names(transformations)) {
            trans_func <- transformations[[trans_name]]

            # Apply transformation
            transformed_data <- tryCatch(
                trans_func(var_data),
                error = function(e) rep(NA, length(var_data))
            )

            # Assess normality
            result <- assess_normality(transformed_data, var_name, trans_name)
            normality_results[[paste(var_name, trans_name, sep = "_")]] <- result
        }
    }
```

```r
}

# Combine results
normality_df <- do.call(rbind, normality_results)

# Rank by normality score
normality_ranking <- normality_df %>%
    arrange(desc(Normality_Score)) %>%
    filter(!is.na(Normality_Score)) %>%
    mutate(Rank = row_number()) %>%
    select(
        Rank, Variable, Transformation, N, Mean, SD, Skewness, Kurtosis,
        Shapiro_p, Anderson_p, Normality_Score
    )

# Display top 15 most normal distributions
cat("Top 15 most normal response variable transformations:\n\n")
```

```
Top 15 most normal response variable transformations:
```

```r
kable(head(normality_ranking, 15),
    caption = "Response variables ranked by normality (higher score = more normal)"
)
```

Table 2: Response variables ranked by normality (higher score = more normal)

	Rank	Variable	Transformation	N	Mean	SD	Skewness	Kurtosis	Shapiro_p	Anderson_p	Normality_Score
butterfly_diff_95th_sqrt	1	butterfly_diff_95th	sqrt	103	-0.809	7.382	0.021	-0.467	0.6501	0.5918	0.8102
butterfly_diff_top3_sqrt	2	butterfly_diff_top3	sqrt	103	-0.751	7.379	0.039	-0.436	0.6273	0.5818	0.8033
butterfly_diff_sqrt	3	butterfly_diff	sqrt	103	-1.148	8.033	0.238	-0.117	0.6179	0.3799	0.7552
butterfly_diff_top3_original	4	butterfly_diff_top3	original	103	-8.547	87.141	-0.026	2.983	0.0000	0.0000	0.3724
butterfly_diff_95th_original	5	butterfly_diff_95th	original	103	-8.919	86.928	-0.402	2.525	0.0000	0.0000	0.3502
butterfly_diff_original	6	butterfly_diff	original	103	-10.097	108.337	0.389	5.076	0.0000	0.0000	0.2417

```r
# Create summary by variable
variable_summary <- normality_ranking %>%
    group_by(Variable) %>%
    slice_max(Normality_Score, n = 1) %>%
    ungroup() %>%
    arrange(desc(Normality_Score)) %>%
    select(Variable,
        Best_Transformation = Transformation, Best_Score = Normality_Score,
        Skewness, Kurtosis, Shapiro_p
    )

cat("\n\nBest transformation for each response variable:\n")
```

```
Best transformation for each response variable:
```

```r
kable(variable_summary,
    caption = "Best transformation for each response variable"
)
```

Table 3: Best transformation for each response variable

Variable	Best_Transformation	Best_Score	Skewness	Kurtosis	Shapiro_p
butterfly_diff_95th	sqrt	0.8102	0.021	-0.467	0.6501
butterfly_diff_top3	sqrt	0.8033	0.039	-0.436	0.6273
butterfly_diff	sqrt	0.7552	0.238	-0.117	0.6179

```r
cat("\n\nUsing the best response variable transformation: butterfly_diff_95th_sqrt\n")
```

```
Using the best response variable transformation: butterfly_diff_95th_sqrt
```

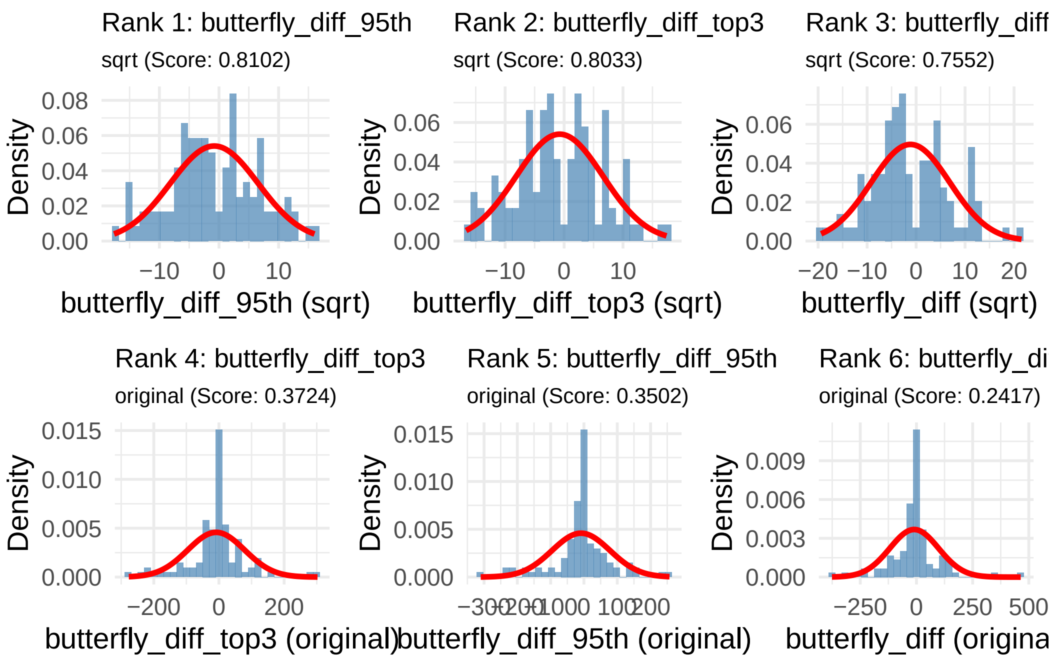
```r
cat("Summary of transformed response variable:\n")
```

```
Summary of transformed response variable:
```

```r
print(summary(daily_data$butterfly_diff_95th_sqrt))
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
-17.6068  -5.5649  -1.7176  -0.8088   4.2426  16.0187 
```

```r
# Visualize all transformations (original and sqrt)
top_transformations <- normality_ranking

plots <- list()
for (i in 1:nrow(top_transformations)) {
    row <- top_transformations[i, ]
    var_name <- row$Variable
    trans_name <- row$Transformation

    if (var_name %in% names(daily_data)) {
        var_data <- daily_data[[var_name]]
        trans_func <- transformations[[trans_name]]
        transformed_data <- trans_func(var_data)

        # Create histogram with normal overlay
        p <- ggplot(data.frame(x = transformed_data), aes(x = x)) +
            geom_histogram(aes(y = after_stat(density)),
                bins = 30,
                fill = "steelblue", alpha = 0.7
            ) +
            stat_function(
                fun = dnorm,
                args = list(
                    mean = mean(transformed_data, na.rm = TRUE),
                    sd = sd(transformed_data, na.rm = TRUE)
                ),
                color = "red", size = 1
            ) +
            labs(
                title = paste0("Rank ", i, ": ", var_name),
                subtitle = paste0(trans_name, " (Score: ", row$Normality_Score, ")"),
                x = paste0(var_name, " (", trans_name, ")"),
                y = "Density"
            ) +
            theme_minimal() +
            theme(
                plot.title = element_text(size = 10),
                plot.subtitle = element_text(size = 8)
            )

        plots[[i]] <- p
    }
}

# Arrange plots in grid
normality_grid <- do.call(grid.arrange, c(plots, ncol = 3))
```



```
normality_grid
```

```
TableGrob (2 x 3) "arrange": 6 grobs
  z     cells    name           grob
1 1 (1-1,1-1) arrange gtable[layout]
2 2 (1-1,2-2) arrange gtable[layout]
3 3 (1-1,3-3) arrange gtable[layout]
4 4 (2-2,1-1) arrange gtable[layout]
5 5 (2-2,2-2) arrange gtable[layout]
6 6 (2-2,3-3) arrange gtable[layout]
```

```
# Export normality visualization for results write-up
png(here("analysis", "reports", "figures", "response_variable_normality.png"),
    width = 14, height = 10, units = "in", res = 300)
do.call(grid.arrange, c(plots, ncol = 3))
dev.off()
```

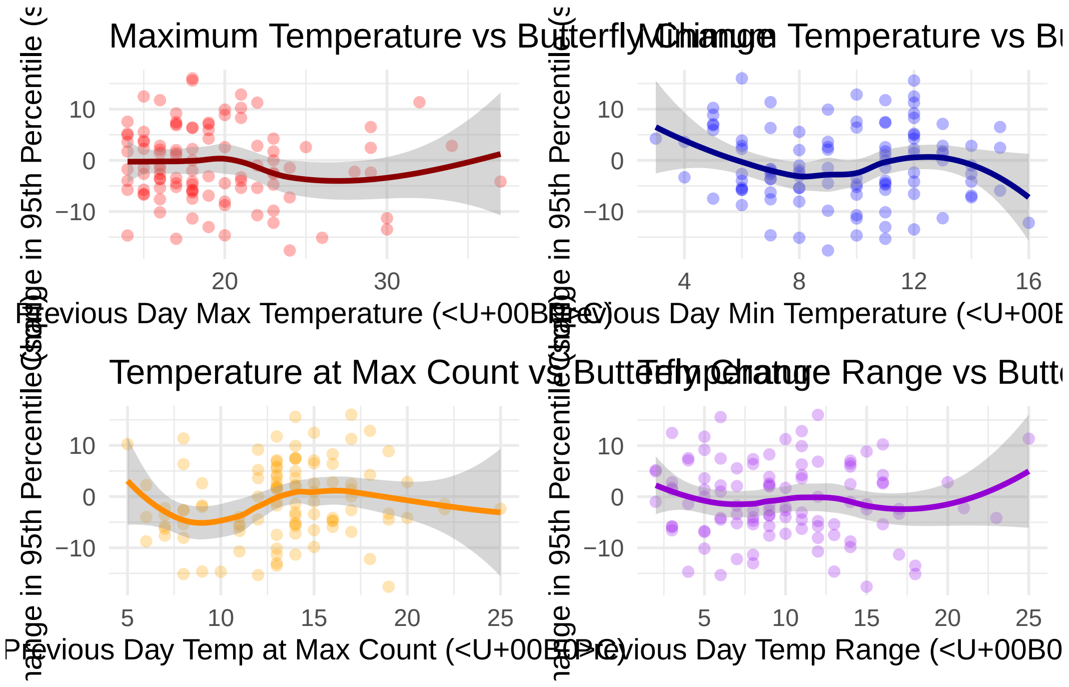
```
pdf
  2
```

```
cat("Exported normality visualization to: analysis/reports/figures/response_variable_normali
```

```
Exported normality visualization to: analysis/reports/figures/response_variable_normality.png
```

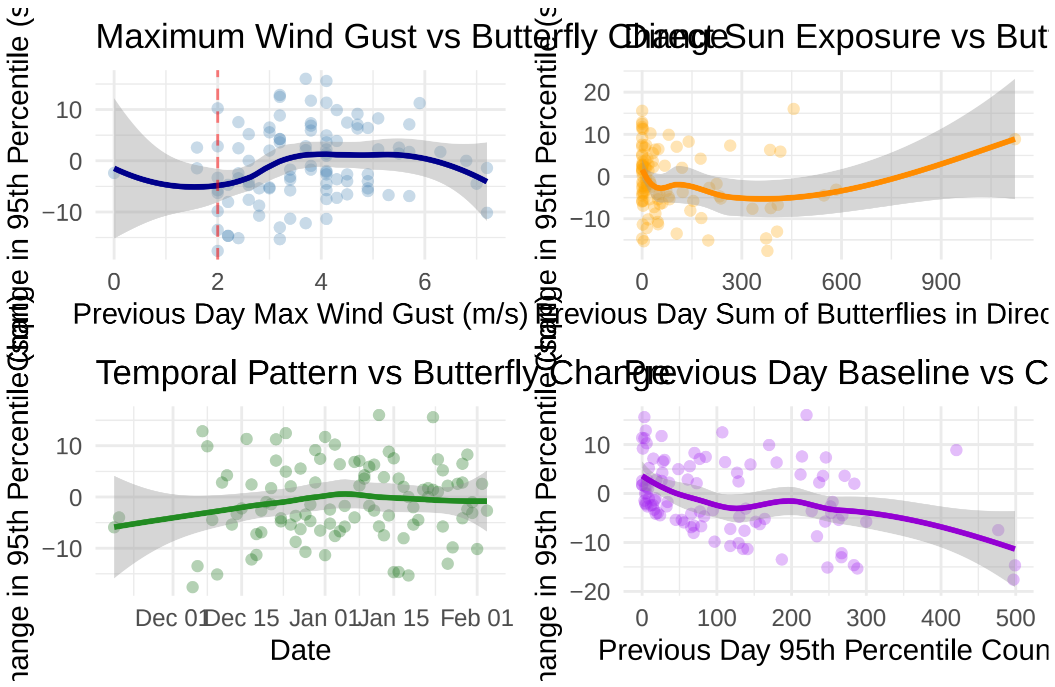
Temperature Patterns

```
# Temperature relationships
p1 <- ggplot(daily_data, aes(x = temp_max_t_1, y = butterfly_diff_95th_sqrt)) +
    geom_point(alpha = 0.3, color = "red") +
    geom_smooth(method = "loess", se = TRUE, color = "darkred") +
    labs(
        title = "Maximum Temperature vs Butterfly Change",
        x = "Previous Day Max Temperature (°C)",
        y = "Change in 95th Percentile (sqrt)"
    ) +
    theme_minimal()

p2 <- ggplot(daily_data, aes(x = temp_min_t_1, y = butterfly_diff_95th_sqrt)) +
    geom_point(alpha = 0.3, color = "blue") +
    geom_smooth(method = "loess", se = TRUE, color = "darkblue") +
    labs(
        title = "Minimum Temperature vs Butterfly Change",
        x = "Previous Day Min Temperature (°C)",
        y = "Change in 95th Percentile (sqrt)"
    ) +
    theme_minimal()

p3 <- ggplot(daily_data, aes(x = temp_at_max_count_t_1, y = butterfly_diff_95th_sqrt)) +
    geom_point(alpha = 0.3, color = "orange") +
    geom_smooth(method = "loess", se = TRUE, color = "darkorange") +
    labs(
        title = "Temperature at Max Count vs Butterfly Change",
        x = "Previous Day Temp at Max Count (°C)",
        y = "Change in 95th Percentile (sqrt)"
    ) +
    theme_minimal()

# Temperature range
daily_data <- daily_data %>%
    mutate(temp_range_t_1 = temp_max_t_1 - temp_min_t_1)

p4 <- ggplot(daily_data, aes(x = temp_range_t_1, y = butterfly_diff_95th_sqrt)) +
    geom_point(alpha = 0.3, color = "purple") +
    geom_smooth(method = "loess", se = TRUE, color = "darkviolet") +
    labs(
        title = "Temperature Range vs Butterfly Change",
        x = "Previous Day Temp Range (°C)",
        y = "Change in 95th Percentile (sqrt)"
    ) +
    theme_minimal()

grid.arrange(p1, p2, p3, p4, ncol = 2)
```



Wind and Sun Exposure

```
# Wind effect
p1 <- ggplot(daily_data, aes(x = wind_max_gust_t_1, y = butterfly_diff_95th_sqrt)) +
    geom_point(alpha = 0.3, color = "steelblue") +
    geom_smooth(method = "loess", se = TRUE, color = "darkblue") +
    geom_vline(xintercept = 2, linetype = "dashed", color = "red", alpha = 0.5) +
    labs(
        title = "Maximum Wind Gust vs Butterfly Change",
        x = "Previous Day Max Wind Gust (m/s)",
        y = "Change in 95th Percentile (sqrt)"
    ) +
    theme_minimal()

# Sun exposure
p2 <- ggplot(daily_data, aes(x = sum_butterflies_direct_sun_t_1, y = butterfly_diff_95th_sqr
    geom_point(alpha = 0.3, color = "orange") +
    geom_smooth(method = "loess", se = TRUE, color = "darkorange") +
    labs(
        title = "Direct Sun Exposure vs Butterfly Change",
        x = "Previous Day Sum of Butterflies in Direct Sun",
        y = "Change in 95th Percentile (sqrt)"
    ) +
    theme_minimal()

# Note: Seasonal progression will be handled via temporal autocorrelation
# rather than as a fixed effect
p3 <- ggplot(daily_data, aes(x = date_t, y = butterfly_diff_95th_sqrt)) +
    geom_point(alpha = 0.3, color = "darkgreen") +
    geom_smooth(method = "loess", se = TRUE, color = "forestgreen") +
    labs(
        title = "Temporal Pattern vs Butterfly Change",
        x = "Date",
        y = "Change in 95th Percentile (sqrt)"
    ) +
    theme_minimal()

# Previous day baseline
p4 <- ggplot(daily_data, aes(x = butterflies_95th_percentile_t_1, y = butterfly_diff_95th_sq
    geom_point(alpha = 0.3, color = "purple") +
    geom_smooth(method = "loess", se = TRUE, color = "darkviolet") +
    labs(
        title = "Previous Day Baseline vs Change",
        x = "Previous Day 95th Percentile Count",
        y = "Change in 95th Percentile (sqrt)"
    ) +
    theme_minimal()

grid.arrange(p1, p2, p3, p4, ncol = 2)
```



Data Preparation

```
# Remove missing values and prepare modeling dataset
model_data <- daily_data %>%
    filter(
        !is.na(butterfly_diff_95th_sqrt),
        !is.na(butterflies_95th_percentile_t_1),
        !is.na(temp_max_t_1),
        !is.na(temp_min_t_1),
        !is.na(temp_at_max_count_t_1),
        !is.na(wind_max_gust_t_1),
        !is.na(sum_butterflies_direct_sun_t_1),
        !is.na(deployment_id)
    ) %>%
    # Create standardized versions for interpretation
    mutate(
        wind_max_gust_std = scale(wind_max_gust_t_1)[, 1],
        temp_max_std = scale(temp_max_t_1)[, 1],
        temp_min_std = scale(temp_min_t_1)[, 1],
        temp_at_max_std = scale(temp_at_max_count_t_1)[, 1],
        sun_exposure_std = scale(sum_butterflies_direct_sun_t_1)[, 1],
        baseline_std = scale(butterflies_95th_percentile_t_1)[, 1],
        # Note: day_sequence is now provided by the data preparation script
        # Each deployment has its own day counter starting from 1
    )

cat("Clean dataset has", nrow(model_data), "observations\n")
```

```
Clean dataset has 100 observations
```

```
cat("Number of unique deployment days:", n_distinct(paste(model_data$deployment_id, model_da
```

```
Number of unique deployment days: 100
```

Modeling Strategy

Our modeling approach for daily-level data tests both **absolute effects** and **proportional effects** of environmental variables on butterfly abundance changes:

1. **Response Variable**: `butterfly_diff_95th_sqrt` - square root transformed difference in 95th percentile butterfly counts between consecutive days (selected as the most normal transformation)
2. **Two Model Sets**:

   M Models (Absolute Effects): Test whether environmental variables have direct effects on absolute changes in abundance:

   - Do NOT include previous day's butterfly count
   - Test if weather has consistent magnitude effects regardless of population size

   B Models (Proportional/Density-Dependent Effects): Test whether environmental effects depend on baseline population:

- Include `butterflies_95th_percentile_t_1` as a covariate
- Test if weather effects scale with population size
- Include interactions between baseline count and environmental variables

3. **Fixed Effects** (tested in various combinations):

   - Temperature variables: max, min, and temperature at max count
   - Wind: maximum gust from previous day
   - Sun exposure: sum of butterflies in direct sun from previous day
   - Previous day baseline: 95th percentile count (B models only)

4. **Random Effects**:

   - Deployment ID (random intercept)
   - AR1 temporal autocorrelation within deployments using `day_sequence | deployment_id`

5. **Correlation Structures**:

   - No correlation (baseline)
   - AR1 within deployments to account for temporal autocorrelation

This dual approach allows us to distinguish between: - **Absolute effects**: Environmental variables cause fixed-magnitude changes regardless of population size - **Proportional effects**: Environmental impacts scale with the existing population (density-dependence)

Model Building and Selection

```r
library(nlme)

# Define random effects structure with temporal autocorrelation
# We'll test different correlation structures
random_structure <- list(deployment_id = ~1)

# Define correlation structures to test
correlation_structures <- list(
    "no_corr" = NULL, # No temporal correlation
    "AR1" = corAR1(form = ~ day_sequence | deployment_id) # AR1 within deployments
)

# Model specifications for AIC comparison - WITHOUT previous day baseline
model_specs <- list(
    # Null model
    "M1" = "butterfly_diff_95th_sqrt ~ 1",

    # Single predictor models (linear)
    "M2" = "butterfly_diff_95th_sqrt ~ wind_max_gust_t_1",
    "M3" = "butterfly_diff_95th_sqrt ~ temp_max_t_1",
    "M4" = "butterfly_diff_95th_sqrt ~ temp_min_t_1",
    "M5" = "butterfly_diff_95th_sqrt ~ temp_at_max_count_t_1",
    "M6" = "butterfly_diff_95th_sqrt ~ sum_butterflies_direct_sun_t_1",

    # Temperature combinations (linear)
    "M8" = "butterfly_diff_95th_sqrt ~ temp_max_t_1 + temp_min_t_1",
    "M9" = "butterfly_diff_95th_sqrt ~ temp_max_t_1 + temp_at_max_count_t_1",
    "M10" = "butterfly_diff_95th_sqrt ~ temp_min_t_1 + temp_at_max_count_t_1",
    "M11" = "butterfly_diff_95th_sqrt ~ temp_max_t_1 + temp_min_t_1 + temp_at_max_count_t_1"

    # Two-variable combinations
    "M12" = "butterfly_diff_95th_sqrt ~ wind_max_gust_t_1 + temp_max_t_1",
    "M13" = "butterfly_diff_95th_sqrt ~ wind_max_gust_t_1 + temp_min_t_1",
    "M14" = "butterfly_diff_95th_sqrt ~ wind_max_gust_t_1 + temp_at_max_count_t_1",
    "M15" = "butterfly_diff_95th_sqrt ~ wind_max_gust_t_1 + sum_butterflies_direct_sun_t_1",
    "M16" = "butterfly_diff_95th_sqrt ~ temp_at_max_count_t_1 + sum_butterflies_direct_sun_t

    # Full models with various temperature specs (linear)
    "M17" = "butterfly_diff_95th_sqrt ~ temp_max_t_1 + wind_max_gust_t_1 + sum_butterflies_di
    "M18" = "butterfly_diff_95th_sqrt ~ temp_min_t_1 + wind_max_gust_t_1 + sum_butterflies_di
    "M19" = "butterfly_diff_95th_sqrt ~ temp_at_max_count_t_1 + wind_max_gust_t_1 + sum_butte
    "M20" = "butterfly_diff_95th_sqrt ~ temp_max_t_1 + temp_min_t_1 + wind_max_gust_t_1 + sum
    "M21" = "butterfly_diff_95th_sqrt ~ temp_max_t_1 + temp_min_t_1 + temp_at_max_count_t_1 +

    # Smooth terms models - single predictors
    "M24" = "butterfly_diff_95th_sqrt ~ s(wind_max_gust_t_1)",
    "M25" = "butterfly_diff_95th_sqrt ~ s(temp_max_t_1)",
    "M26" = "butterfly_diff_95th_sqrt ~ s(temp_min_t_1)",
    "M27" = "butterfly_diff_95th_sqrt ~ s(temp_at_max_count_t_1)",
    "M28" = "butterfly_diff_95th_sqrt ~ s(sum_butterflies_direct_sun_t_1)",

    # Smooth terms - combinations
    "M30" = "butterfly_diff_95th_sqrt ~ s(temp_max_t_1) + s(temp_min_t_1)",
    "M31" = "butterfly_diff_95th_sqrt ~ s(temp_at_max_count_t_1) + s(wind_max_gust_t_1)",
    "M32" = "butterfly_diff_95th_sqrt ~ s(temp_at_max_count_t_1) + s(sum_butterflies_direct_s
    "M33" = "butterfly_diff_95th_sqrt ~ s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_
    # Complex smooth models
    "M34" = "butterfly_diff_95th_sqrt ~ s(temp_at_max_count_t_1) + s(wind_max_gust_t_1) + s(s
    "M35" = "butterfly_diff_95th_sqrt ~ s(temp_max_t_1) + s(temp_min_t_1) + s(wind_max_gust_t
    "M37" = "butterfly_diff_95th_sqrt ~ s(temp_max_t_1) + s(temp_min_t_1) + s(temp_at_max_cou

    # Mixed linear and smooth
    "M38" = "butterfly_diff_95th_sqrt ~ temp_at_max_count_t_1 + s(wind_max_gust_t_1) + s(sum_
    "M39" = "butterfly_diff_95th_sqrt ~ s(temp_at_max_count_t_1) + wind_max_gust_t_1 + sum_bu
    "M40" = "butterfly_diff_95th_sqrt ~ s(temp_at_max_count_t_1) + wind_max_gust_t_1 + s(sum_

    # Interaction models (without baseline)
    "M41" = "butterfly_diff_95th_sqrt ~ temp_at_max_count_t_1 * wind_max_gust_t_1",
    "M42" = "butterfly_diff_95th_sqrt ~ temp_at_max_count_t_1 * sum_butterflies_direct_sun_t_
    "M43" = "butterfly_diff_95th_sqrt ~ wind_max_gust_t_1 * sum_butterflies_direct_sun_t_1",
    "M44" = "butterfly_diff_95th_sqrt ~ temp_at_max_count_t_1 * wind_max_gust_t_1 + sum_butte
    "M45" = "butterfly_diff_95th_sqrt ~ temp_at_max_count_t_1 + wind_max_gust_t_1 * sum_butte
    "M46" = "butterfly_diff_95th_sqrt ~ temp_at_max_count_t_1 * wind_max_gust_t_1 * sum_butte

    # Temperature range models
    "M47" = "butterfly_diff_95th_sqrt ~ I(temp_max_t_1 - temp_min_t_1)",
    "M48" = "butterfly_diff_95th_sqrt ~ I(temp_max_t_1 - temp_min_t_1) + wind_max_gust_t_1",
    "M49" = "butterfly_diff_95th_sqrt ~ s(I(temp_max_t_1 - temp_min_t_1))",
    "M50" = "butterfly_diff_95th_sqrt ~ s(I(temp_max_t_1 - temp_min_t_1)) + s(wind_max_gust_t

    # ===== MODELS WITH PREVIOUS DAY BASELINE =====
    # All models below include butterflies_95th_percentile_t_1 to test proportional effects

    # Baseline-only model
    "B1" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1",

    # Single predictor models + baseline (linear)
    "B2" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + wind_max_gust_t_1",
    "B3" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_max_t_1",
    "B4" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_min_t_1",
    "B5" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_at_max_count_t_
    "B6" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + sum_butterflies_dire

    # Temperature combinations + baseline (linear)
    "B8" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_max_t_1 + temp_
    "B9" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_max_t_1 + temp_
    "B10" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_min_t_1 + temp
    "B11" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_max_t_1 + temp
```

```
# Two-variable combinations + baseline
"B12" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + wind_max_gust_t_1 +
"B13" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + wind_max_gust_t_1 +
"B14" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + wind_max_gust_t_1 +
"B15" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + wind_max_gust_t_1 +
"B16" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_at_max_count_

# Full models with various temperature specs + baseline (linear)
"B17" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_max_t_1 + wind
"B18" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_min_t_1 + wind
"B19" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_at_max_count_
"B20" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_max_t_1 + temp
"B21" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_max_t_1 + temp

# Smooth terms models - single predictors + baseline
"B24" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(wind_max_gust_t_
"B25" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_max_t_1)",
"B26" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_min_t_1)",
"B27" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_at_max_coun
"B28" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(sum_butterflies_

# Smooth baseline + other predictors
"B29" = "butterfly_diff_95th_sqrt ~ s(butterflies_95th_percentile_t_1)",
"B29a" = "butterfly_diff_95th_sqrt ~ s(butterflies_95th_percentile_t_1) + wind_max_gust_
"B29b" = "butterfly_diff_95th_sqrt ~ s(butterflies_95th_percentile_t_1) + temp_at_max_co
"B29c" = "butterfly_diff_95th_sqrt ~ s(butterflies_95th_percentile_t_1) + s(wind_max_gus
"B29d" = "butterfly_diff_95th_sqrt ~ s(butterflies_95th_percentile_t_1) + s(temp_at_max_

# Smooth terms - combinations + baseline
"B30" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_max_t_1) +
"B31" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_at_max_coun
"B32" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_at_max_coun
"B33" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(wind_max_gust_t_

# Complex smooth models + baseline
"B34" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_at_max_coun
"B35" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_max_t_1) +
"B37" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_max_t_1) +

# Mixed linear and smooth + baseline
"B38" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_at_max_count_
"B39" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_at_max_coun
    "B40" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_at_max_coun

    # Interaction models with baseline
    "B41" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_at_max_count_
    "B42" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_at_max_count_
    "B43" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + wind_max_gust_t_1 
    "B44" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_at_max_count_
    "B45" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_at_max_count_
    "B46" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + temp_at_max_count_

    # Temperature range models + baseline
    "B47" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + I(temp_max_t_1 - te
    "B48" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + I(temp_max_t_1 - te
    "B49" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(I(temp_max_t_1 -
    "B50" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(I(temp_max_t_1 -

    # Interaction with baseline (testing if environmental effects depend on population size)
    "B51" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 * wind_max_gust_t_1"
    "B52" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 * temp_at_max_count_
    "B53" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 * sum_butterflies_di
    "B54" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 * wind_max_gust_t_1 
    "B55" = "butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 * temp_at_max_count_
)

cat("Total models to fit:", length(model_specs), "\n")
```

```
Total models to fit: 100
```

```
cat("- M models (M1-M50):", sum(grepl("^M", names(model_specs))), "models WITHOUT previous da
```

```
- M models (M1-M50): 45 models WITHOUT previous day baseline
```

```
cat("- B models (B1-B55):", sum(grepl("^B", names(model_specs))), "models WITH previous day b
```

```
- B models (B1-B55): 55 models WITH previous day baseline
```

Model Fitting

```
# Function to safely fit models with correlation structures
fit_model_safely <- function(formula_str, data, correlation = NULL, corr_name = "no_corr") {
    tryCatch(
        {
            formula_obj <- as.formula(formula_str)

            # Fit the model with or without correlation structure
            if (is.null(correlation)) {
                model <- gamm(formula_obj,
                    data = data,
                    random = random_structure,
                    method = "REML"
                )
            } else {
                model <- gamm(formula_obj,
                    data = data,
                    random = random_structure,
                    correlation = correlation,
                    method = "REML"
                )
            }

            # Add correlation structure name to the model for tracking
            model$correlation_structure <- corr_name
            return(model)
        },
        error = function(e) {
            message("Failed to fit model: ", formula_str, " with correlation: ", corr_name)
            message("Error: ", e$message)
            return(NULL)
        }
    )
}

# Fit all models with different correlation structures
cat("Fitting models...\n")
```

```
Fitting models...
```

```r
fitted_models <- list()

# Fit each model specification with each correlation structure
for (model_name in names(model_specs)) {
    formula_str <- model_specs[[model_name]]

    for (corr_name in names(correlation_structures)) {
        corr_struct <- correlation_structures[[corr_name]]

        # Create unique model name with correlation structure
        full_model_name <- paste(model_name, corr_name, sep = "_")

        fitted_models[[full_model_name]] <- fit_model_safely(
            formula_str, model_data, corr_struct, corr_name
        )
    }
}

# Remove failed models
successful_models <- fitted_models[!map_lgl(fitted_models, is.null)]
cat(
    "Successfully fitted", length(successful_models), "out of",
    length(model_specs), "models\n"
)
```

```
Successfully fitted 200 out of 100 models
```

Model Comparison

```r
# Extract AIC values
aic_results <- map_dfr(names(successful_models), function(full_model_name) {
    model <- successful_models[[full_model_name]]

    # Parse model name and correlation structure
    name_parts <- strsplit(full_model_name, "_")[[1]]
    corr_suffix <- name_parts[length(name_parts)]
    base_model_name <- paste(name_parts[-length(name_parts)], collapse = "_")

    # Get the formula from the base model name
    formula_str <- model_specs[[base_model_name]]
    if (is.null(formula_str)) {
        formula_str <- "Unknown formula"
    }

    data.frame(
        Model = full_model_name,
        Base_Model = base_model_name,
        Correlation = corr_suffix,
        Formula = formula_str,
        AIC = AIC(model$lme),
        LogLik = logLik(model$lme)[1],
        df = attr(logLik(model$lme), "df"),
        stringsAsFactors = FALSE
    )
}) %>%
    arrange(AIC) %>%
    mutate(
        Delta_AIC = AIC - min(AIC),
        AIC_weight = exp(-0.5 * Delta_AIC) / sum(exp(-0.5 * Delta_AIC))
    )

# Display top 10 models
top_10_table <- aic_results %>%
    head(10) %>%
    select(Model, Correlation, AIC, Delta_AIC, AIC_weight, df)

top_10_table %>%
    kable(digits = 3, caption = "Top 10 models by AIC")
```

Table 4: Top 10 models by AIC

Model	Correlation	AIC	Delta_AIC	AIC_weight	df
B33_AR1	AR1	668.401	0.000	0.148	9
B29c_AR1	AR1	668.671	0.270	0.129	8
B28_AR1	AR1	669.101	0.700	0.104	7
B35_AR1	AR1	669.573	1.172	0.082	13
B37_AR1	AR1	669.594	1.193	0.081	15
B29_AR1	AR1	669.685	1.284	0.078	6
B34_AR1	AR1	670.016	1.615	0.066	11
B29a_AR1	AR1	670.504	2.103	0.052	7
B38_AR1	AR1	670.691	2.289	0.047	10

Model	Correlation	AIC	Delta_AIC	AIC_weight	df
B29d_AR1	AR1	670.864	2.463	0.043	8

```r
# Show model formulas for top 5
cat("\nTop 5 model specifications:\n")
```

```
Top 5 model specifications:
```

```r
top_5_formulas <- head(aic_results, 5) %>%
    select(Base_Model, Correlation, Formula, Delta_AIC)

top_5_formulas %>%
    kable(digits = 3)
```

Base_Model	Correlation	Formula	Delta_AIC
B33	AR1	butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_t_1)	0.000
B29c	AR1	butterfly_diff_95th_sqrt ~ s(butterflies_95th_percentile_t_1) + s(wind_max_gust_t_1)	0.270
B28	AR1	butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(sum_butterflies_direct_sun_t_1)	0.700
B35	AR1	butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_max_t_1) + s(temp_min_t_1) + s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_t_1)	1.172
B37	AR1	butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_max_t_1) + s(temp_min_t_1) + s(temp_at_max_count_t_1) + s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_t_1)	1.193

```r
# Export model comparison tables as CSV for results write-up
write_csv(
    aic_results,
    here("analysis", "reports", "figures", "all_models_aic_table.csv")
)

write_csv(
    top_10_table,
    here("analysis", "reports", "figures", "top_10_models.csv")
)

cat("Exported AIC tables to: analysis/reports/figures/\n")
```

```
Exported AIC tables to: analysis/reports/figures/
```

```
cat("Note: strong_support_models will be exported after it's created\n")
```

```
Note: strong_support_models will be exported after it's created
```

Best Model Analysis

```
# Get the best model
best_model_name <- aic_results$Model[1]
best_model <- successful_models[[best_model_name]]

cat("Best model:", best_model_name, "\n")
```

```
Best model: B33_AR1
```

```
cat("Formula:", aic_results$Formula[1], "\n\n")
```

```
Formula: butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(wind_max_gust_t_1) +
```

```
# Model summary
best_model_summary <- summary(best_model$gam)
print(best_model_summary)
```

```


Family: gaussian
Link function: identity

Formula:
butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 +
    s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_t_1)
```

```
Parametric coefficients:
                                 Estimate Std. Error t value Pr(>|t|)
(Intercept)                      3.444416   1.263453   2.726  0.00766 **
butterflies_95th_percentile_t_1 -0.037703   0.006972  -5.408 4.95e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                                    edf Ref.df     F p-value
s(wind_max_gust_t_1)              2.466  2.466 2.725 0.08649 .
s(sum_butterflies_direct_sun_t_1) 2.918  2.918 6.122 0.00245 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.226
  Scale est. = 43.072    n = 100
```

```r
# Calculate R-squared
r_squared <- best_model_summary$r.sq
dev_explained <- best_model_summary$dev.expl

cat("\n\nModel Performance:\n")
```

```
Model Performance:
```

```r
cat("R-squared:", round(r_squared, 4), "\n")
```

```
R-squared: 0.2264
```

```r
cat("Deviance explained:", round(dev_explained * 100, 2), "%\n")
```

```
Deviance explained:  %
```

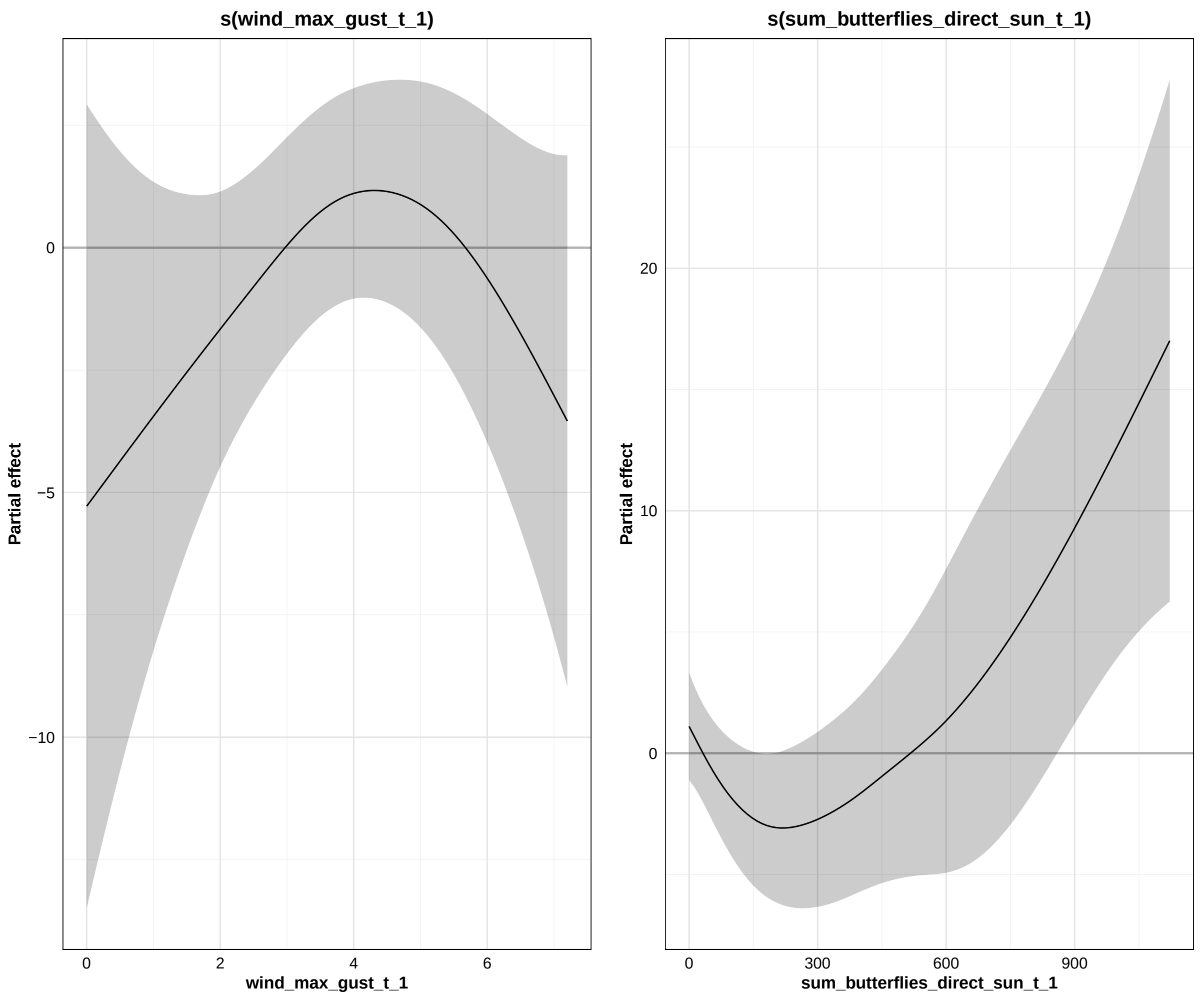
```r
# Export best model summary info for results write-up
# Use list-column approach to avoid vector length issues
best_model_info <- tribble(
    ~Metric, ~Value,
    "Model_Name", as.character(best_model_name)[1],
    "Formula", as.character(aic_results$Formula[1])[1],
    "Correlation", as.character(aic_results$Correlation[1])[1],
    "AIC", as.character(round(aic_results$AIC[1], 3)),
    "Delta_AIC", "0",
    "AIC_Weight", as.character(round(aic_results$AIC_weight[1], 4)),
    "R_squared", as.character(round(r_squared, 4)),
    "Deviance_Explained", as.character(round(dev_explained * 100, 2)),
    "N_obs", as.character(nrow(model_data))
)

write_csv(
    best_model_info,
    here("analysis", "reports", "figures", "best_model_summary.csv")
)

cat("Exported best model summary to: analysis/reports/figures/best_model_summary.csv\n")
```

```
Exported best model summary to: analysis/reports/figures/best_model_summary.csv
```

Effect Visualizations

```r
# Define custom theme
custom_theme <- theme_minimal(base_size = 12) +
    theme(
        panel.grid.major = element_line(color = "gray90", size = 0.5),
        panel.grid.minor = element_line(color = "gray95", size = 0.3),
        axis.text = element_text(color = "black", size = 11),
        axis.title = element_text(color = "black", size = 12, face = "bold"),
        plot.title = element_text(color = "black", size = 14, face = "bold", hjust = 0.5),
        panel.border = element_rect(color = "black", fill = NA, size = 0.5),
        plot.margin = margin(10, 10, 10, 10)
    )

# Function to add zero line
add_zero_line <- function(plot) {
    zero_line_layer <- geom_hline(yintercept = 0, color = "gray70", size = 0.8, alpha = 1)
    plot$layers <- c(list(zero_line_layer), plot$layers)
    return(plot)
}
```

```r
# Create effect plots for the best model
# Extract which terms are in the best model
best_formula <- aic_results$Formula[1]
has_smooth <- grepl("s\\(", best_formula)

if (has_smooth) {
    # For GAM with smooth terms
    plots <- list()

    # Check which smooth terms are in the model
    smooth_terms <- summary(best_model$gam)$s.table

    # Plot each smooth term
    for (i in 1:nrow(smooth_terms)) {
        term_name <- rownames(smooth_terms)[i]
        p <- draw(best_model$gam, select = term_name, rug = FALSE, residuals = FALSE) +
            custom_theme +
            theme(plot.caption = element_blank())
        p <- add_zero_line(p)
        plots[[i]] <- p
    }

    # Combine plots
    if (length(plots) > 0) {
        if (length(plots) <= 2) {
            combined_plots <- wrap_plots(plots, nrow = 1)
        } else if (length(plots) <= 4) {
            combined_plots <- wrap_plots(plots, nrow = 2)
        } else {
            combined_plots <- wrap_plots(plots, nrow = 3)
        }
        print(combined_plots)
    }
} else {
    # For linear models, create partial residual plots
    cat("Best model uses linear terms. Creating partial residual plots...\n")

    # Extract coefficients
    coef_summary <- summary(best_model$gam)$p.table
    print(coef_summary)
}
```

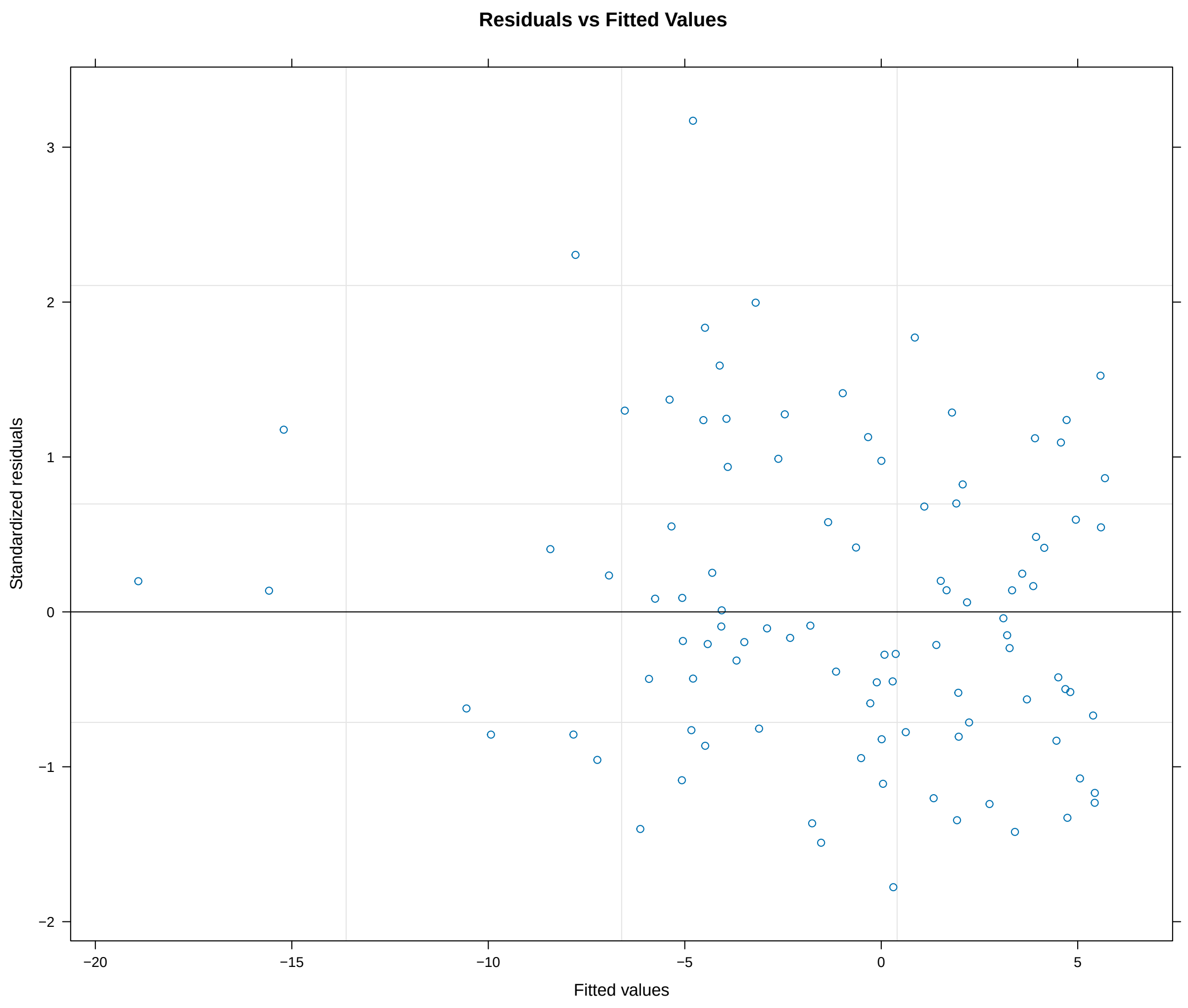


```r
# Export partial effects plot for best model (for results write-up)
if (has_smooth && length(plots) > 0) {
    ggsave(
        here("analysis", "reports", "figures", "best_model_partial_effects.png"),
        plot = combined_plots,
        width = 12, height = 8, dpi = 300
    )
    cat("Exported best model partial effects to: analysis/reports/figures/best_model_partial_
}
```

```
Exported best model partial effects to: analysis/reports/figures/best_model_partial_effects.
```

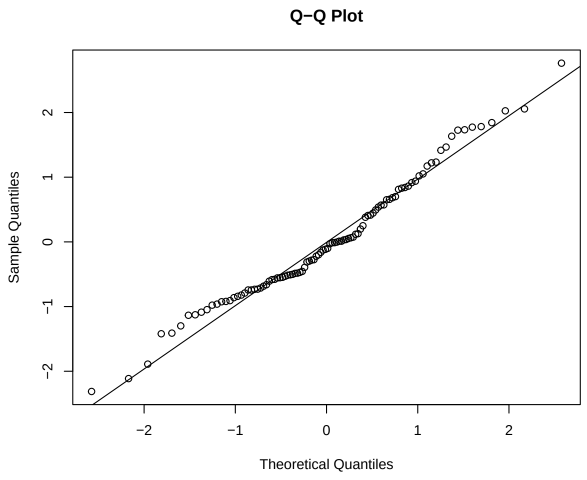
Model Diagnostics

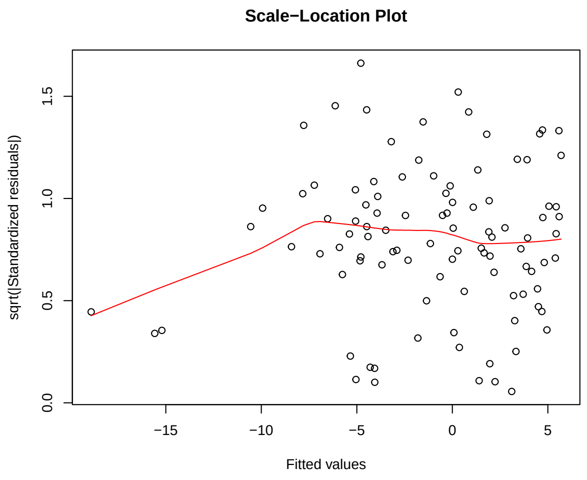
```r
# Create diagnostic plots
par(mfrow = c(2, 2))

# Residuals vs Fitted
plot(best_model$lme, main = "Residuals vs Fitted Values")
```

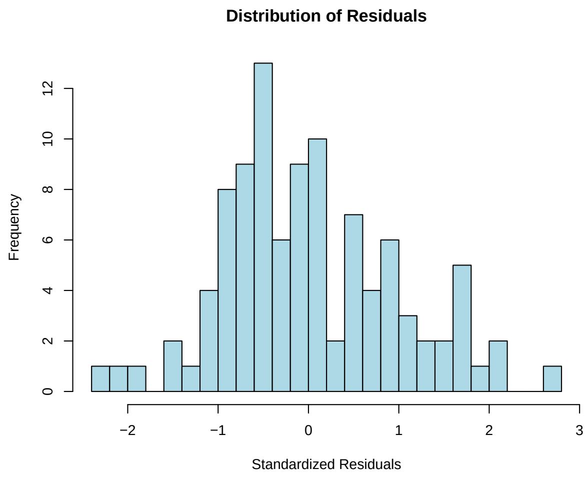


```r
# Q-Q plot
qqnorm(residuals(best_model$lme, type = "normalized"), main = "Q-Q Plot")
qqline(residuals(best_model$lme, type = "normalized"))

# Scale-location plot
```

```r
plot(fitted(best_model$lme), sqrt(abs(residuals(best_model$lme, type = "normalized"))),
    main = "Scale-Location Plot",
    xlab = "Fitted values",
    ylab = "sqrt(|Standardized residuals|)"
)
lines(lowess(fitted(best_model$lme), sqrt(abs(residuals(best_model$lme, type = "normalized")

# Histogram of residuals
hist(residuals(best_model$lme, type = "normalized"),
    breaks = 30,
    main = "Distribution of Residuals",
    xlab = "Standardized Residuals",
    col = "lightblue"
)

par(mfrow = c(1, 1))
```







```
# Export diagnostic plots for results write-up
png(here("analysis", "reports", "figures", "best_model_diagnostics.png"),
    width = 12, height = 10, units = "in", res = 300)
par(mfrow = c(2, 2))

# Residuals vs Fitted
plot(best_model$lme, main = "Residuals vs Fitted Values")

# Q-Q plot
qqnorm(residuals(best_model$lme, type = "normalized"), main = "Q-Q Plot")
qqline(residuals(best_model$lme, type = "normalized"))

# Scale-location plot
plot(fitted(best_model$lme), sqrt(abs(residuals(best_model$lme, type = "normalized"))),
    main = "Scale-Location Plot",
```

```r
    xlab = "Fitted values",
    ylab = "sqrt(|Standardized residuals|)"
)
lines(lowess(fitted(best_model$lme), sqrt(abs(residuals(best_model$lme, type = "normalized")

# Histogram of residuals
hist(residuals(best_model$lme, type = "normalized"),
    breaks = 30,
    main = "Distribution of Residuals",
    xlab = "Standardized Residuals",
    col = "lightblue"
)

par(mfrow = c(1, 1))
dev.off()
```

```
pdf
  2
```

```r
cat("Exported diagnostic plots to: analysis/reports/figures/best_model_diagnostics.png\n")
```

```
Exported diagnostic plots to: analysis/reports/figures/best_model_diagnostics.png
```

Models with Strong Support (ΔAICc $<$ 2)

This section examines all models with AIC differences less than 2 from the best model, as these represent models with substantial empirical support. For each supported model, we display the model summary and visualize partial effects.

```r
# Filter models with Delta AIC < 2
strong_support_models <- aic_results %>%
    filter(Delta_AIC < 2) %>%
    arrange(Delta_AIC)

cat("Number of models with ΔAIC < 2:", nrow(strong_support_models), "\n\n")
```

```
Number of models with <U+0394>AIC < 2: 7
```

```r
# Display the supported models
strong_support_models %>%
    select(Model, Correlation, Formula, AIC, Delta_AIC, AIC_weight, df) %>%
    kable(digits = 4, caption = "Models with Strong Empirical Support (ΔAIC < 2)")
```

Table 6: Models with Strong Empirical Support (<U+0394>AIC < 2)

Model	Correlation	Formula	AIC	Delta_AIC	AIC_weight	df
B33_**AR1**		butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_t_1)	668.4010000 0.14779			
B29c_AR1	AR1	butterfly_diff_95th_sqrt ~ s(butterflies_95th_percentile_t_1) + s(wind_max_gust_t_1)	668.6712700 0.12918			
B28_**AR1**		butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(sum_butterflies_direct_sun_t_1)	669.1016999 0.10417			
B35_**AR1**		butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_max_t_1) + s(temp_min_t_1) + s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_t_1)	669.5713071 9 0.082213			
B37_**AR1**		butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_max_t_1) + s(temp_min_t_1) + s(temp_at_max_count_t_1) + s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_t_1)	669.5941930 0.081415			
B29_**AR1**		butterfly_diff_95th_sqrt ~ s(butterflies_95th_percentile_t_1)	669.6815284 2 0.07776			
B34_**AR1**		butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_at_max_count_t_1) + s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_t_1)	670.0163152 0.065911			

```r
# Export strong support models
write_csv(
    strong_support_models,
    here("analysis", "reports", "figures", "strong_support_models.csv")
)
cat("Exported strong support models table\n")
```

```
Exported strong support models table
```

Model Summaries for Supported Models

```r
# Display summary for each supported model
for (i in 1:nrow(strong_support_models)) {
    model_name <- strong_support_models$Model[i]
    model_obj <- successful_models[[model_name]]

    cat("\n")
    cat("==================================================\n")
    cat("MODEL:", model_name, "\n")
    cat("Formula:", strong_support_models$Formula[i], "\n")
    cat("ΔAIC:", round(strong_support_models$Delta_AIC[i], 3), "\n")
    cat("AIC Weight:", round(strong_support_models$AIC_weight[i], 4), "\n")
    cat("==================================================\n")

    # Model summary
    print(summary(model_obj$gam))

    # Calculate performance metrics
    r_squared <- summary(model_obj$gam)$r.sq
    dev_explained <- summary(model_obj$gam)$dev.expl

    cat("\nModel Performance:\n")
    cat("R-squared:", round(r_squared, 4), "\n")
    cat("Deviance explained:", round(dev_explained * 100, 2), "%\n")
    cat("\n")
}
```

```
==================================================
MODEL: B33_AR1
Formula: butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(wind_max_gust_t_1) +
<U+0394>AIC: 0
AIC Weight: 0.1477
==================================================

Family: gaussian
Link function: identity
```

```
Formula:
butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 +
    s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_t_1)

Parametric coefficients:
                                 Estimate Std. Error t value Pr(>|t|)
(Intercept)                      3.444416   1.263453   2.726  0.00766 **
butterflies_95th_percentile_t_1 -0.037703   0.006972  -5.408 4.95e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                                    edf Ref.df     F p-value
s(wind_max_gust_t_1)              2.466  2.466 2.725 0.08649 .
s(sum_butterflies_direct_sun_t_1) 2.918  2.918 6.122 0.00245 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.226
  Scale est. = 43.072    n = 100

Model Performance:
R-squared: 0.2264
Deviance explained:  %


==================================================
MODEL: B29c_AR1
Formula: butterfly_diff_95th_sqrt ~ s(butterflies_95th_percentile_t_1) + s(wind_max_gust_t_1
<U+0394>AIC: 0.27
AIC Weight: 0.1291
==================================================

Family: gaussian
Link function: identity

Formula:
butterfly_diff_95th_sqrt ~ s(butterflies_95th_percentile_t_1) +
    s(wind_max_gust_t_1)

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   -1.078      1.068  -1.009    0.315
```

```
Approximate significance of smooth terms:
                                     edf Ref.df      F  p-value
s(butterflies_95th_percentile_t_1) 1.153  1.153 24.194 1.55e-06 ***
s(wind_max_gust_t_1)               2.491  2.491  2.877   0.0649 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.175
  Scale est. = 44.916    n = 100

Model Performance:
R-squared: 0.1753
Deviance explained:  %


==================================================
MODEL: B28_AR1
Formula: butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(sum_butterflies_dire
<U+0394>AIC: 0.7
AIC Weight: 0.1041
==================================================

Family: gaussian
Link function: identity

Formula:
butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 +
    s(sum_butterflies_direct_sun_t_1)

Parametric coefficients:
                                 Estimate Std. Error t value Pr(>|t|)
(Intercept)                      3.560773   1.315741   2.706  0.00807 **
butterflies_95th_percentile_t_1 -0.038876   0.007134  -5.449 3.97e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                                    edf Ref.df     F p-value
s(sum_butterflies_direct_sun_t_1) 2.886  2.886 6.284 0.00297 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
R-sq.(adj) =  0.168
  Scale est. = 46.265    n = 100

Model Performance:
R-squared: 0.1679
Deviance explained:  %


==================================================
MODEL: B35_AR1
Formula: butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_max_t_1) + s(ten
<U+0394>AIC: 1.172
AIC Weight: 0.0822
==================================================

Family: gaussian
Link function: identity

Formula:
butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 +
    s(temp_max_t_1) + s(temp_min_t_1) + s(wind_max_gust_t_1) +
    s(sum_butterflies_direct_sun_t_1)

Parametric coefficients:
                                 Estimate Std. Error t value Pr(>|t|)
(Intercept)                      3.711029   1.247939   2.974  0.00377 **
butterflies_95th_percentile_t_1 -0.039470   0.006988  -5.648 1.84e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                                    edf Ref.df     F p-value
s(temp_max_t_1)                   1.000  1.000 2.272 0.13522
s(temp_min_t_1)                   1.000  1.000 0.842 0.36135
s(wind_max_gust_t_1)              2.362  2.362 2.184 0.16218
s(sum_butterflies_direct_sun_t_1) 2.856  2.856 6.450 0.00261 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.254
  Scale est. = 41.83     n = 100

Model Performance:
```

```
R-squared: 0.2541
Deviance explained:  %


==================================================
MODEL: B37_AR1
Formula: butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_max_t_1) + s(te
<U+0394>AIC: 1.193
AIC Weight: 0.0814
==================================================

Family: gaussian
Link function: identity

Formula:
butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 +
    s(temp_max_t_1) + s(temp_min_t_1) + s(temp_at_max_count_t_1) +
    s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_t_1)

Parametric coefficients:
                                 Estimate Std. Error t value Pr(>|t|)
(Intercept)                      3.436246   1.240927   2.769  0.00684 **
butterflies_95th_percentile_t_1 -0.037152   0.006818  -5.449 4.47e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                                   edf Ref.df     F p-value
s(temp_max_t_1)                  1.000  1.000 1.714  0.1939
s(temp_min_t_1)                  1.000  1.000 2.726  0.1023
s(temp_at_max_count_t_1)         1.713  1.713 1.396  0.1648
s(wind_max_gust_t_1)             2.508  2.508 1.695  0.1173
s(sum_butterflies_direct_sun_t_1) 2.876  2.876 5.087  0.0067 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.301
  Scale est. = 38.569    n = 100

Model Performance:
R-squared: 0.3011
Deviance explained:  %
```

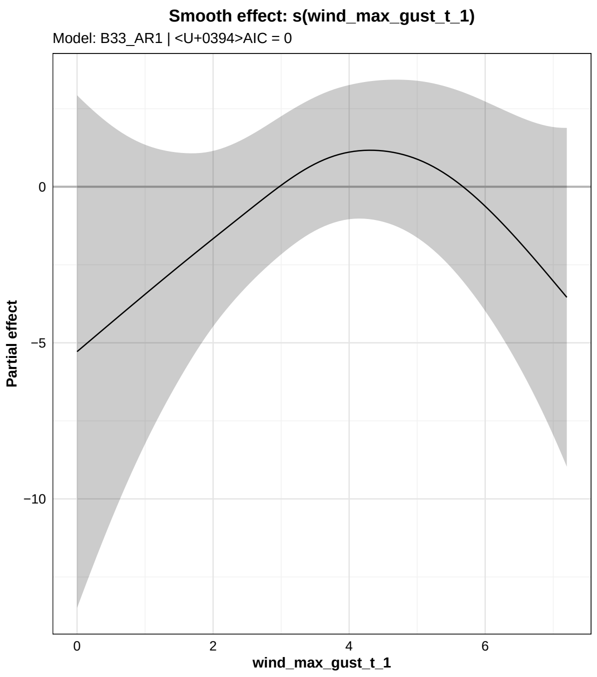
```
==================================================
MODEL: B29_AR1
Formula: butterfly_diff_95th_sqrt ~ s(butterflies_95th_percentile_t_1)
<U+0394>AIC: 1.284
AIC Weight: 0.0777
==================================================

Family: gaussian
Link function: identity

Formula:
butterfly_diff_95th_sqrt ~ s(butterflies_95th_percentile_t_1)

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -1.0431     0.9863  -1.058    0.293

Approximate significance of smooth terms:
                                   edf Ref.df     F  p-value
s(butterflies_95th_percentile_t_1)   1      1 29.03 6.26e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.103
  Scale est. = 50.178    n = 100

Model Performance:
R-squared: 0.1026
Deviance explained:  %


==================================================
MODEL: B34_AR1
Formula: butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_at_max_count_t_
<U+0394>AIC: 1.615
AIC Weight: 0.0659
==================================================

Family: gaussian
Link function: identity

Formula:
```

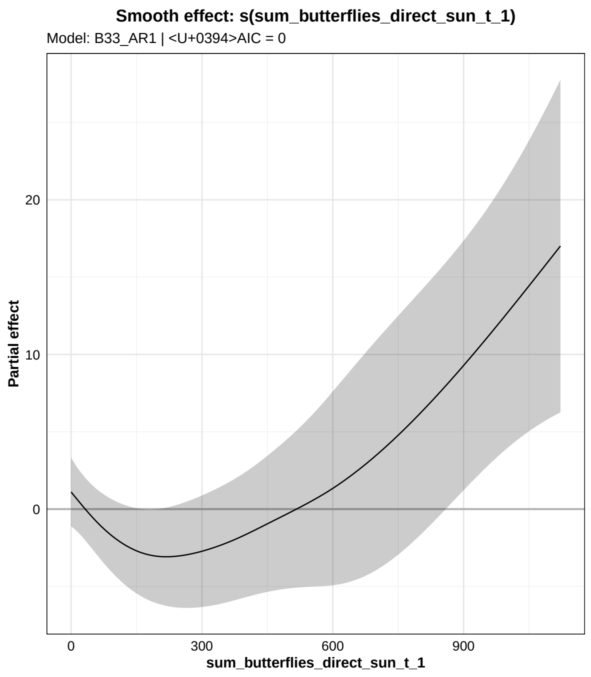
```
butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 +
    s(temp_at_max_count_t_1) + s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_t_1)

Parametric coefficients:
                                 Estimate Std. Error t value Pr(>|t|)
(Intercept)                      3.322212   1.285420   2.585   0.0113 *
butterflies_95th_percentile_t_1 -0.036858   0.007003  -5.264 9.26e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                                   edf Ref.df     F p-value
s(temp_at_max_count_t_1)         1.000  1.000 1.600 0.20906
s(wind_max_gust_t_1)             2.585  2.585 3.234 0.04986 *
s(sum_butterflies_direct_sun_t_1) 2.845  2.845 5.172 0.00734 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.227
  Scale est. = 42.79     n = 100

Model Performance:
R-squared: 0.2268
Deviance explained:  %
```

Partial Effects for Supported Models

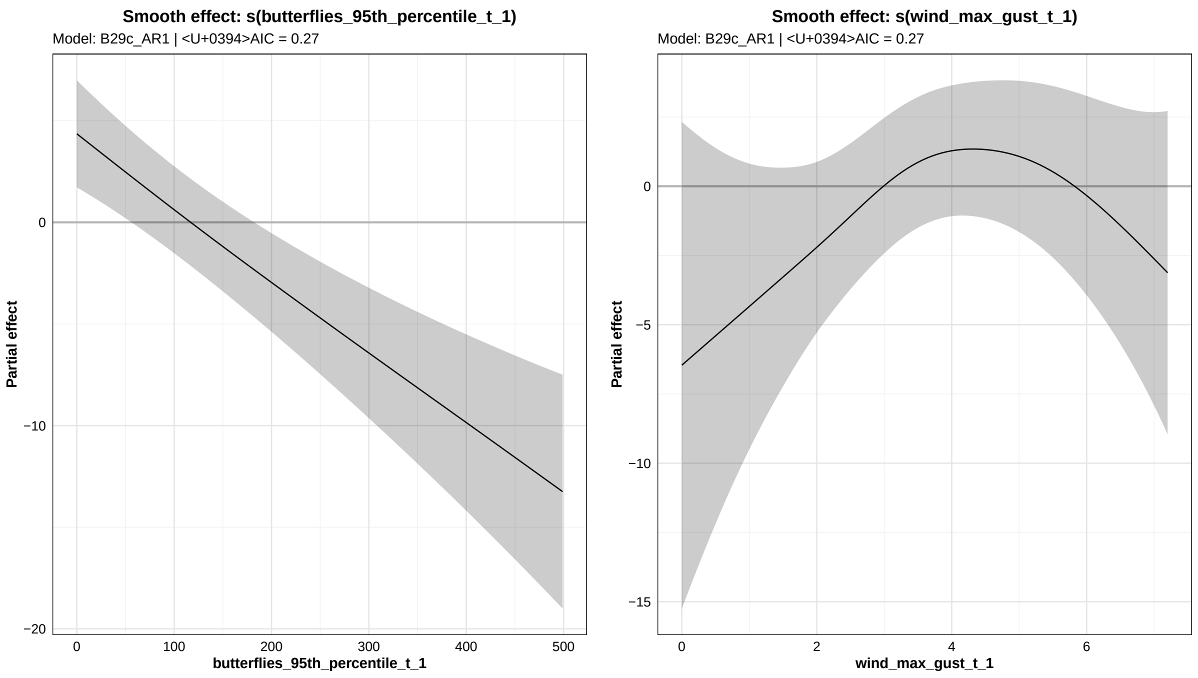
```r
# Create effect plots for each supported model
for (i in 1:nrow(strong_support_models)) {
    model_name <- strong_support_models$Model[i]
    model_obj <- successful_models[[model_name]]
    formula_str <- strong_support_models$Formula[i]

    cat("\n")
    cat("PARTIAL EFFECTS FOR MODEL:", model_name, "\n")
    cat("Formula:", formula_str, "\n")
    cat("ΔAIC:", round(strong_support_models$Delta_AIC[i], 3), "\n\n")

    # Check if model has smooth terms
    has_smooth <- grepl("s\\(", formula_str)
    if (has_smooth) {
        # For GAM with smooth terms
        tryCatch(
            {
                smooth_terms <- summary(model_obj$gam)$s.table

                if (nrow(smooth_terms) > 0) {
                    plots <- list()

                    # Plot each smooth term
                    for (j in 1:nrow(smooth_terms)) {
                        term_name <- rownames(smooth_terms)[j]
                        p <- draw(model_obj$gam, select = term_name, rug = FALSE, residuals =
                            custom_theme +
                            theme(plot.caption = element_blank()) +
                            labs(
                                title = paste("Smooth effect:", term_name),
                                subtitle = paste("Model:", model_name, "| ΔAIC =", round(str
                            )
                        p <- add_zero_line(p)
                        plots[[j]] <- p
                    }

                    # Combine plots
                    if (length(plots) > 0) {
                        if (length(plots) <= 2) {
                            combined_plots <- wrap_plots(plots, nrow = 1)
                        } else if (length(plots) <= 4) {
                            combined_plots <- wrap_plots(plots, nrow = 2)
                        } else {
                            combined_plots <- wrap_plots(plots, nrow = 3)
                        }
                        print(combined_plots)
                    }
                } else {
                    cat("No smooth terms found in this model.\n")
                }
            },
            error = function(e) {
                cat("Error creating smooth term plots:", e$message, "\n")
            }
        )
    }

    # Always show coefficient table if available
    tryCatch(
        {
            if (nrow(summary(model_obj$gam)$p.table) > 0) {
                cat("\nParametric coefficients:\n")
                coef_table <- summary(model_obj$gam)$p.table
                print(kable(coef_table, digits = 4, caption = paste("Parametric coefficients
            }
        },
        error = function(e) {
            cat("Error displaying coefficients:", e$message, "\n")
        }
    )

    cat("\n", paste(rep("=", 80), collapse = ""), "\n")
}
```

```
PARTIAL EFFECTS FOR MODEL: B33_AR1
Formula: butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(wind_max_gust_t_1) +
<U+0394>AIC: 0
```





```
Parametric coefficients:


Table: Parametric coefficients for B33_AR1

|                                | Estimate| Std. Error| t value| Pr(>&#124;t&#124;)|
|:-------------------------------|--------:|----------:|-------:|------------------:|
|(Intercept)                     |   3.4444|     1.2635|  2.7262|             0.0077|
|butterflies_95th_percentile_t_1 |  -0.0377|     0.0070| -5.4078|             0.0000|

 ================================================================================

PARTIAL EFFECTS FOR MODEL: B29c_AR1
Formula: butterfly_diff_95th_sqrt ~ s(butterflies_95th_percentile_t_1) + s(wind_max_gust_t_1
<U+0394>AIC: 0.27
```

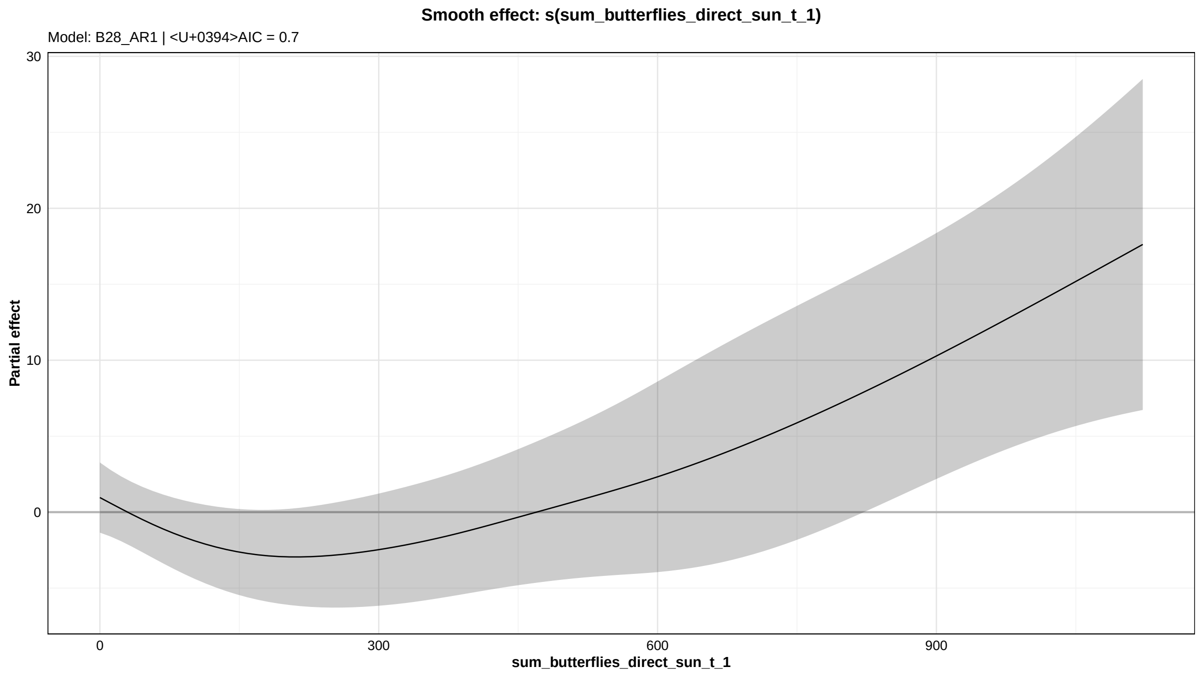


```
Parametric coefficients:


Table: Parametric coefficients for B29c_AR1
```

	Estimate	Std. Error	t value	Pr(>&#124;t&#124;)
(Intercept)	-1.078	1.0681	-1.0093	0.3154

 ==

PARTIAL EFFECTS FOR MODEL: B28_AR1
Formula: butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(sum_butterflies_dire
<U+0394>AIC: 0.7



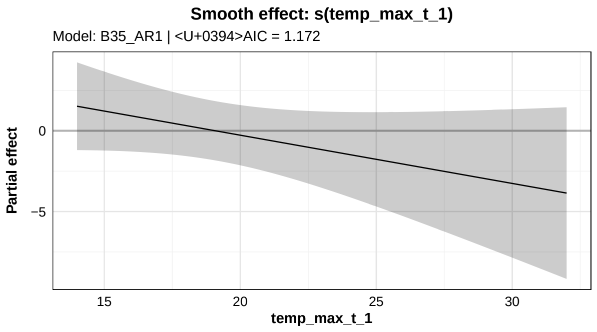
Parametric coefficients:

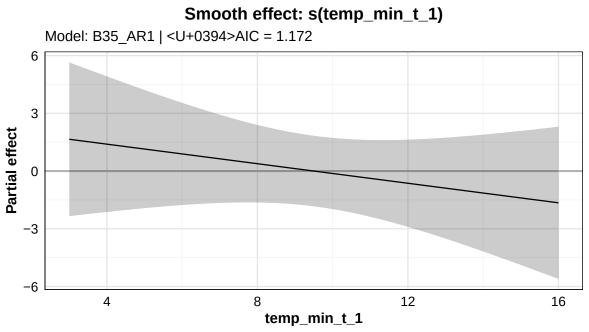
Table: Parametric coefficients for B28_AR1

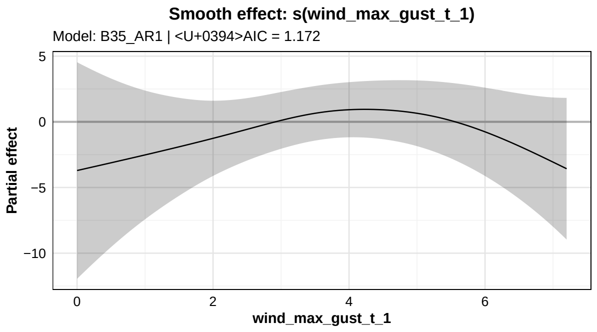
	Estimate	Std. Error	t value	Pr(>&#124;t&#124;)
(Intercept)	3.5608	1.3157	2.7063	0.0081
butterflies_95th_percentile_t_1	-0.0389	0.0071	-5.4491	0.0000

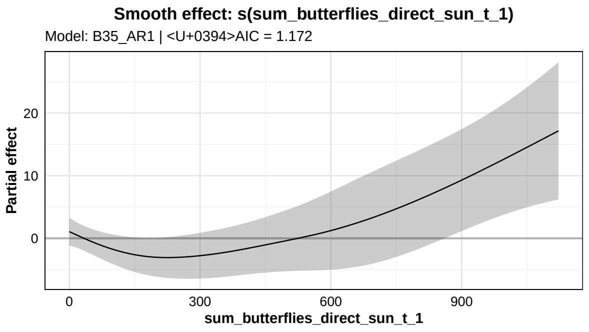
 ==

```
PARTIAL EFFECTS FOR MODEL: B35_AR1
Formula: butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_max_t_1) + s(ter
<U+0394>AIC: 1.172
```

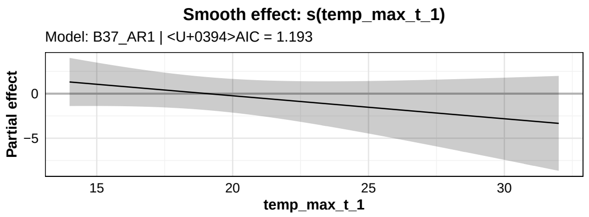


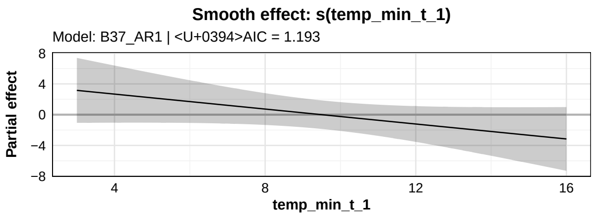


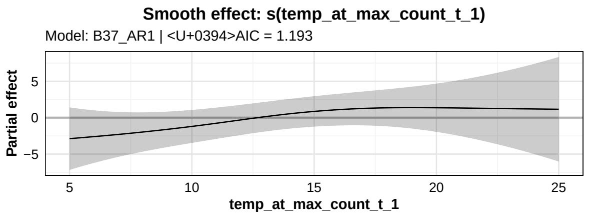


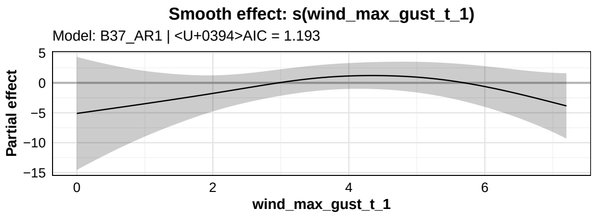


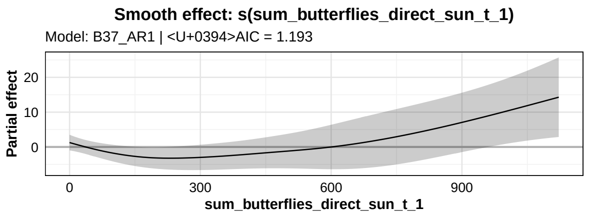
```
Parametric coefficients:


Table: Parametric coefficients for B35_AR1

|                                | Estimate| Std. Error| t value| Pr(>&#124;t&#124;)|
|:-------------------------------|--------:|----------:|-------:|------------------:|
|(Intercept)                     |   3.7110|     1.2479|  2.9737|             0.0038|
|butterflies_95th_percentile_t_1 |  -0.0395|     0.0070| -5.6480|             0.0000|

 ================================================================================

PARTIAL EFFECTS FOR MODEL: B37_AR1
Formula: butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_max_t_1) + s(ter
<U+0394>AIC: 1.193
```



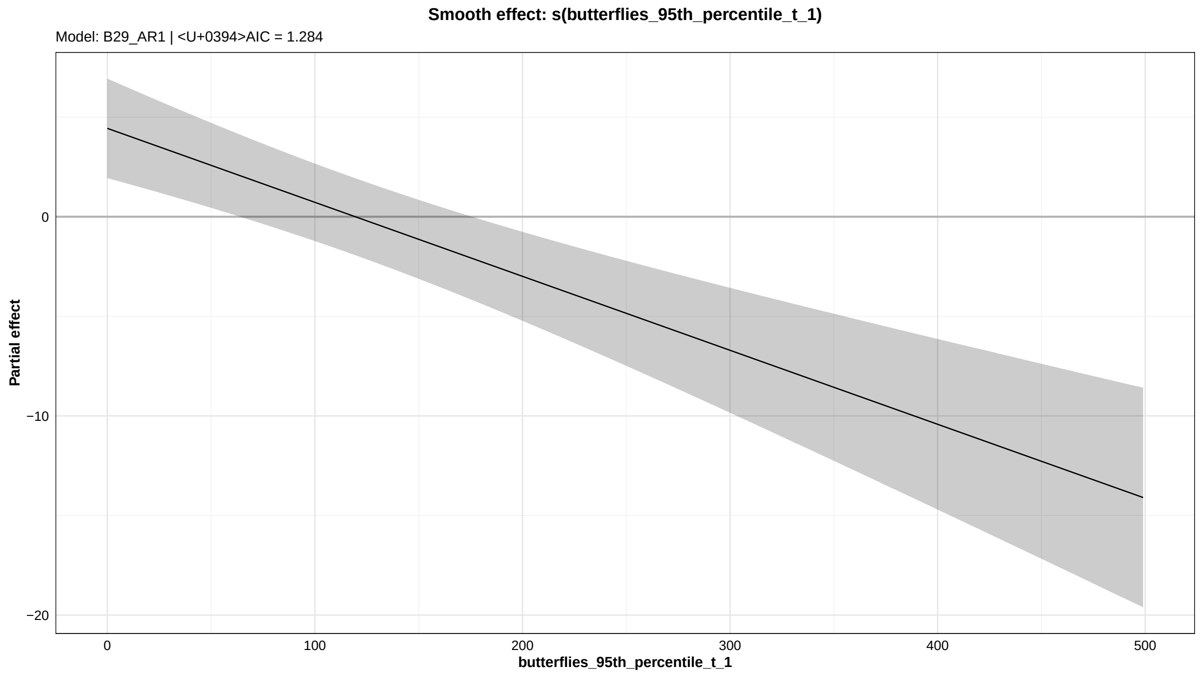




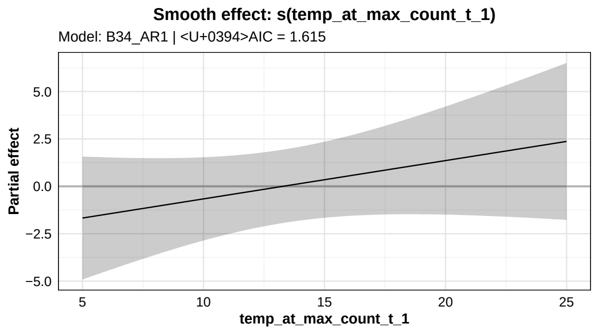


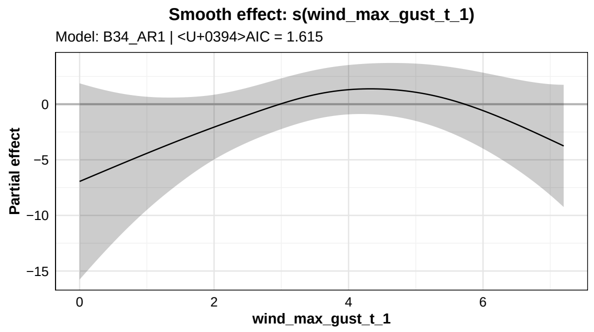


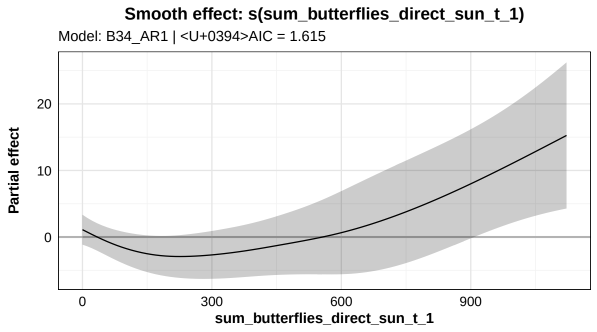
```
Parametric coefficients:


Table: Parametric coefficients for B37_AR1

|                               | Estimate| Std. Error| t value| Pr(>&#124;t&#124;)|
|:------------------------------|--------:|----------:|-------:|------------------:|
|(Intercept)                    |   3.4362|     1.2409|  2.7691|             0.0068|
|butterflies_95th_percentile_t_1 |  -0.0372|     0.0068| -5.4491|             0.0000|

 ===========================================================================

PARTIAL EFFECTS FOR MODEL: B29_AR1
Formula: butterfly_diff_95th_sqrt ~ s(butterflies_95th_percentile_t_1)
<U+0394>AIC: 1.284
```



```
Parametric coefficients:


Table: Parametric coefficients for B29_AR1

|            | Estimate| Std. Error| t value| Pr(>&#124;t&#124;)|
|:-----------|--------:|----------:|-------:|------------------:|
|(Intercept) |  -1.0431|     0.9863| -1.0577|             0.2928|

 ================================================================================

PARTIAL EFFECTS FOR MODEL: B34_AR1
Formula: butterfly_diff_95th_sqrt ~ butterflies_95th_percentile_t_1 + s(temp_at_max_count_t_
<U+0394>AIC: 1.615
```







```
Parametric coefficients:


Table: Parametric coefficients for B34_AR1

|                                 | Estimate| Std. Error| t value| Pr(>&#124;t&#124;)|
|:--------------------------------|--------:|----------:|-------:|------------------:|
|(Intercept)                      |   3.3222|     1.2854|  2.5845|             0.0113|
|butterflies_95th_percentile_t_1  |  -0.0369|     0.0070| -5.2635|             0.0000|

 ================================================================================
```

```r
# Export partial effects for all supported models (for results write-up)
for (i in 1:nrow(strong_support_models)) {
    model_name <- strong_support_models$Model[i]
    model_obj <- successful_models[[model_name]]
    formula_str <- strong_support_models$Formula[i]

    has_smooth <- grepl("s\\(", formula_str)

    if (has_smooth) {
        tryCatch({
            smooth_terms <- summary(model_obj$gam)$s.table

            if (nrow(smooth_terms) > 0) {
                plots <- list()

                for (j in 1:nrow(smooth_terms)) {
                    term_name <- rownames(smooth_terms)[j]
                    p <- draw(model_obj$gam, select = term_name, rug = FALSE, residuals = FAL
                        custom_theme +
                        theme(plot.caption = element_blank()) +
                        labs(
                            title = paste("Smooth effect:", term_name),
                            subtitle = paste("Model:", model_name, "| ΔAIC =", round(strong_s
                        )
                    p <- add_zero_line(p)
                    plots[[j]] <- p
                }

                if (length(plots) > 0) {
                    if (length(plots) <= 2) {
                        combined_plots <- wrap_plots(plots, nrow = 1)
                    } else if (length(plots) <= 4) {
                        combined_plots <- wrap_plots(plots, nrow = 2)
                    } else {
                        combined_plots <- wrap_plots(plots, nrow = 3)
                    }

                    # Export this model's plots
                    ggsave(
                        here("analysis", "reports", "figures",
                             paste0("model_", model_name, "_partial_effects.png")),
                        plot = combined_plots,
                        width = 14, height = 8, dpi = 300
                    )
                }
            }
        }, error = function(e) {
            cat("Could not export plots for model", model_name, "\n")
        })
    }
}
cat("Exported partial effects for all supported models to: analysis/reports/figures/\n")
```

```
Exported partial effects for all supported models to: analysis/reports/figures/
```