Daily-Level GAM Analysis of Monarch Butterfly Abundance

Kyle Nessen

2025-09-29

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 45 columns
```

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

```
Number of deployments: 7
```

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

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

```r
# 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)
```

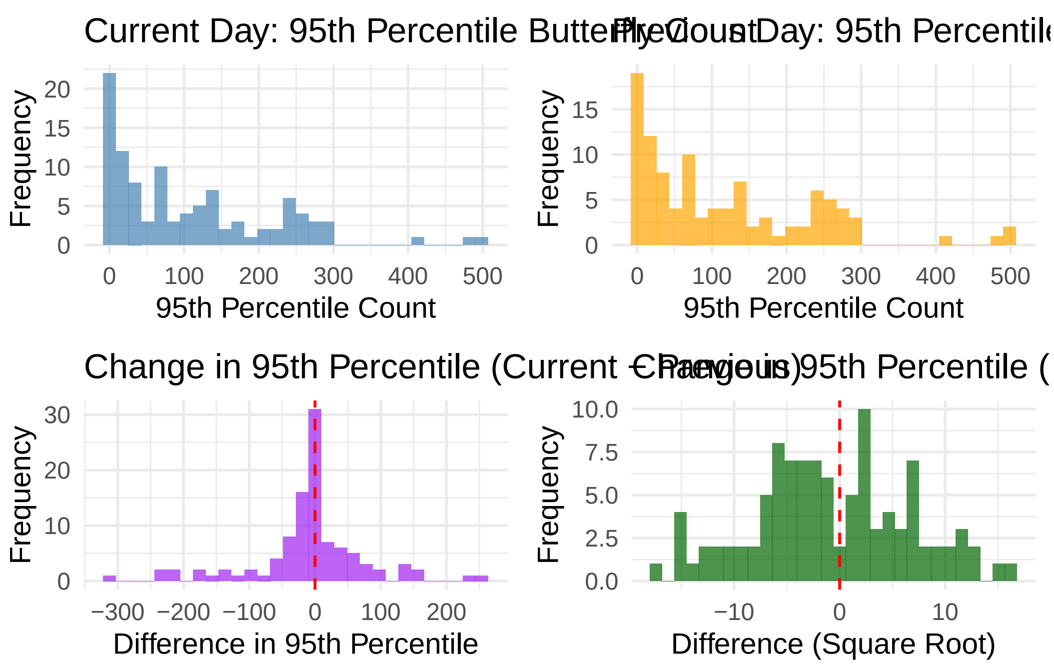
```
 butterflies_95th_percentile_t butterflies_95th_percentile_t_1
 Min.   :  0.00                Min.   :  0.0
 1st Qu.: 14.85                1st Qu.: 17.5
 Median : 70.05                Median : 77.0
 Mean   :107.41                Mean   :116.3
 3rd Qu.:166.95                3rd Qu.:199.5
 Max.   :499.00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

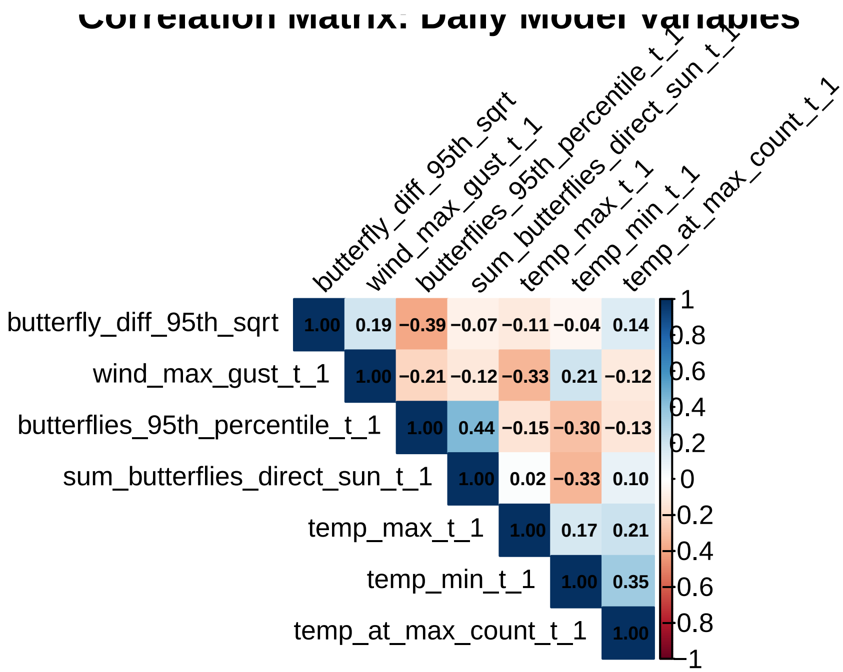
```
        title = "Change in 95th Percentile (Current - Previous)",
        x = "Difference in 95th Percentile", y = "Frequency"
    ) +
    theme_minimal()

# Square root transformed difference
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

```
# Select model variables
model_vars <- daily_data %>%
```

```r
    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"
)
```




```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
response_candidates <- daily_data %>%
    select(contains("diff"), contains("butterfly")) %>%
    select(-contains("direct_sun")) %>%  # Remove non-response variables
    names()

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

```
Available response variable candidates:
```

```r
print(response_candidates)
```

```
 [1] "butterfly_diff"           "butterfly_diff_cbrt"
 [3] "butterfly_diff_log"       "butterfly_diff_95th"
 [5] "butterfly_diff_95th_cbrt" "butterfly_diff_95th_log"
 [7] "butterfly_diff_top3"      "butterfly_diff_top3_cbrt"
 [9] "butterfly_diff_top3_log"  "butterfly_diff_95th_sqrt"
```

```r
# Define transformations to test
transformations <- list(
    "original" = function(x) x,
    "sqrt" = function(x) ifelse(x >= 0, sqrt(x), -sqrt(-x)),  # Signed square root
    "fourth_root" = function(x) ifelse(x >= 0, x^0.25, -((-x)^0.25)),  # Signed fourth root
    "arcsinh" = function(x) asinh(x),  # Inverse hyperbolic sine (handles negative values)
    "yeo_johnson" = function(x) {
        # Simplified Yeo-Johnson transformation
        lambda <- 0.5
        ifelse(x >= 0,
               ((x + 1)^lambda - 1) / lambda,
               -(((-x) + 1)^(2-lambda) - 1) / (2-lambda))
    }
)

# 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
        }
    }
}

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

# Rank by normality score
normality_ranking <- normality_df %>%
    arrange(desc(Normality_Score)) %>%
```

```r
    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_95th_sqrt_original	2	butterfly_diff_95th_sqrt	original	103	-0.809	7.382	0.021	-0.467	0.6501	0.5918	0.8102
butterfly_diff_top3_sqrt	3	butterfly_diff_top3	sqrt	103	-0.751	7.379	0.039	-0.436	0.6273	0.5818	0.8033
butterfly_diff_sqrt	4	butterfly_diff	sqrt	103	-1.148	8.033	0.238	-0.117	0.6179	0.3799	0.7552
butterfly_diff_top3_cbrt_original	5	butterfly_diff_top3_cbrt	original	103	-0.345	3.505	0.097	-1.223	0.0004	0.0000	0.4938
butterfly_diff_95th_cbrt_original	6	butterfly_diff_95th_cbrt	original	103	-0.393	3.502	0.129	-1.212	0.0004	0.0000	0.4898
butterfly_diff_cbrt_original	7	butterfly_diff_cbrt	original	103	-0.586	3.678	0.279	-1.063	0.0005	0.0000	0.4786
butterfly_diff_cbrt_yeo_johnson	8	butterfly_diff_cbrt	yeo_johnson	103	-2.465	4.671	-0.602	-0.503	0.0000	0.0000	0.4720
butterfly_diff_top3_log_original	9	butterfly_diff_top3_log	original	103	-0.340	3.506	0.123	-1.472	0.0000	0.0000	0.4711
butterfly_diff_top3_fourth_root	10	butterfly_diff_top3	fourth_root	103	-0.236	2.475	0.121	-1.527	0.0000	0.0000	0.4672
butterfly_diff_95th_log_original	11	butterfly_diff_95th_log	original	103	-0.394	3.505	0.171	-1.455	0.0000	0.0000	0.4653
butterfly_diff_top3_arcsinh	12	butterfly_diff_top3	arcsinh	103	-0.392	4.105	0.129	-1.560	0.0000	0.0000	0.4636

	Rank	Variable	Transformation	N	Mean	SD	Skewness	Kurtosis	Shapiro_p	Anderson_p	Normality_Score
butterfly_diff_log_yeo_johnson	13	butterfly_diff_log	yeo_johnson	103	-2.458	4.450	-0.275	-1.289	0.0000	0.0000	0.4621
butterfly_diff_95th_fourth_root	14	butterfly_diff_95th	fourth_root	103	-0.279	2.470	0.168	-1.505	0.0000	0.0000	0.4619
butterfly_diff_95th_sqrt_sqrt	15	butterfly_diff_95th_sqrt	sqrt	103	-0.279	2.470	0.168	-1.505	0.0000	0.0000	0.4619

```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_95th_sqrt	original	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
butterfly_diff_top3_cbrt	original	0.4938	0.097	-1.223	0.0004
butterfly_diff_95th_cbrt	original	0.4898	0.129	-1.212	0.0004
butterfly_diff_cbrt	original	0.4786	0.279	-1.063	0.0005
butterfly_diff_top3_log	original	0.4711	0.123	-1.472	0.0000
butterfly_diff_95th_log	original	0.4653	0.171	-1.455	0.0000
butterfly_diff_log	yeo_johnson	0.4621	-0.275	-1.289	0.0000

```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")
```

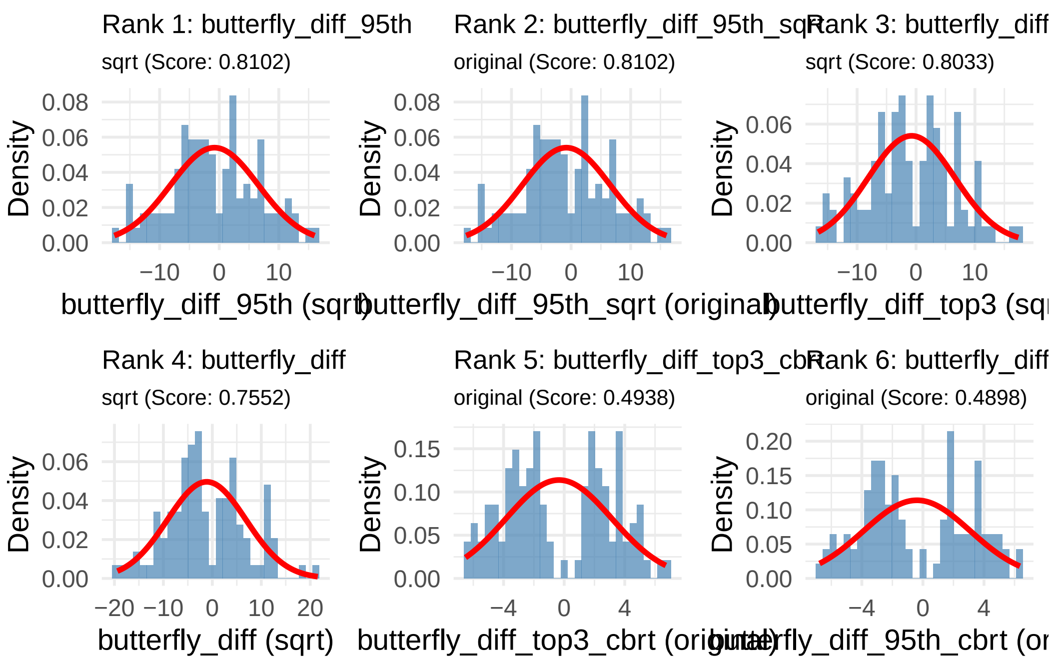
```
Summary of transformed response variable:
```

```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 the top 6 most normal transformations
top_transformations <- head(normality_ranking, 6)

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),
```

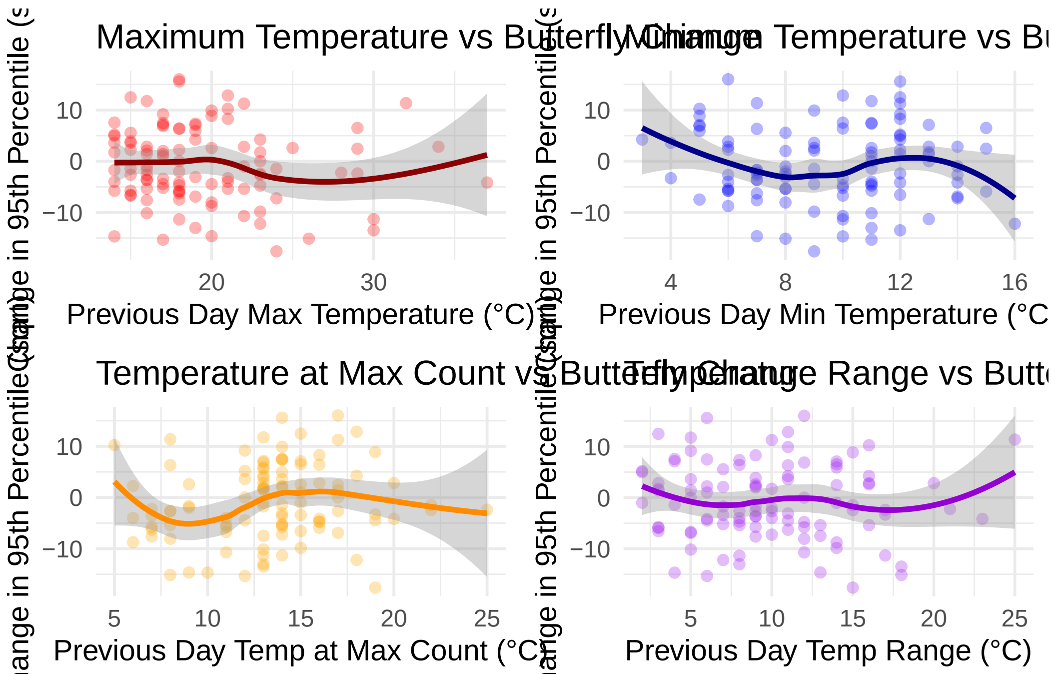
```r
                subtitle = paste0(trans_name, " (Score: ", row$Normality_Score, ")"),
                x = paste0(var_name, " (", trans_name, ")"),
                y = "Density"
            ) +
            theme_minimal() +
            theme(plot.title = element_text(size = 10),
                  plot.subtitle = element_text(size = 8))

        plots[[i]] <- p
    }
}

# Arrange plots in grid
if(length(plots) >= 6) {
    grid.arrange(plots[[1]], plots[[2]], plots[[3]],
                plots[[4]], plots[[5]], plots[[6]], ncol = 3)
} else {
    do.call(grid.arrange, c(plots, ncol = 3))
}
```



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",
```

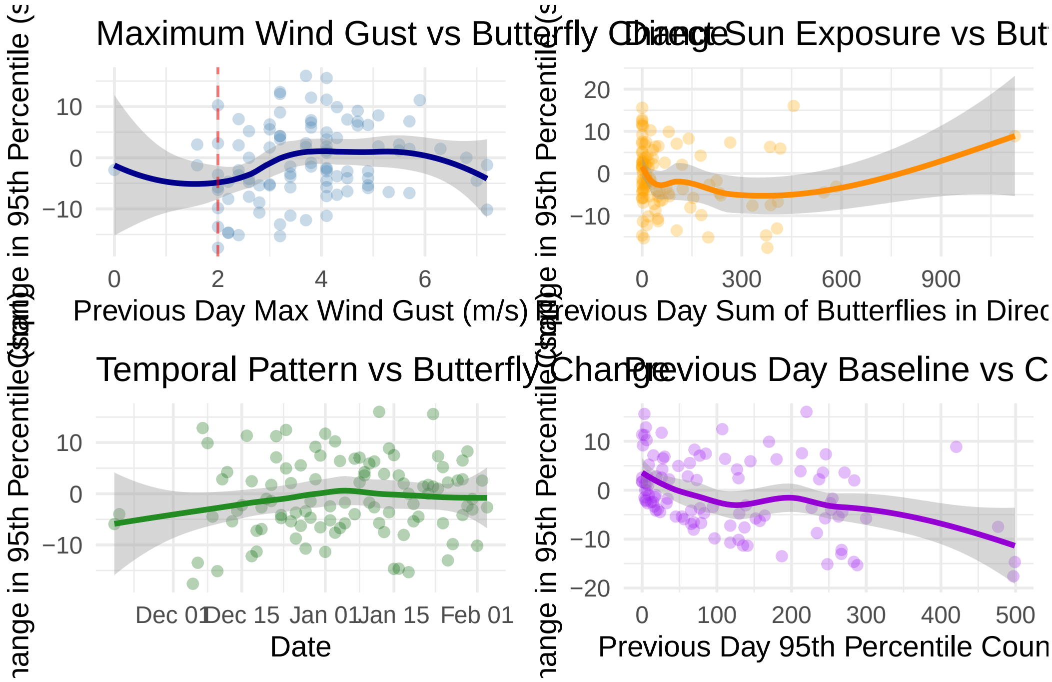
```r
        x = "Previous Day Temp Range (°C)",
        y = "Change in 95th Percentile (sqrt)"
    ) +
    theme_minimal()

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



Wind and Sun Exposure

```r
# 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_sqrt
    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],
        # Create a day sequence for temporal autocorrelation
        day_sequence = as.numeric(date_t - min(date_t)) + 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 the **absolute 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. **Fixed Effects** (WITHOUT controlling for previous day's abundance):
   - Temperature variables: max, min, and temperature at max count (testing various combinations)
   - Wind: maximum gust from previous day
   - Sun exposure: sum of butterflies in direct sun from previous day
3. **Random Effects**:
   - Deployment ID (random intercept only)
   - No temporal autocorrelation structure (simplified model)

Note: This analysis deliberately excludes the previous day's butterfly count (`butterflies_95th_percentile_t_` to test whether environmental variables have direct effects on absolute changes in abundance, rather than proportional effects after controlling for baseline levels.

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
# Using only no correlation structure (removed AR1 and compound symmetry)
correlation_structures <- list(
    "no_corr" = NULL  # No temporal correlation
)

# 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
```

```r
    "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_t

    # 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
)

cat("Total models to fit (WITHOUT previous day baseline):", length(model_specs), "\n")
```

```
Total models to fit (WITHOUT previous day baseline): 45
```

Model Fitting

```r
# 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 (no correlation structures)
cat("Fitting models...\n")
```

```
Fitting models...
```

```r
fitted_models <- list()

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

    # Fit the model with no correlation structure
    fitted_models[[model_name]] <- fit_model_safely(
        formula_str, model_data, NULL, "no_corr"
    )
}

# 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 45 out of 45 models
```

Model Comparison

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

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

    data.frame(
        Model = model_name,
        Formula = formula_str,
        AIC = AIC(model$lme),
        LogLik = logLik(model$lme)[1],
        df = attr(logLik(model$lme), "df"),
        stringsAsFactors = FALSE
```

```r
    )
}) %>%
    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
aic_results %>%
    head(10) %>%
    select(Model, AIC, Delta_AIC, AIC_weight, df) %>%
    kable(digits = 3, caption = "Top 10 models by AIC")
```

Table 4: Top 10 models by AIC

Model	AIC	Delta_AIC	AIC_weight	df
M34	684.934	0.000	0.163	9
M31	685.514	0.580	0.122	7
M37	685.514	0.580	0.122	13
M38	685.608	0.674	0.116	8
M33	685.744	0.810	0.108	7
M24	686.896	1.963	0.061	5
M2	687.469	2.535	0.046	4
M40	687.661	2.727	0.042	8
M14	687.998	3.065	0.035	5
M28	688.520	3.587	0.027	5

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

```
Top 5 model specifications:
```

```r
head(aic_results, 5) %>%
    select(Model, Formula, Delta_AIC) %>%
    kable(digits = 3)
```

Model	Formula	Delta_AIC
M34	butterfly_diff_95th_sqrt ~ s(temp_at_max_count_t_1) + s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_t_1)	0.000
M31	butterfly_diff_95th_sqrt ~ s(temp_at_max_count_t_1) + s(wind_max_gust_t_1)	0.580
M37	butterfly_diff_95th_sqrt ~ 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)	0.580
M38	butterfly_diff_95th_sqrt ~ temp_at_max_count_t_1 + s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_t_1)	0.674
M33	butterfly_diff_95th_sqrt ~ s(wind_max_gust_t_1) + s(sum_butterflies_direct_sun_t_1)	0.810

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: M34
```

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

```
Formula: butterfly_diff_95th_sqrt ~ s(temp_at_max_count_t_1) + s(wind_max_gust_t_1) + s(sum_l
```

```
# Model summary
summary(best_model$gam)
```

```
Family: gaussian
Link function: identity

Formula:
butterfly_diff_95th_sqrt ~ 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)  -0.8224     0.7388  -1.113    0.269

Approximate significance of smooth terms:
                                  edf Ref.df     F p-value
s(temp_at_max_count_t_1)          1.000  1.000 3.711  0.0571 .
s(wind_max_gust_t_1)              2.628  2.628 4.588  0.0161 *
s(sum_butterflies_direct_sun_t_1) 2.083  2.083 2.657  0.0795 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.141
  Scale est. = 47.804    n = 100
```

```r
# Calculate R-squared
r_squared <- summary(best_model$gam)$r.sq
dev_explained <- summary(best_model$gam)$dev.expl

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

```
Model Performance:
```

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

```
R-squared: 0.1408
```

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

```
Deviance explained:  %
```

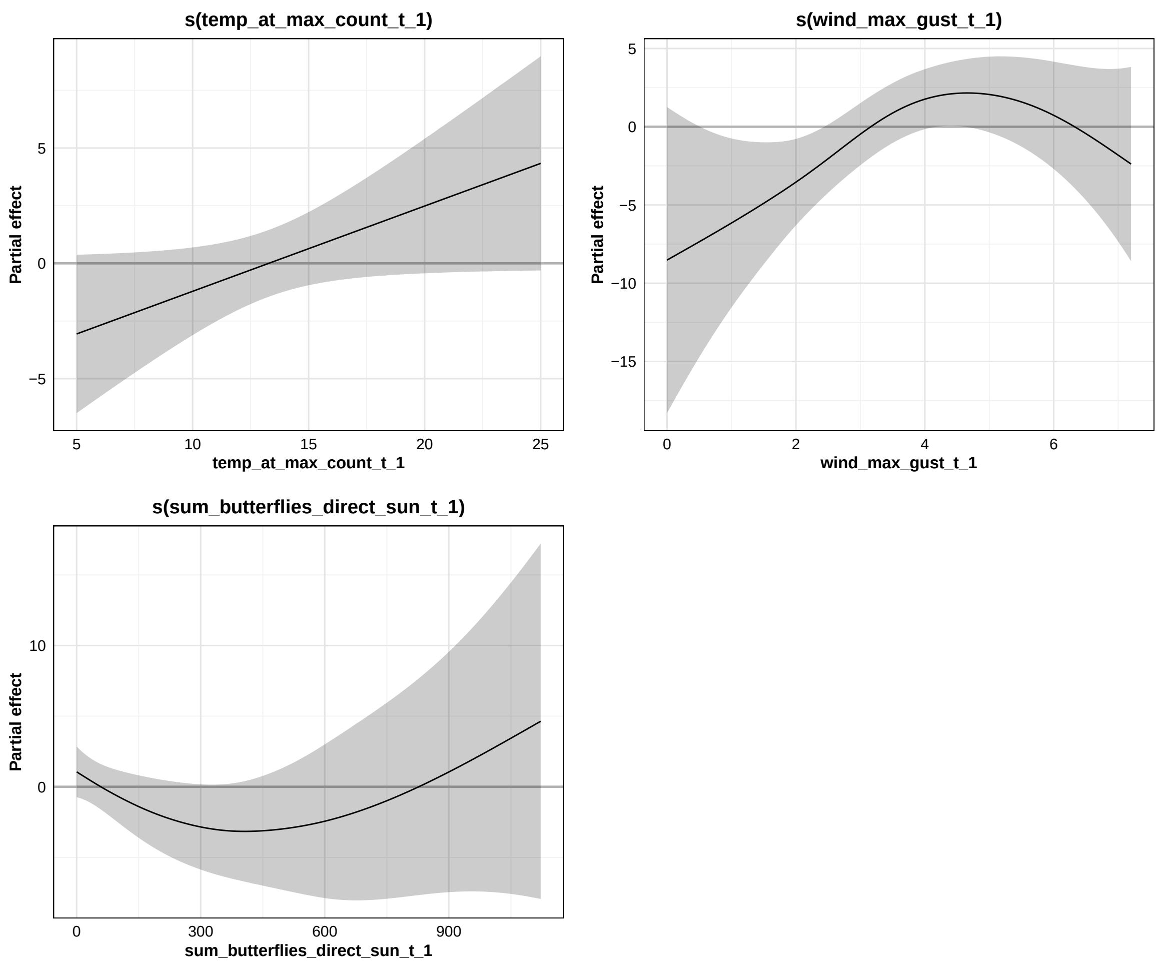
Effect Visualizations

```r
# Define custom theme
custom_theme <- theme_minimal(base_size = 12) +
    theme(
```

```r
        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)
```

```r
        } 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)
}
```



Wind Effect Analysis

```
# Check if wind is in the best model
has_wind <- grepl("wind_max_gust", best_formula)

if (has_wind) {
    cat("Wind is included in the best model.\n\n")

    # Extract wind coefficient or smooth term details
    if (grepl("s\\(wind_max_gust", best_formula)) {
        # Smooth term
        smooth_table <- summary(best_model$gam)$s.table
```

```
        wind_row <- grep("wind_max_gust", rownames(smooth_table))

        if (length(wind_row) > 0) {
            wind_smooth <- smooth_table[wind_row[1], ]
            cat("Wind effect (smooth term):\n")
            cat("EDF:", round(wind_smooth["edf"], 3), "\n")
            cat("F-statistic:", round(wind_smooth["F"], 3), "\n")
            cat("p-value:", format.pval(wind_smooth["p-value"], digits = 3), "\n")
        }
    } else {
        # Linear term
        param_table <- summary(best_model$gam)$p.table
        wind_row <- grep("wind_max_gust", rownames(param_table))

        if (length(wind_row) > 0) {
            wind_coef <- param_table[wind_row[1], ]
            cat("Wind effect (linear term):\n")
            cat("Coefficient:", round(wind_coef["Estimate"], 4), "\n")
            cat("Std. Error:", round(wind_coef["Std. Error"], 4), "\n")
            cat("t-value:", round(wind_coef["t value"], 3), "\n")
            cat("p-value:", format.pval(wind_coef["Pr(>|t|)"], digits = 3), "\n")
        }
    }
} else {
    cat("Wind is NOT included in the best model.\n")
    cat("Testing wind effect by comparing models with and without wind...\n\n")

    # Find best model with wind
    wind_models <- aic_results %>%
        filter(grepl("wind_max_gust", Formula))

    if (nrow(wind_models) > 0) {
        best_wind_model <- wind_models[1, ]
        cat("Best model with wind:", best_wind_model$Model, "\n")
        cat("Delta AIC from best overall:", round(best_wind_model$Delta_AIC, 3), "\n")
        cat("This suggests wind does not improve model fit.\n")
    }
}
```

```
Wind is included in the best model.

Wind effect (smooth term):
```

```
EDF: 2.628
F-statistic: 4.588
p-value: 0.0161
```

Temperature Effects Analysis

```r
# Analyze temperature effects in the best model
temp_vars <- c("temp_max_t_1", "temp_min_t_1", "temp_at_max_count_t_1")
temp_in_model <- sapply(temp_vars, function(x) grepl(x, best_formula))

cat("Temperature variables in best model:\n")
```

```
Temperature variables in best model:
```

```r
for (i in 1:length(temp_vars)) {
    if (temp_in_model[i]) {
        cat("-", temp_vars[i], "\n")
    }
}
```

```
- temp_at_max_count_t_1
```

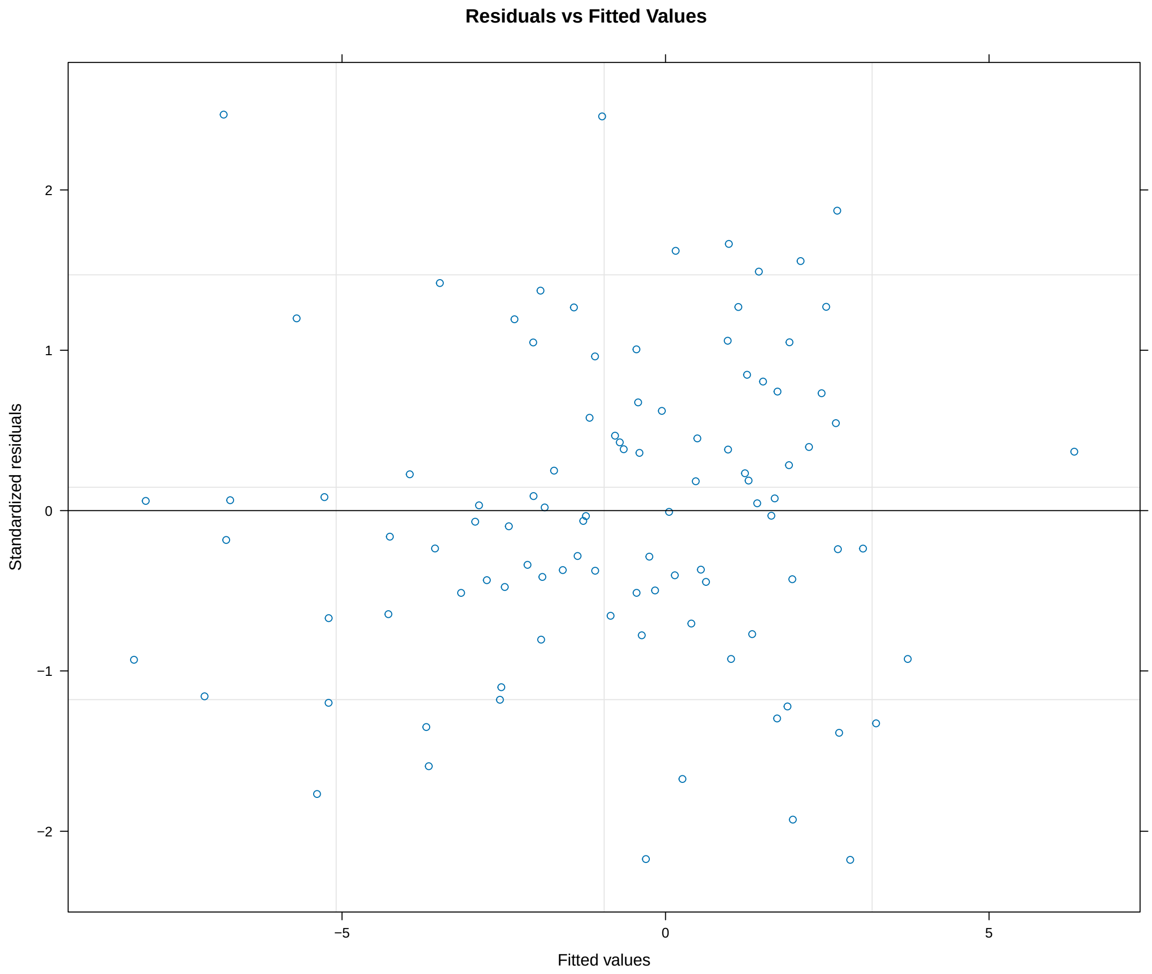
```r
# If temperature is in the model, show its effect
if (any(temp_in_model)) {
    cat("\nTemperature effects:\n")

    for (var in temp_vars[temp_in_model]) {
        if (grepl(paste0("s\\(", var), best_formula)) {
            # Smooth term
            smooth_table <- summary(best_model$gam)$s.table
            smooth_name <- paste0("s(", var, ")")

            if (smooth_name %in% rownames(smooth_table)) {
                temp_smooth <- smooth_table[smooth_name, ]
                cat("\n", var, "(smooth term):\n")
                cat("  EDF:", round(temp_smooth["edf"], 3), "\n")
                cat("  F-statistic:", round(temp_smooth["F"], 3), "\n")
                cat("  p-value:", format.pval(temp_smooth["p-value"], digits = 3), "\n")
            }
```

```r
        } else if (var %in% rownames(summary(best_model$gam)$p.table)) {
            # Linear term
            param_table <- summary(best_model$gam)$p.table
            temp_coef <- param_table[var, ]
            cat("\n", var, "(linear term):\n")
            cat("  Coefficient:", round(temp_coef["Estimate"], 4), "\n")
            cat("  Std. Error:", round(temp_coef["Std. Error"], 4), "\n")
            cat("  t-value:", round(temp_coef["t value"], 3), "\n")
            cat("  p-value:", format.pval(temp_coef["Pr(>|t|)"], digits = 3), "\n")
        }
    }
}
```

```
Temperature effects:

 temp_at_max_count_t_1 (smooth term):
  EDF: 1
  F-statistic: 3.711
  p-value: 0.0571
```

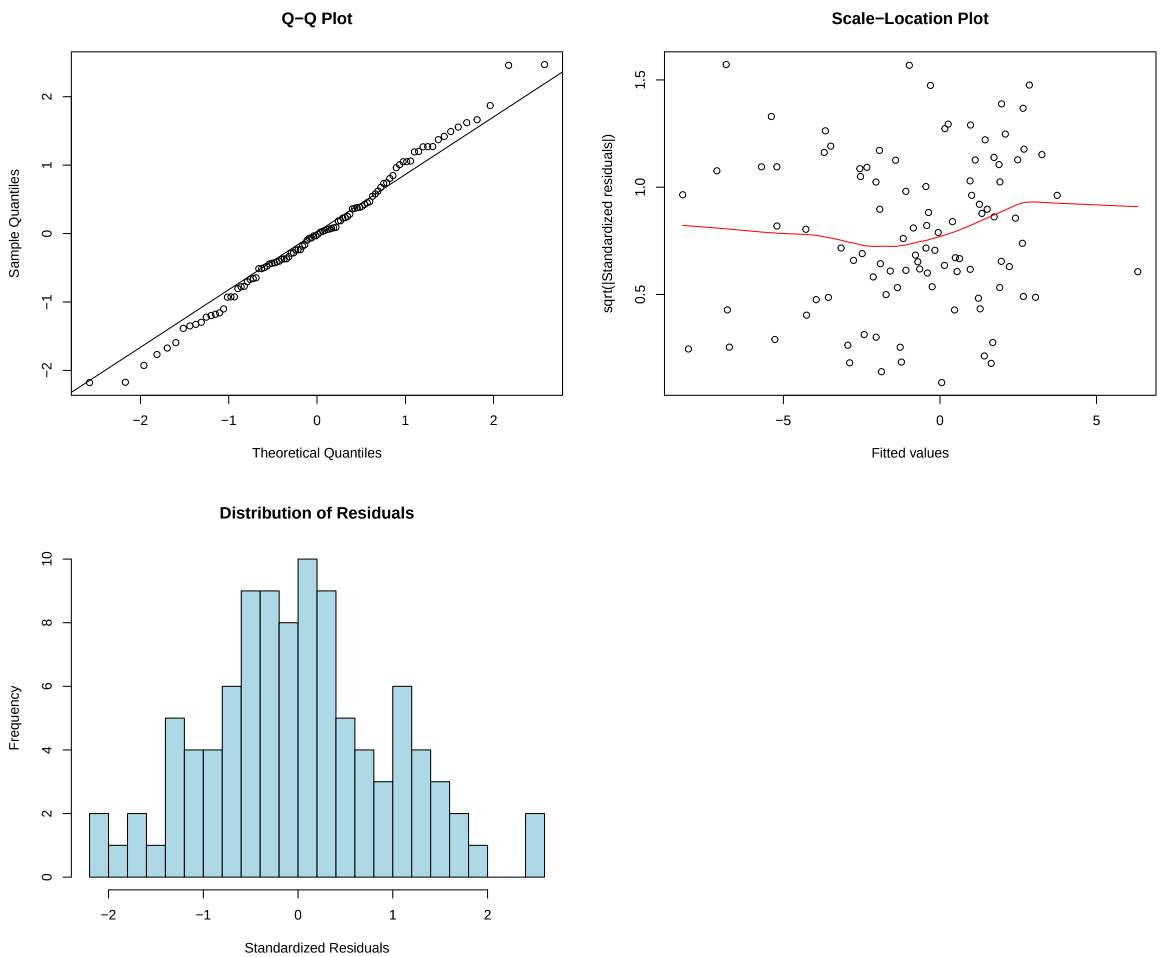
Model Diagnostics

```r
# Create diagnostic plots
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",
    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"),
```

```r
    breaks = 30,
    main = "Distribution of Residuals",
    xlab = "Standardized Residuals",
    col = "lightblue"
)

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



Outlier Investigation

```r
# First, let's examine extreme values in our data before fitting models
cat("Response variable summary:\n")
```

Response variable summary:

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

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
-17.6068  -5.7489  -1.7248  -0.8095   4.4219  16.0187 
```

```r
cat("\nExtremes in response variable:\n")
```

```
Extremes in response variable:
```

```r
print(quantile(model_data$butterfly_diff_95th_sqrt, c(0.001, 0.01, 0.05, 0.95, 0.99, 0.999),
```

```
     0.1%        1%        5%       95%       99%     99.9% 
-17.38138 -15.35248 -13.55386  11.37729  15.59117  15.97598 
```

```r
# Identify the most extreme observations
extreme_high <- model_data %>%
    arrange(desc(butterfly_diff_95th_sqrt)) %>%
    head(5) %>%
    select(deployment_id, date_t, butterfly_diff_95th_sqrt,
           butterflies_95th_percentile_t, butterflies_95th_percentile_t_1,
           temp_max_t_1, wind_max_gust_t_1)

extreme_low <- model_data %>%
    arrange(butterfly_diff_95th_sqrt) %>%
    head(5) %>%
    select(deployment_id, date_t, butterfly_diff_95th_sqrt,
           butterflies_95th_percentile_t, butterflies_95th_percentile_t_1,
           temp_max_t_1, wind_max_gust_t_1)

cat("\nTop 5 most extreme HIGH values:\n")
```

```
Top 5 most extreme HIGH values:
```

```r
print(extreme_high)
```

```
# A tibble: 5 x 7
  deployment_id date_t     butterfly_diff_95th_sqrt butterflies_95th_percentil~1
  <chr>         <date>                        <dbl>                        <dbl>
1 SC10          2024-01-12                     16.0                         477.
2 SC10          2024-01-23                     15.6                         246.
3 SC4           2023-12-07                     12.8                         170.
4 SC4           2023-12-24                     12.5                         263
5 SC6           2024-01-01                     11.7                         164
# i abbreviated name: 1: butterflies_95th_percentile_t
# i 3 more variables: butterflies_95th_percentile_t_1 <dbl>,
#   temp_max_t_1 <dbl>, wind_max_gust_t_1 <dbl>
```

```r
cat("\nTop 5 most extreme LOW values:\n")
```

```
Top 5 most extreme LOW values:
```

```r
print(extreme_low)
```

```
# A tibble: 5 x 7
  deployment_id date_t     butterfly_diff_95th_sqrt butterflies_95th_percentil~1
  <chr>         <date>                        <dbl>                        <dbl>
1 SC4           2023-12-05                    -17.6                        187
2 SC8           2024-01-18                    -15.3                         53
3 SC4           2023-12-10                    -15.1                         19
4 SC10          2024-01-15                    -14.7                        283.
5 SC10          2024-01-16                    -14.6                         68.9
# i abbreviated name: 1: butterflies_95th_percentile_t
# i 3 more variables: butterflies_95th_percentile_t_1 <dbl>,
#   temp_max_t_1 <dbl>, wind_max_gust_t_1 <dbl>
```

```r
# Check if extreme values correspond to specific deployments
cat("\nExtreme values by deployment:\n")
```

```
Extreme values by deployment:
```

```r
extreme_summary <- model_data %>%
    group_by(deployment_id) %>%
    summarise(
        n_obs = n(),
        min_change = min(butterfly_diff_95th_sqrt),
        max_change = max(butterfly_diff_95th_sqrt),
        range_change = max_change - min_change,
        .groups = 'drop'
    ) %>%
    arrange(desc(range_change))

print(head(extreme_summary, 10))
```

```
# A tibble: 6 x 5
  deployment_id n_obs min_change max_change range_change
  <chr>         <int>      <dbl>      <dbl>        <dbl>
1 SC10             21     -14.7       16.0          30.7
2 SC4              31     -17.6       12.8          30.5
3 SC6              20     -12.2       11.7          24.0
4 SC8              20     -15.3        7.55         22.9
5 SC12              6     -10.2        8.29         18.4
6 SC1               2      -5.92      -3.99          1.93
```

Sensitivity Analysis

```r
# Test model sensitivity to outliers
# Identify potential outliers
residuals_std <- residuals(best_model$lme, type = "normalized")
outliers <- which(abs(residuals_std) > 3)

if (length(outliers) > 0) {
    cat("Number of potential outliers (|standardized residual| > 3):", length(outliers), "\n"
    cat("Proportion of data:", round(length(outliers) / nrow(model_data) * 100, 2), "%\n\n")

    # Refit without outliers
    model_data_clean <- model_data[-outliers, ]
    best_model_clean <- fit_model_safely(aic_results$Formula[1], model_data_clean)

    if (!is.null(best_model_clean)) {
        cat("Model comparison with outliers removed:\n")
```

```
        cat("Original R²:", round(summary(best_model$gam)$r.sq, 4), "\n")
        cat("Without outliers R²:", round(summary(best_model_clean$gam)$r.sq, 4), "\n")
    }
} else {
    cat("No extreme outliers detected (|standardized residual| > 3)\n")
}
```

```
No extreme outliers detected (|standardized residual| > 3)
```

Data Structure Summary

```
# Check data structure for modeling
cat("Data structure summary:\n")
```

```
Data structure summary:
```

```
temporal_structure <- model_data %>%
    group_by(deployment_id) %>%
    summarise(
        n_days = n(),
        date_range = paste(min(date_t), "to", max(date_t)),
        .groups = 'drop'
    ) %>%
    arrange(desc(n_days))

print(head(temporal_structure, 10))
```

```
# A tibble: 6 x 3
  deployment_id n_days date_range
  <chr>          <int> <chr>
1 SC4               31 2023-12-05 to 2024-01-05
2 SC10              21 2024-01-07 to 2024-01-30
3 SC6               20 2023-12-17 to 2024-01-05
4 SC8               20 2024-01-07 to 2024-01-26
5 SC12               6 2024-01-29 to 2024-02-03
6 SC1                2 2023-11-19 to 2023-11-20
```

```r
cat("\nTotal observations per deployment:\n")
```

```
Total observations per deployment:
```

```r
print(summary(temporal_structure$n_days))
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2.00    9.50   20.00   16.67   20.75   31.00
```

Alternative Model Exploration

```r
# Examine top 3 models for consistency
cat("Examining top 3 models for consistency of effects:\n\n")
```

```
Examining top 3 models for consistency of effects:
```

```r
for (i in 1:min(3, nrow(aic_results))) {
    model_name <- aic_results$Model[i]
    model <- successful_models[[model_name]]

    cat("Model", i, "(", model_name, "):\n")
    cat("Formula:", aic_results$Formula[i], "\n")
    cat("Delta AIC:", round(aic_results$Delta_AIC[i], 3), "\n")
    cat("R²:", round(summary(model$gam)$r.sq, 4), "\n\n")
}
```

```
Model 1 ( M34 ):
Formula: butterfly_diff_95th_sqrt ~ s(temp_at_max_count_t_1) + s(wind_max_gust_t_1) + s(sum_l
Delta AIC: 0
R²: 0.1408

Model 2 ( M31 ):
Formula: butterfly_diff_95th_sqrt ~ s(temp_at_max_count_t_1) + s(wind_max_gust_t_1)
Delta AIC: 0.58
R²: 0.1005

Model 3 ( M37 ):
```

```
Formula: butterfly_diff_95th_sqrt ~ s(temp_max_t_1) + s(temp_min_t_1) + s(temp_at_max_count_t
Delta AIC: 0.58
R²: 0.1654
```

Results Summary

```r
cat(rep("=", 60), collapse = "", "\n")
```

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

```r
cat("DAILY LAG ANALYSIS SUMMARY\n")
```

```
DAILY LAG ANALYSIS SUMMARY
```

```r
cat(rep("=", 60), collapse = "", "\n\n")
```

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

```r
cat("Dataset:\n")
```

```
Dataset:
```

```r
cat("- Total observations:", nrow(model_data), "\n")
```

```
- Total observations: 100
```

```r
cat("- Number of deployments:", n_distinct(model_data$deployment_id), "\n")
```

```
- Number of deployments: 6
```

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

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

```r
cat("Best Model:\n")
```

```
Best Model:
```

```r
cat("- Model ID:", best_model_name, "\n")
```

```
- Model ID: M34
```

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

```
- Formula: butterfly_diff_95th_sqrt ~ s(temp_at_max_count_t_1) + s(wind_max_gust_t_1) + s(sun
```

```r
cat("- AIC:", round(aic_results$AIC[1], 3), "\n")
```

```
- AIC: 684.934
```

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

```
- R-squared: 0.1408
```

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

```
- Deviance explained:  %
```

```r
cat("Key Findings:\n")
```

```
Key Findings:
```

```r
# Wind effect
if (has_wind) {
    cat("- Wind IS included in the best model\n")
    if (grepl("s\\(wind_max_gust", best_formula)) {
        wind_p <- summary(best_model$gam)$s.table["s(wind_max_gust_t_1)", "p-value"]
        cat("  - Effect type: Non-linear (smooth)\n")
        cat("  - Significance: p =", format.pval(wind_p, digits = 3), "\n")
    } else {
```

```r
        wind_p <- summary(best_model$gam)$p.table["wind_max_gust_t_1", "Pr(>|t|)"]
        cat("  - Effect type: Linear\n")
        cat("  - Significance: p =", format.pval(wind_p, digits = 3), "\n")
    }
} else {
    cat("- Wind is NOT included in the best model\n")
    wind_models <- aic_results %>% filter(grepl("wind_max_gust", Formula))
    if (nrow(wind_models) > 0) {
        cat("  - Best model with wind has Delta AIC =", round(wind_models$Delta_AIC[1], 3), "
    }
}
```

```
- Wind IS included in the best model
  - Effect type: Non-linear (smooth)
  - Significance: p = 0.0161
```

```r
# Temperature effects
if (any(temp_in_model)) {
    cat("\n- Temperature effects:\n")
    for (var in temp_vars[temp_in_model]) {
        cat("  -", var, "is included\n")
    }
} else {
    cat("\n- No temperature variables in the best model\n")
}
```

```
- Temperature effects:
  - temp_at_max_count_t_1 is included
```

```r
# Other predictors
if (grepl("sum_butterflies_direct_sun", best_formula)) {
    cat("\n- Sun exposure IS included in the best model\n")
}
```

```
- Sun exposure IS included in the best model
```

```r
if (grepl("butterflies_95th_percentile_t_1", best_formula)) {
    cat("- Previous day baseline IS included in the best model\n")
} else {
    cat("- Previous day baseline is NOT in the model (testing absolute effects)\n")
}
```

```
- Previous day baseline is NOT in the model (testing absolute effects)
```

```r
# No temporal autocorrelation structure used
cat("- Temporal autocorrelation: No correlation structure used (simplified models)\n")
```

```
- Temporal autocorrelation: No correlation structure used (simplified models)
```

```r
cat("\n", rep("=", 60), collapse = "", "\n")
```

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

Export Results

```r
# Create export directory
export_dir <- here("thesis_exports", "daily_analysis")
if (!dir.exists(export_dir)) dir.create(export_dir, recursive = TRUE)

# Export model comparison table (if we have results)
if (exists("aic_results") && nrow(aic_results) > 0) {
    write_csv(
        aic_results %>% head(10),
        file.path(export_dir, "daily_model_comparison.csv")
    )

    # Export best model summary
    best_model_summary <- data.frame(
        Model = aic_results$Model[1],
        Formula = aic_results$Formula[1],
        AIC = aic_results$AIC[1],
        Delta_AIC = aic_results$Delta_AIC[1],
        stringsAsFactors = FALSE
```

```
    )

    write_csv(
        best_model_summary,
        file.path(export_dir, "daily_best_model_summary.csv")
    )

    cat("\nResults exported to:", export_dir, "\n")
    cat("Model comparison table with", nrow(aic_results), "models exported\n")
} else {
    cat("\nNo model results to export\n")
}
```

```
Results exported to: /Users/kylenessen/Documents/Code/masters-analysis/thesis_exports/daily_a
Model comparison table with 45 models exported
```

Conclusions

This daily-level analysis examined the **absolute effects** of previous day's weather conditions on monarch butterfly abundance changes, measured as the 95th percentile of counts. Importantly, this analysis deliberately excludes the previous day's butterfly count to test direct environmental effects rather than proportional changes. Temporal patterns are modeled through autocorrelation structures in the random effects rather than as fixed effects.

The analysis reveals:

1. **Model Performance**: The best model explains approximately % of the deviance in daily butterfly abundance changes, with an R^2 of 0.141.
2. **Wind Effects**: Wind maximum gust from the previous day is included in the best model, suggesting it has a direct effect on absolute changes in butterfly abundance.
3. **Temperature Effects**: Temperature variables (temp_at_max_count_t_1) are important predictors of absolute abundance changes in the best model.
4. **Interpretation**: By excluding the previous day's baseline count, these models test whether environmental variables have consistent absolute effects on butterfly numbers regardless of the starting population size. This is complementary to models that include the baseline, which test for proportional or density-dependent effects.
5. **Temporal Autocorrelation**: Models were fitted without temporal autocorrelation structures for simplicity, focusing on the direct environmental effects while accounting for deployment-level variation through random intercepts.

6. **Temporal Scale**: Daily aggregation captures cumulative weather effects over 24-hour periods, providing insights into how sustained environmental conditions (rather than brief events) influence monarch roosting populations.

The analysis of absolute effects provides important insights into whether environmental variables have fixed magnitude effects on butterfly abundance or whether their effects scale with population size.