Song Sparrow Dataset Analysis

TO MINH ANH

2025-01-11

Import libraries and load the dataset

```
# Load necessary libraries
library(ggplot2)
library(patchwork)  # For side-by-side plots
library(gridExtra)
library(dplyr)
```

```
Attaching package: 'dplyr'

The following object is masked from 'package:gridExtra':

    combine

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
library(lme4)
```

```
Loading required package: Matrix
```

```r
library(nlme)
```

```
Attaching package: 'nlme'

The following object is masked from 'package:lme4':

    lmList

The following object is masked from 'package:dplyr':

    collapse
```

```r
# Install and load the car package
#install.packages("car")
library(car)
```

```
Loading required package: carData


Attaching package: 'car'

The following object is masked from 'package:dplyr':

    recode
```

```r
# Load the dataset
data = read.csv("female.csv")

# Summary/Structure of the dataset
summary(data)
```

```
      band           cohort            year             fpop      
 Min.   : 2009   Min.   : 2.000   Min.   : 1.000   Min.   : 4.00  
 1st Qu.:30683   1st Qu.: 7.000   1st Qu.: 5.000   1st Qu.:42.00  
 Median :58853   Median :10.000   Median :10.000   Median :53.00  
 Mean   :46959   Mean   : 9.966   Mean   : 9.912   Mean   :49.55  
 3rd Qu.:60124   3rd Qu.:12.000   3rd Qu.:13.000   3rd Qu.:61.00  
 Max.   :86268   Max.   :19.000   Max.   :19.000   Max.   :72.00  
```

```
 NA's   :1       NA's   :101
      age             spf               x                y
 Min.   :1.000   Min.   : 0.000   Min.   : 0.25   Min.   :1.375
 1st Qu.:1.000   1st Qu.: 2.000   1st Qu.:10.90   1st Qu.:2.433
 Median :2.000   Median : 3.000   Median :17.50   Median :3.250
 Mean   :1.936   Mean   : 3.369   Mean   :17.19   Mean   :3.092
 3rd Qu.:3.000   3rd Qu.: 5.000   3rd Qu.:23.75   3rd Qu.:3.625
 Max.   :7.000   Max.   :12.000   Max.   :32.83   Max.   :5.250
 NA's   :101                      NA's   :5       NA's   :5
```

```
#glimpse(data)
```

```
length(unique(data$band))
```

```
[1] 360
```

The data set comprises 742 entries and 8 columns, with data corresponding to 360 unique sparrows included in the study. In this analysis, `year` and `cohort` are converted into categorical variables.

```
# Convert year and cohort to factors
data$year = as.factor(data$year)
data$cohort = as.factor(data$cohort)
```

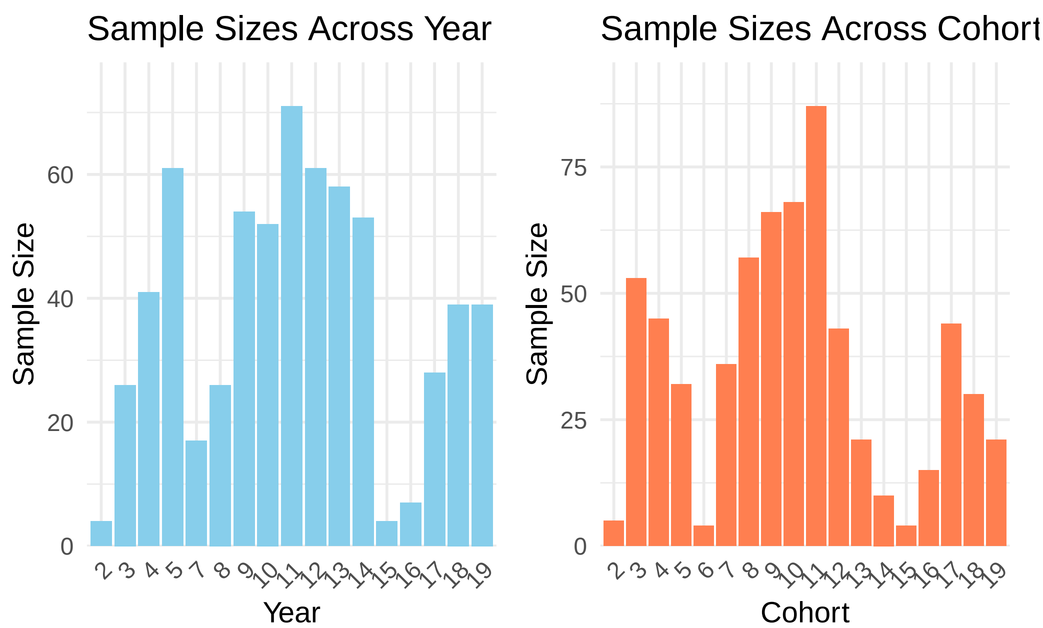
Study Design

```
# Handle missing values
data_clean = na.omit(data)
```

Data Description

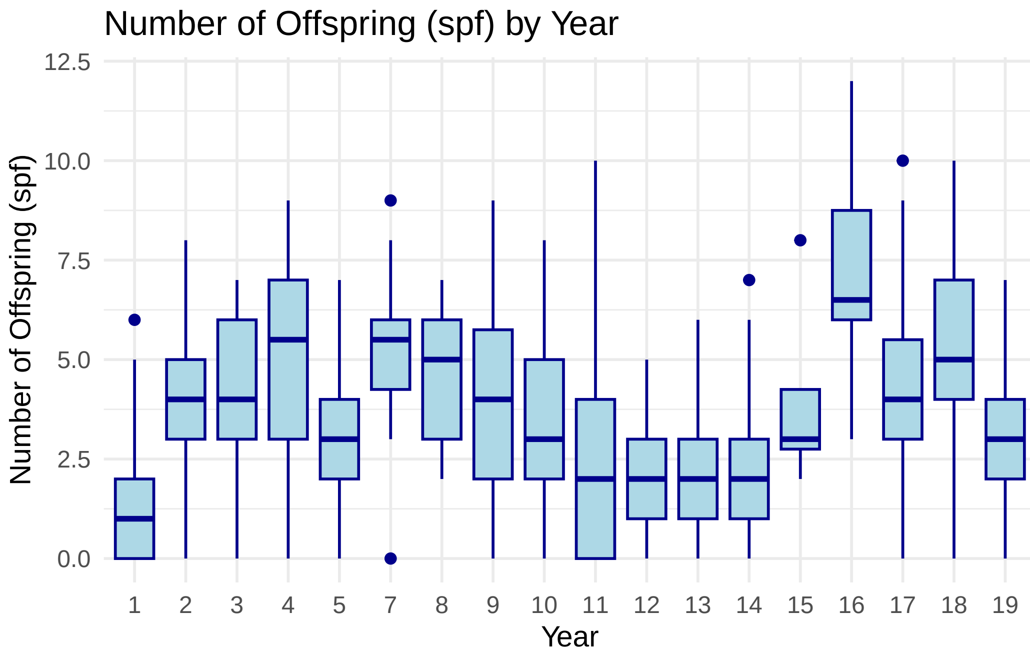
Visualizing the Relationship Between the Number of Offspring and Grouping Factors

```r
# Ensure 'year' and 'cohort' are numeric (convert factor to numeric if necessary)
data$year = as.numeric(as.character(data$year))
data$cohort = as.numeric(as.character(data$cohort))

# Drop missing values for 'year' and 'cohort'
data_clean = data[!is.na(data$year) & !is.na(data$cohort), ]

# Count sample sizes for each year
year_counts = as.data.frame(table(data_clean$year))
colnames(year_counts) = c("year", "sample_size")
year_counts$year = as.numeric(as.character(year_counts$year))

# Count sample sizes for each cohort
cohort_counts = as.data.frame(table(data_clean$cohort))
colnames(cohort_counts) = c("cohort", "sample_size")
cohort_counts$cohort = as.numeric(as.character(cohort_counts$cohort))

# Create the plot for year
plot_year = ggplot(year_counts, aes(x = as.factor(year), y = sample_size)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(title = "Sample Sizes Across Year", x = "Year", y = "Sample Size") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_y_continuous(expand = expansion(mult = c(0, 0.1)))

# Create the plot for cohort
plot_cohort = ggplot(cohort_counts, aes(x = as.factor(cohort), y = sample_size)) +
  geom_bar(stat = "identity", fill = "coral") +
  labs(title = "Sample Sizes Across Cohort", x = "Cohort", y = "Sample Size") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_y_continuous(expand = expansion(mult = c(0, 0.1)))

# Combine the two plots side by side
combined_plot = plot_year + plot_cohort

# Print the combined plot
print(combined_plot)
```

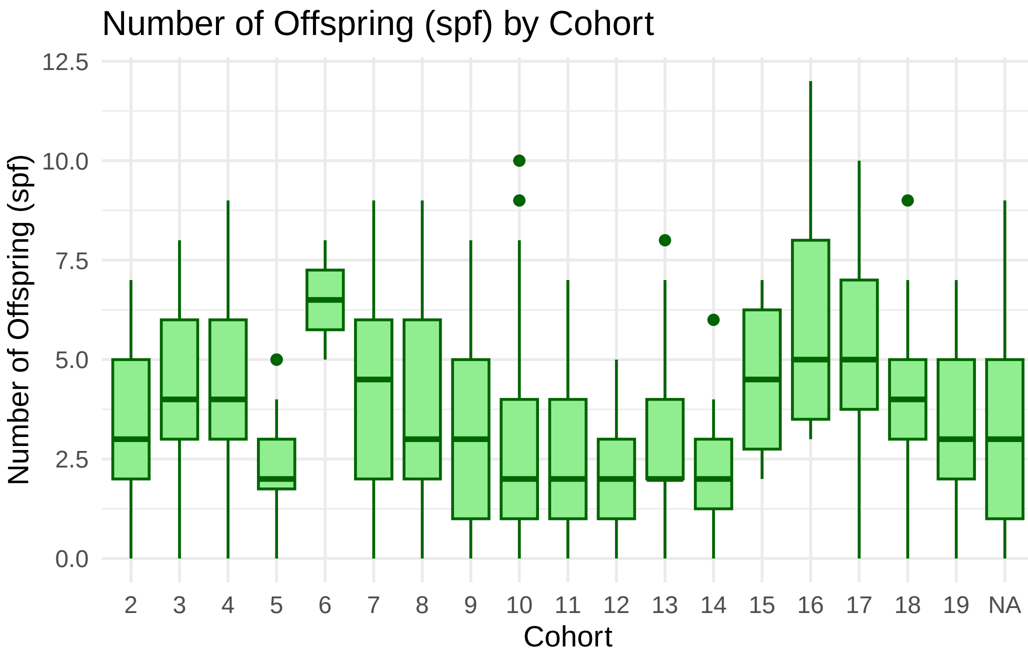



```r
# Convert year and cohort to factors
data$year = as.factor(data$year)
data$cohort = as.factor(data$cohort)
```

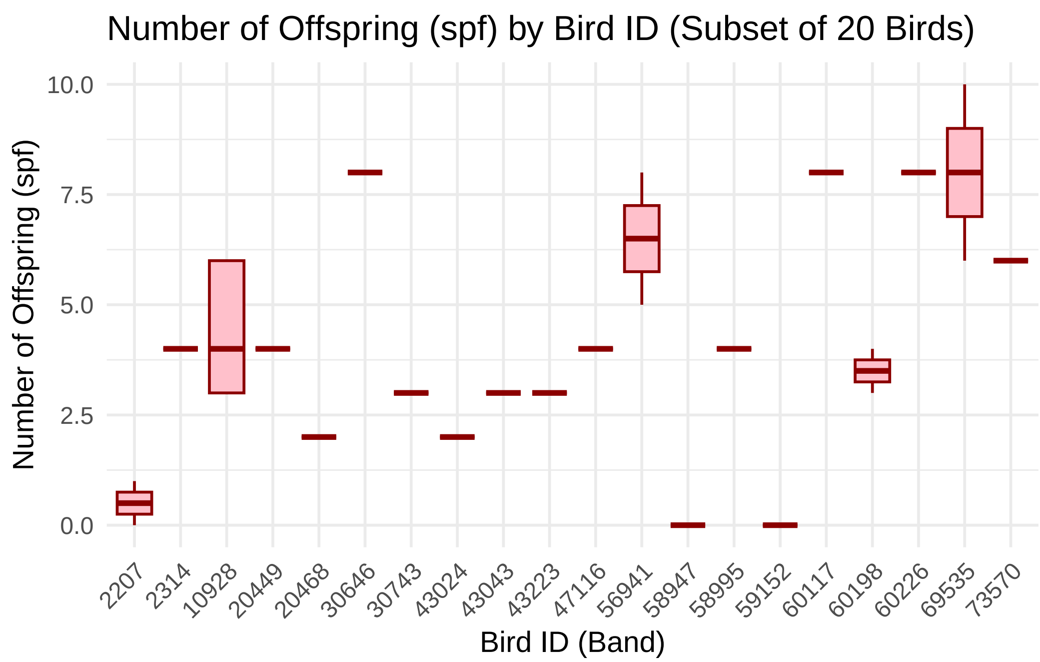
```r
# Box plot of spf by Year (Macro Grouping Factor)
ggplot(data, aes(x = as.factor(year), y = spf)) +
  geom_boxplot(fill = "lightblue", color = "darkblue") +
  labs(
    title = "Number of Offspring (spf) by Year",
    x = "Year",
    y = "Number of Offspring (spf)"
  ) +
  theme_minimal()
```




```r
# Box plot of spf by Cohort (Can be Macro or Micro Grouping Factor)
ggplot(data, aes(x = as.factor(cohort), y = spf)) +
  geom_boxplot(fill = "lightgreen", color = "darkgreen") +
  labs(
    title = "Number of Offspring (spf) by Cohort",
    x = "Cohort",
    y = "Number of Offspring (spf)"
  ) +
  theme_minimal()
```




```
# Box plot of spf by Bird ID (Micro Grouping Factor)
# Since there may be too many birds, we'll limit to a subset of birds
subset_birds = data %>%
  filter(band %in% sample(unique(band), 20))  # Sample 20 random birds for visualization

ggplot(subset_birds, aes(x = as.factor(band), y = spf)) +
  geom_boxplot(fill = "pink", color = "darkred") +
  labs(
    title = "Number of Offspring (spf) by Bird ID (Subset of 20 Birds)",
    x = "Bird ID (Band)",
    y = "Number of Offspring (spf)"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))  # Rotate x-axis labels for readal
```



Relationship Between Cohort, Year, and Age

Check if all the values in the columns `age`, `year`, and `cohort` follow the formula `age=year-cohort+1`

```r
# Ensure year, cohort, and age are numeric (in case they are read as factors)
data$year = as.numeric(as.character(data$year))
data$cohort = as.numeric(as.character(data$cohort))
data$age = as.numeric(as.character(data$age))

# Remove rows with missing values in age, year, or cohort
filtered_data = na.omit(data[, c("age", "year", "cohort")])

# Check if the formula holds for all rows
valid_formula = filtered_data$age == (filtered_data$year - filtered_data$cohort + 1)

# Check if all the values satisfy the condition
all_valid = all(valid_formula, na.rm = TRUE)

# Print result
if (all_valid) {
  print("All non-missing rows satisfy the formula age = year - cohort + 1")
```

```r
} else {
  print("There are rows where the formula does not hold.")
}
```

```
[1] "All non-missing rows satisfy the formula age = year - cohort + 1"
```

```r
# See which rows violate the formula, print them
violating_rows = filtered_data[!valid_formula, ]
if (nrow(violating_rows) > 0) {
  print("Rows that do not satisfy the formula:")
  print(violating_rows)
}
```

```r
# Get all rows with at least one missing value
rows_with_missing = data[!complete.cases(data), ]

# Count the number of rows with missing values
num_rows_with_missing = nrow(rows_with_missing)
print(paste("Number of rows with at least one missing value:", num_rows_with_missing))
```

```
[1] "Number of rows with at least one missing value: 106"
```

```r
# Create a table
table_data = table(data$year, data$cohort)
print(table_data)
```

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	16	25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	12	17	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	3	0	3	11	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	1	8	17	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	7	15	32	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	5	11	14	22	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	5	8	10	17	31	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	3	5	13	21	19	0	0	0	0	0	0	0
13	0	0	0	0	0	0	2	4	9	19	13	11	0	0	0	0	0	0

```
  14   0   0   0   0   0   0   1   1   7  16   9   9  10   0   0   0   0   0
  15   0   0   0   0   0   0   0   0   0   0   1   1   0   2   0   0   0   0
  16   0   0   0   0   0   0   0   0   0   0   1   0   0   1   5   0   0   0
  17   0   0   0   0   0   0   0   0   0   0   0   0   0   0   5  23   0   0
  18   0   0   0   0   0   0   0   0   0   0   0   0   0   1   3  16  19   0
  19   0   0   0   0   0   0   0   0   0   0   0   0   0   0   2   5  11  21
```

```r
# Calculate the correlation between year and cohort
cor(as.numeric(data$year), as.numeric(data$cohort), use = "complete.obs")
```

```
[1] 0.9679038
```

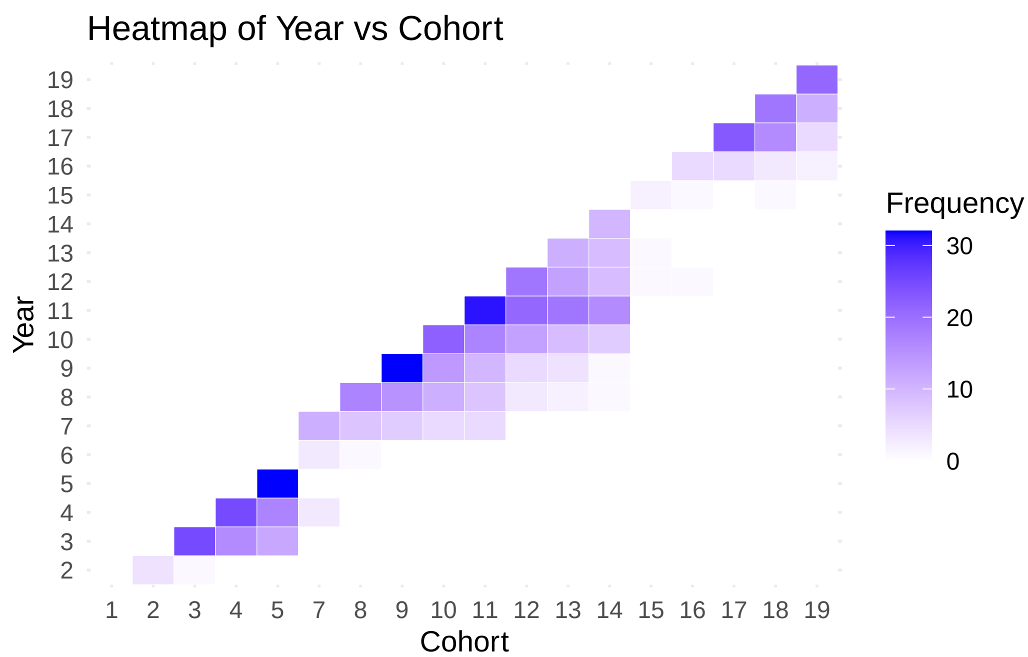
```r
cor(as.numeric(data$year), as.numeric(data$age), use = "complete.obs")
```

```
[1] 0.1766496
```

```r
cor(as.numeric(data$cohort), as.numeric(data$age), use = "complete.obs")
```

```
[1] -0.07638897
```

```r
# Convert table to data frame
df_table = as.data.frame(table(data$year, data$cohort))

# Plot as heatmap
ggplot(df_table, aes(Var1, Var2, fill = Freq)) +
  geom_tile(color = "white") +
  scale_fill_gradient(low = "white", high = "blue") +
  labs(title = "Heatmap of Year vs Cohort",
       x = "Cohort",
       y = "Year",
       fill = "Frequency") +
  theme_minimal()
```



Preliminary Modeling

Sequential Hypothesis Testing

Micro Variables

```
# Fit the full model
lm_full = lm(spf ~ age + fpop + x + y, data = data)
# Set contrasts to be compatible with Type III ANOVA
options(contrasts = c("contr.sum", "contr.poly"))

# Perform Type III ANOVA
Anova(lm_full, type = "III")
```

```
Anova Table (Type III tests)

Response: spf
             Sum Sq  Df  F value    Pr(>F)
(Intercept)  999.33   1 209.2728 < 2.2e-16 ***
age           14.77   1   3.0927  0.079131 .
fpop         423.38   1  88.6618 < 2.2e-16 ***
x             49.98   1  10.4669  0.001279 **
```

```
y               0.09   1   0.0196  0.888700
Residuals    3013.18 631
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(lm_full)
```

```
Call:
lm(formula = spf ~ age + fpop + x + y, data = data)

Residuals:
    Min      1Q  Median      3Q     Max
-5.0700 -1.5779 -0.2345  1.5303  7.5555

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  7.242896   0.500675  14.466  < 2e-16 ***
age         -0.136523   0.077632  -1.759  0.07913 .
fpop        -0.058092   0.006169  -9.416  < 2e-16 ***
x           -0.035778   0.011059  -3.235  0.00128 **
y            0.016041   0.114577   0.140  0.88870
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.185 on 631 degrees of freedom
  (106 observations deleted due to missingness)
Multiple R-squared:  0.155, Adjusted R-squared:  0.1496
F-statistic: 28.94 on 4 and 631 DF,  p-value: < 2.2e-16
```

```r
vif(lm_full)  # Variance Inflation Factor
```

```
     age     fpop        x        y
1.063113 1.045432 1.025550 1.018981
```

Macro Variables

```r
lm = lm(spf ~ fpop + x + age, data = data_clean)

lm_inter1 = lm(spf ~ fpop + x + age + fpop:age, data = data_clean)
```

```r
lm_inter2 = lm(spf ~ fpop + x + age + x:y, data = data_clean)

lm_cohort = lm(spf ~ fpop + x + age + as.factor(cohort), data = data_clean)
lm_year = lm(spf ~ fpop + x + age + as.factor(year), data = data_clean)
lm_both = lm(spf ~ fpop + x + age + as.factor(year) + as.factor(cohort), data = data_clean)

anova(lm, lm_year)
```

```
Analysis of Variance Table

Model 1: spf ~ fpop + x + age
Model 2: spf ~ fpop + x + age + as.factor(year)
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1    632 3013.3
2    617 2602.8 15    410.45 6.4865 5.706e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(lm, lm_cohort)
```

```
Analysis of Variance Table

Model 1: spf ~ fpop + x + age
Model 2: spf ~ fpop + x + age + as.factor(cohort)
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1    632 3013.3
2    615 2681.2 17    332.05 4.4802 7.452e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(lm_cohort, lm_both)
```

```
Analysis of Variance Table

Model 1: spf ~ fpop + x + age + as.factor(cohort)
Model 2: spf ~ fpop + x + age + as.factor(year) + as.factor(cohort)
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1    615 2681.2
2    601 2504.4 14    176.79 3.0303 0.0001539 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(lm_year, lm_both)
```

```
Analysis of Variance Table

Model 1: spf ~ fpop + x + age + as.factor(year)
Model 2: spf ~ fpop + x + age + as.factor(year) + as.factor(cohort)
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1    617 2602.8
2    601 2504.4 16    98.385 1.4756 0.1028
```

```r
AIC(lm, lm_inter1, lm_inter2, lm_cohort, lm_year, lm_both)
```

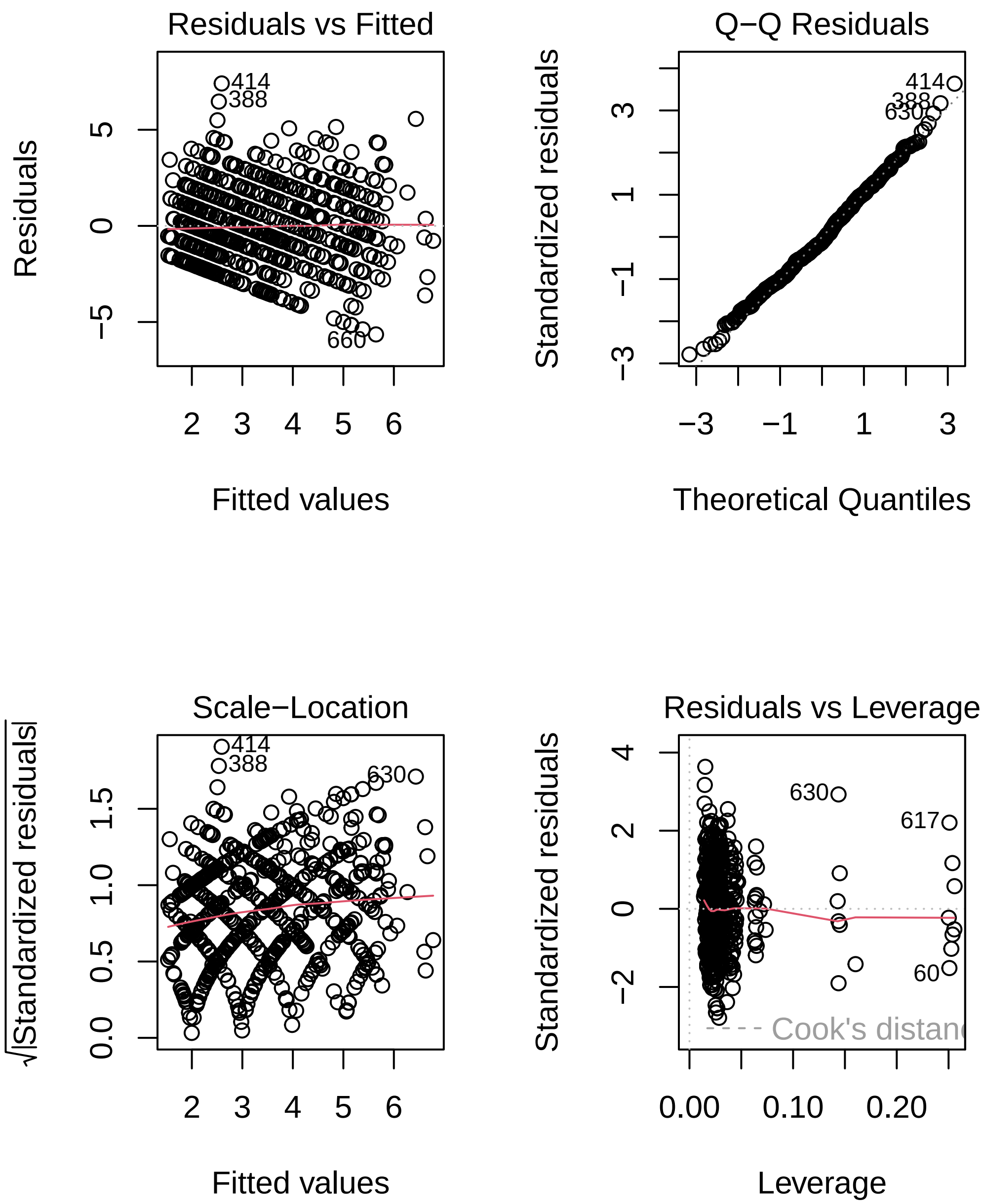
```
          df      AIC
lm         5 2804.242
lm_inter1  6 2806.013
lm_inter2  6 2806.176
lm_cohort 22 2763.987
lm_year   20 2741.112
lm_both   36 2748.606
```

```r
BIC(lm, lm_inter1, lm_inter2, lm_cohort, lm_year, lm_both)
```

```
          df      BIC
lm         5 2826.518
lm_inter1  6 2832.744
lm_inter2  6 2832.907
lm_cohort 22 2862.001
lm_year   20 2830.216
lm_both   36 2908.993
```

Checking assumptions

```r
# Diagnostic plots for assumptions
par(mfrow = c(1, 2))  # Arrange plots in a grid
plot(lm_year)  # Residuals vs fitted, Q-Q plot, Scale-location, Residuals leverage
```


Residuals vs Fitted
Residuals
Fitted values
414
388
660
Q−Q Residuals
Standardized residuals
Theoretical Quantiles
414
388
630
Scale−Location
√|Standardized residuals|
Fitted values
414
388
630
Residuals vs Leverage
Standardized residuals
Leverage
630
617
60
Cook's distanc

Model fitting and diagnostics

Linear Mixed-Effects Models

```
fit1 = lmer(spf ~ fpop + x + age + (1 | band) + (1 | year), data = data_clean)
fit2 = lmer(spf ~ fpop + x + age + (1 | year), data = data)
fit3 = lmer(spf ~ fpop + x + age + (1 | band), data = data)
BIC(fit1, fit2, fit3)
```

```
     df      BIC
fit1  7 2809.272
fit2  6 2806.269
fit3  6 2843.636
```

Modeling Temporal Correlation

```
# Handle missing values
data_clean = na.omit(data)
```

```
# Fit the corAR1 model with year as the grouping factor
fit4 = lme(
  spf ~ fpop + x + age,                 # Fixed effects
  random = ~1 | year,                   # Random intercept for year
  correlation = corAR1(),               # AR(1) structure for temporal correlation
  data = data_clean,                    # Cleaned dataset
  method = "ML"                         # Use Maximum Likelihood for comparison
)

# Check BIC of the model
BIC(fit4)
```

```
[1] 2789.482
```

```
# Fit a model with random slopes for age by year
fit5 = lme(
  spf ~ fpop + x + age,                  # Fixed effects
  random = list(year = pdDiag(~ age)),   # Random intercept and slope for age by year
  correlation = corAR1(),                # AR(1) structure for temporal correlation
```

```
  data = data_clean,
  method = "ML"
)

# Compare BIC with fit_corAR1
BIC(fit4, fit5)
```

```
     df      BIC
fit4  7 2789.482
fit5  8 2791.575
```

Checking interactions

```
add1(fit4, "fpop*age", data = data, test = "Chisq")
```

```
Single term additions

Model:
spf ~ fpop + x + age
         Df    AIC     LRT Pr(>Chi)
<none>      2758.3
fpop*age  1 2760.2 0.10305   0.7482
```

```
add1(fit4, "fpop*x", data = data, test = "Chisq")
```

```
Single term additions

Model:
spf ~ fpop + x + age
       Df    AIC     LRT Pr(>Chi)
<none>    2758.3
fpop*x  1 2759.5 0.83423   0.3611
```

```
add1(fit4, "x*y", data = data, test = "Chisq")
```

```
Single term additions

Model:
spf ~ fpop + x + age
```

```
       Df    AIC     LRT Pr(>Chi)
<none>    2758.3
x*y     2 2761.8 0.46732   0.7916
```

```r
add1(fit4, "fpop*y", data = data, test = "Chisq")
```

```
Single term additions

Model:
spf ~ fpop + x + age
       Df    AIC     LRT Pr(>Chi)
<none>    2758.3
fpop*y  2 2761.9 0.38445   0.8251
```

```r
add1(fit4, "age*y", data = data, test = "Chisq")
```

```
Single term additions

Model:
spf ~ fpop + x + age
       Df    AIC    LRT Pr(>Chi)
<none>    2758.3
age*y   2 2757.6 4.6579   0.0974 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
add1(fit4, "age*x", data = data, test = "Chisq")
```

```
Single term additions

Model:
spf ~ fpop + x + age
       Df    AIC    LRT Pr(>Chi)
<none>    2758.3
age*x   1 2759.4 0.8695   0.3511
```

```r
add1(fit4, "band*age", data = data, test = "Chisq")
```

```
Single term additions

Model:
spf ~ fpop + x + age
         Df    AIC    LRT Pr(>Chi)
<none>      2758.3
band*age  2 2757.5 4.7939    0.091 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
add1(fit4, "band*x", data = data, test = "Chisq")
```

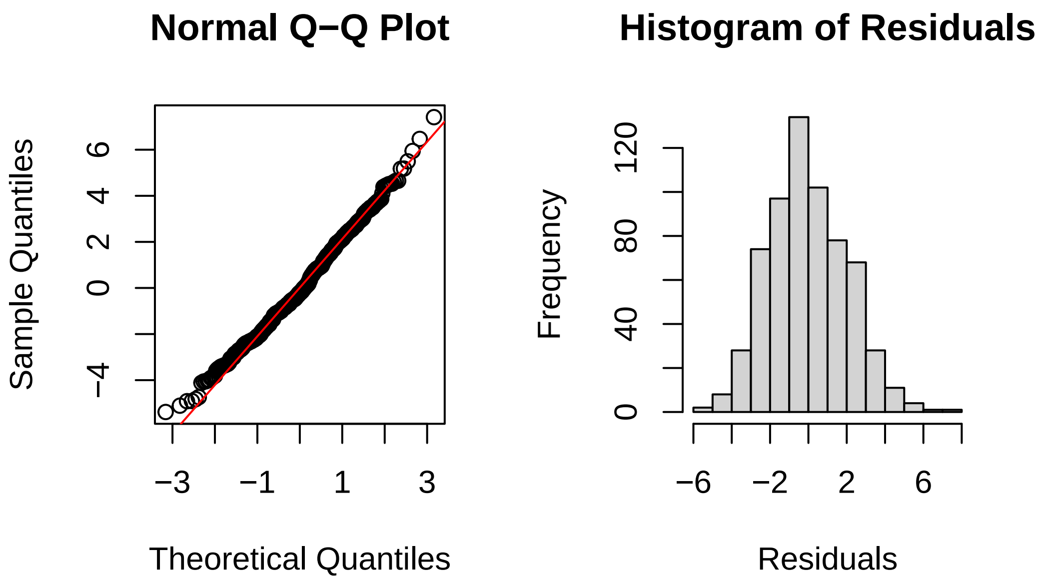
```
Single term additions

Model:
spf ~ fpop + x + age
       Df    AIC      LRT Pr(>Chi)
<none>    2758.3
band*x  2 2762.3 0.035903   0.9822
```

```r
add1(fit4, "band*fpop", data = data, test = "Chisq")
```

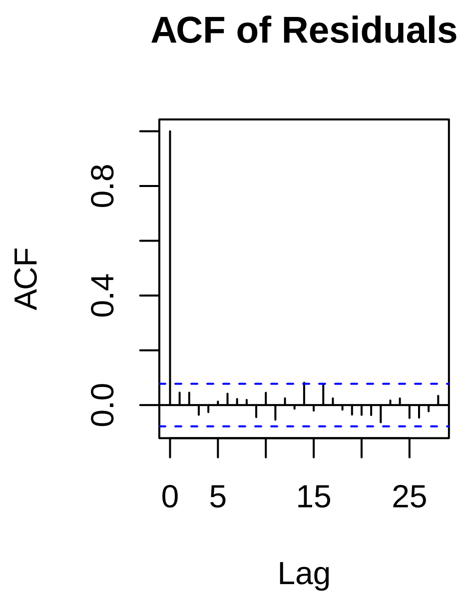
```
Single term additions

Model:
spf ~ fpop + x + age
          Df    AIC      LRT Pr(>Chi)
<none>       2758.3
band*fpop  2 2762.3 0.033896   0.9832
```

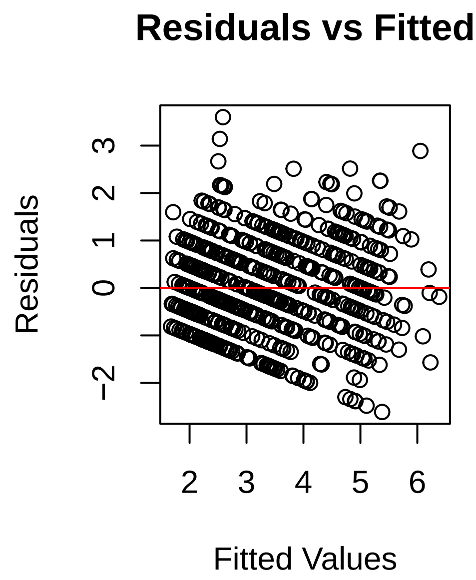
Checking assumptions

```r
par(mfrow = c(1, 2))

# Generate plots
qqnorm(resid(fit4), main = "Normal Q-Q Plot")
qqline(resid(fit4), col = "red")

hist(resid(fit4), breaks = 10, main = "Histogram of Residuals", xlab = "Residuals")
```




```r
# Set up a 2x2 plotting canvas
par(mfrow = c(1, 2))

acf(resid(fit4), main = "ACF of Residuals")

plot(fitted(fit4), residuals(fit4, type = "pearson"),
     main = "Residuals vs Fitted", xlab = "Fitted Values", ylab = "Residuals")
abline(h = 0, col = "red")
```





Model and data analysis interpretation

```
# Summary of the final model
summary(fit4)
```

```
Linear mixed-effects model fit by maximum likelihood
  Data: data_clean
       AIC      BIC    logLik
  2758.296 2789.482 -1372.148

Random effects:
 Formula: ~1 | year
        (Intercept) Residual
StdDev:   0.7567319 2.059245

Correlation Structure: AR(1)
 Formula: ~1 | year
 Parameter estimate(s):
       Phi
0.09130045
```

```
Fixed effects:  spf ~ fpop + x + age
                Value Std.Error  DF   t-value p-value
(Intercept)  6.700299 0.6385954 617 10.492244  0.0000
fpop        -0.054218 0.0125607  15 -4.316481  0.0006
x           -0.033736 0.0102683 617 -3.285464  0.0011
age          0.049070 0.0861044 617  0.569891  0.5690
 Correlation:
     (Intr) fpop   x
fpop -0.873
x    -0.249 -0.005
age  -0.140 -0.090 -0.092

Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-2.6136498 -0.6845426 -0.1059738  0.6984295  3.6000961

Number of Observations: 636
Number of Groups: 17
```

```r
# Compute confidence intervals for the model
intervals(fit4, level = 0.95)  # Default is 95% confidence level
```

```
Approximate 95% confidence intervals

 Fixed effects:
                  lower        est.       upper
(Intercept)  5.45016452  6.70029859  7.95043267
fpop        -0.08090599 -0.05421788 -0.02752978
x           -0.05383743 -0.03373598 -0.01363452
age         -0.11949053  0.04907012  0.21763078

 Random Effects:
  Level: year
                    lower      est.    upper
sd((Intercept)) 0.4740508 0.7567319 1.207979

 Correlation structure:
          lower       est.     upper
Phi 0.008649042 0.09130045 0.1727128

 Within-group standard error:
   lower     est.    upper
```

```
1.945922 2.059245 2.179166
```

```r
# Install from CRAN
install.packages("glmmTMB")
```

```
Installing package into '/home/guest/R/x86_64-pc-linux-gnu-library/4.4'
(as 'lib' is unspecified)
```

```r
# If you need lme4 as well
install.packages("lme4")
```

```
Installing package into '/home/guest/R/x86_64-pc-linux-gnu-library/4.4'
(as 'lib' is unspecified)
```

```r
# Optional but recommended: install for model comparison and diagnostics
install.packages(c("performance", "DHARMa"))
```

```
Installing packages into '/home/guest/R/x86_64-pc-linux-gnu-library/4.4'
(as 'lib' is unspecified)
```

```
also installing the dependencies 'bayestestR', 'insight', 'datawizard'
```

```r
# Load the packages after installation
library(glmmTMB)
```

```
Warning in check_dep_version(dep_pkg = "TMB"): package version mismatch:
glmmTMB was built with TMB package version 1.9.16
Current TMB package version is 1.9.15
Please re-install glmmTMB from source or restore original 'TMB' package (see '?reinstalling'
```

```r
library(lme4)
library(performance)  # for model diagnostics
library(DHARMa)       # for residual diagnostics
```

```
This is DHARMa 0.4.7. For overview type '?DHARMa'. For recent changes, type news(package = 'I
```

```r
# Poisson mixed-effects model
fit_poisson = glmmTMB(
  spf ~ fpop + x + age +  # Fixed effects
    (1 | year),           # Random intercept for year
  family = poisson(),     # Poisson distribution
  data = data_clean,
  REML = FALSE            # Equivalent to ML method
)
```