# Black/Asian carp model selection

Eddie Wu

2023-09-06

## Introduction

This .Rmd file is to show the progress on black carp and asian carp temperature and condition analyses. Since sub-sampling from spatial autocorrelation does not give significantly different results from normal analysis for black carp, we present the results without sub-sampling here. For other asian carp species, we still subsample.

For each species, we have four different models:

1. Simple linear model (same slope, same intercept)
2. Linear additive model (same slope, different intercept)
3. Interaction model (different slope, same intercept)
4. Group-specific model (different slope, different intercept)

And we consider two temperature metrics:

1. Annual temperature
2. Winter temperature (temperature from the coldest quarter)

```r
library(ggplot2)
library(ggfortify)
library(dplyr)
library(knitr)
library(tidyverse)
library(AICcmodavg) # for AICc and akaike weights
library(pwr)

## Import data
asian.carp <- read.csv("asian carp final.csv")
asian.carp$Condition <- as.factor(asian.carp$Condition)

Black <- read.csv("eddie_carp_new.csv")
Black$condition <- as.factor(Black$condition)

## Separate by species
Grass <- asian.carp[asian.carp$Species=="Grass",]
Bighead <- asian.carp[asian.carp$Species=="Bighead",]
Silver <- asian.carp[asian.carp$Species=="Silver",]
Big.sil <- rbind(Bighead, Silver) # combine the two groups


## Define two functions for AICs
compute_akaike_weights <- function(aic_scores) {
  # Find the AIC of the best model
  aic_min <- min(aic_scores)
```

```r
  # Calculate delta AIC values
  d_aic <- aic_scores - aic_min

  # Compute Akaike weights
  akaike_weights <- exp(-0.5 * d_aic) / sum(exp(-0.5 * d_aic))

  return(akaike_weights)
}
compare_aic_scores <- function(aic_scores) {
  # Find the AIC of the best model
  aic_min <- min(aic_scores)

  # Determining if the smallest value is 2 units smaller than the others
  is_smaller_by_two <- all(aic_min + 2 <= aic_scores[aic_scores != aic_min])

  # Return the index if
  if (is_smaller_by_two) {
    min_index <- which(aic_scores == aic_min)
    return(min_index)
  } else {
    return(-999)
  }
}
```

## Black carp

For black carp data, we do not subsample at any distances. But we removed the South Ukarine data point for all the following analyses.

### Temperature prediction of black carp AAM

```r
# Clean data
Black <- Black %>% filter(!row_number() == 5) %>% filter(sex != "male")

# Remove the South Ukarine data point
black.clean <- Black %>% filter(!row_number() == 20)

## Build the models with three temperature metrics
black.annual <- lm(log(AAM)~AnnualTemp, data = black.clean)
black.cold <- lm(log(AAM)~ColdTemp, data = black.clean)
black.warm <- lm(log(AAM)~WarmTemp, data = black.clean)

summary(black.annual)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp, data = black.clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.42489 -0.12464  0.00059  0.09959  0.30683
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)  1.984762   0.074361  26.691  < 2e-16 ***
## AnnualTemp  -0.017186   0.005344  -3.216  0.00433 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1754 on 20 degrees of freedom
## Multiple R-squared:  0.3409, Adjusted R-squared:  0.3079
## F-statistic: 10.34 on 1 and 20 DF,  p-value: 0.004333
```

```r
summary(black.cold)
```

```
##
## Call:
## lm(formula = log(AAM) ~ ColdTemp, data = black.clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.39468 -0.12079 -0.00699  0.08961  0.29562
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.767262   0.035603  49.638   <2e-16 ***
## ColdTemp    -0.011423   0.003084  -3.704   0.0014 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1664 on 20 degrees of freedom
## Multiple R-squared:  0.4069, Adjusted R-squared:  0.3772
## F-statistic: 13.72 on 1 and 20 DF,  p-value: 0.001405
```

```r
summary(black.warm)
```

```
##
## Call:
## lm(formula = log(AAM) ~ WarmTemp, data = black.clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.44602 -0.15359  0.03875  0.13845  0.30624
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.13752    0.26482   8.072 1.02e-07 ***
## WarmTemp    -0.01511    0.01098  -1.377    0.184
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2065 on 20 degrees of freedom
## Multiple R-squared:  0.08655,    Adjusted R-squared:  0.04087
## F-statistic: 1.895 on 1 and 20 DF,  p-value: 0.1839
```

```r
## Power analyses - annual
# calculate the coefficient of determination
coe.annual <- summary(black.annual)$adj.r.squared
pwr.f2.test(u = 1, v = 22 - 1 - 1, f2 = coe.annual/(1 -coe.annual),
            sig.level = 0.05)
```

```
##
##      Multiple regression power calculation
##
##               u = 1
##               v = 20
##              f2 = 0.4449434
##       sig.level = 0.05
##           power = 0.8450604
```

```r
## Power analyses - annual
# calculate the coefficient of determination
coe.cold <- summary(black.cold)$adj.r.squared
pwr.f2.test(u = 1, v = 22 - 1 - 1, f2 = coe.cold/(1 -coe.cold),
            sig.level = 0.05)
```

```
##
##      Multiple regression power calculation
##
##               u = 1
##               v = 20
##              f2 = 0.6056428
##       sig.level = 0.05
##           power = 0.9344838
```

```r
pwr.f2.test(u = 1, f2 = coe.cold/(1 -coe.cold),
            sig.level = 0.05, power = 0.8)
```

```
##
##      Multiple regression power calculation
##
##               u = 1
##               v = 13.14405
##              f2 = 0.6056428
##       sig.level = 0.05
##           power = 0.8
```

- We can see that annual temperature and cold temperature are significant predictors of black carp AAM. Warm temperature is not.
- Power analyses suggested that our current sample size is sufficient enought to produce a strong statistical power.

### Model selection using annual temperature - no subsample

```r
# Build the models
black.simple <- lm(log(AAM)~AnnualTemp, data = black.clean)
black.linear <- lm(log(AAM)~AnnualTemp+condition, data = black.clean)
black.int <- lm(log(AAM)~AnnualTemp:condition, data = black.clean)
black.group <- lm(log(AAM)~AnnualTemp*condition, data = black.clean)


## Compare the AICs
AIC(black.simple, black.linear, black.int, black.group)
```

```
##              df        AIC
## black.simple  3 -10.243418
## black.linear  4  -8.744915
```

```
## black.int     4  -9.987204
## black.group   5  -8.736108
```

```r
# Get a table of corrected AICs and their Akaike weights
models <- list(black.simple, black.linear, black.int, black.group)
mod.names <- c('simple linear', 'linear additive',
               'interaction', "grouped-specific")
aictab(cand.set = models, modnames = mod.names, sort = FALSE)
```

```
##
## Model selection based on AICc:
##
##                  K  AICc Delta_AICc AICcWt   LL
## simple linear    3 -8.91       0.00   0.51 8.12
## linear additive  4 -6.39       2.52   0.15 8.37
## interaction      4 -7.63       1.28   0.27 8.99
## grouped-specific 5 -4.99       3.92   0.07 9.37
```

```r
# R^2 value for the four models
r_2 <- data.frame(
  Model = c("Simple linear", "Linear additive", "Interaction", "Grouped"),
  R2 = c(summary(black.simple)$adj.r.squared,
         summary(black.linear)$adj.r.squared,
         summary(black.int)$adj.r.squared,
         summary(black.group)$adj.r.squared)
)
kable(r_2)
```

| Model | R2 |
|---|---|
| Simple linear | 0.3079314 |
| Linear additive | 0.2879251 |
| Interaction | 0.3270202 |
| Grouped | 0.3134071 |

```r
## Look at the summary (especially the slope for each model)
summary(black.simple)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp, data = black.clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.42489 -0.12464  0.00059  0.09959  0.30683
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.984762   0.074361  26.691  < 2e-16 ***
## AnnualTemp  -0.017186   0.005344  -3.216  0.00433 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1754 on 20 degrees of freedom
## Multiple R-squared:  0.3409, Adjusted R-squared:  0.3079
```

```
## F-statistic: 10.34 on 1 and 20 DF,  p-value: 0.004333
```

```r
summary(black.linear)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp + condition, data = black.clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.44968 -0.12574  0.02118  0.12338  0.28093
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2.007664   0.082986  24.193 9.76e-16 ***
## AnnualTemp       -0.016999   0.005428  -3.132  0.00549 **
## conditionnatural -0.050293   0.075985  -0.662  0.51600
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.178 on 19 degrees of freedom
## Multiple R-squared:  0.3557, Adjusted R-squared:  0.2879
## F-statistic: 5.246 on 2 and 19 DF,  p-value: 0.01535
```

```r
summary(black.int)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp:condition, data = black.clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.45878 -0.10360  0.01486  0.12414  0.25006
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     1.980132   0.073422  26.969  < 2e-16 ***
## AnnualTemp:conditionartificial -0.013372   0.006087  -2.197  0.04063 *
## AnnualTemp:conditionnatural    -0.020029   0.005738  -3.491  0.00245 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.173 on 19 degrees of freedom
## Multiple R-squared:  0.3911, Adjusted R-squared:  0.327
## F-statistic: 6.102 on 2 and 19 DF,  p-value: 0.008976
```

```r
summary(black.group)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp * condition, data = black.clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.43816 -0.06466 -0.00710  0.12129  0.24825
##
```

```
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)    
## (Intercept)                  1.921742   0.104737  18.348 4.24e-13 ***
## AnnualTemp                  -0.009633   0.007761  -1.241    0.230    
## conditionnatural             0.117098   0.148321   0.789    0.440    
## AnnualTemp:conditionnatural -0.013941   0.010676  -1.306    0.208    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.1747 on 18 degrees of freedom
## Multiple R-squared:  0.4115, Adjusted R-squared:  0.3134 
## F-statistic: 4.195 on 3 and 18 DF,  p-value: 0.02043
```

**Model selection using cold temperature - no subsample**

```
# Build the models
black.simple <- lm(log(AAM)~ColdTemp, data = black.clean)
black.linear <- lm(log(AAM)~ColdTemp+condition, data = black.clean)
black.int <- lm(log(AAM)~ColdTemp:condition, data = black.clean)
black.group <- lm(log(AAM)~ColdTemp*condition, data = black.clean)


# Compare the AICs
AIC(black.simple, black.linear, black.int, black.group)
```

```
##              df       AIC
## black.simple  3 -12.56341
## black.linear  4 -11.02060
## black.int     4 -12.29867
## black.group   5 -10.97660
```

```
# Get a table of corrected AICs and their Akaike weights
models <- list(black.simple, black.linear, black.int, black.group)
mod.names <- c('simple linear', 'linear additive',
               'interaction', "grouped-specific")
aictab(cand.set = models, modnames = mod.names, sort = FALSE)
```

```
## 
## Model selection based on AICc:
## 
##                  K   AICc Delta_AICc AICcWt    LL
## simple linear    3 -11.23       0.00   0.52  9.28
## linear additive  4  -8.67       2.56   0.14  9.51
## interaction      4  -9.95       1.28   0.27 10.15
## grouped-specific 5  -7.23       4.00   0.07 10.49
```

```
## R^2 value for the four models
r_2 <- data.frame(
  Model = c("Simple linear", "Linear additive", "Interaction", "Grouped"),
  R2 = c(summary(black.simple)$adj.r.squared,
         summary(black.linear)$adj.r.squared,
         summary(black.int)$adj.r.squared,
         summary(black.group)$adj.r.squared)
)
kable(r_2)
```

| Model | R2 |
|---|---|
| Simple linear | 0.3771965 |
| Linear additive | 0.3579008 |
| Interaction | 0.3941400 |
| Grouped | 0.3798876 |

```r
## Look at the summary (especially the slope for each model)
summary(black.simple)
```

```
##
## Call:
## lm(formula = log(AAM) ~ ColdTemp, data = black.clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.39468 -0.12079 -0.00699  0.08961  0.29562
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.767262   0.035603  49.638   <2e-16 ***
## ColdTemp    -0.011423   0.003084  -3.704   0.0014 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1664 on 20 degrees of freedom
## Multiple R-squared:  0.4069, Adjusted R-squared:  0.3772
## F-statistic: 13.72 on 1 and 20 DF,  p-value: 0.001405
```

```r
summary(black.linear)
```

```
##
## Call:
## lm(formula = log(AAM) ~ ColdTemp + condition, data = black.clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.41745 -0.10672  0.01471  0.11155  0.27214
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.790191   0.051231  34.944  < 2e-16 ***
## ColdTemp         -0.011293   0.003138  -3.598  0.00192 **
## conditionnatural -0.045613   0.072213  -0.632  0.53514
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.169 on 19 degrees of freedom
## Multiple R-squared:  0.4191, Adjusted R-squared:  0.3579
## F-statistic: 6.853 on 2 and 19 DF,  p-value: 0.005745
```

```r
summary(black.int)
```

```
##
## Call:
```

```
## lm(formula = log(AAM) ~ ColdTemp:condition, data = black.clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.39294 -0.09375 -0.00773  0.10732  0.27597
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  1.770280   0.035198  50.295  < 2e-16 ***
## ColdTemp:conditionartificial -0.007465   0.004394  -1.699  0.10564
## ColdTemp:conditionnatural    -0.015059   0.004211  -3.576  0.00201 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1642 on 19 degrees of freedom
## Multiple R-squared:  0.4518, Adjusted R-squared:  0.3941
## F-statistic: 7.831 on 2 and 19 DF,  p-value: 0.003308
```

```r
summary(black.group)
```

```
##
## Call:
## lm(formula = log(AAM) ~ ColdTemp * condition, data = black.clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.41954 -0.07945  0.00692  0.11033  0.24744
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)               1.797309   0.050646  35.488   <2e-16 ***
## ColdTemp                 -0.007107   0.004471  -1.590    0.129
## conditionnatural         -0.053456   0.071224  -0.751    0.463
## ColdTemp:conditionnatural -0.007989   0.006176  -1.294    0.212
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1661 on 18 degrees of freedom
## Multiple R-squared:  0.4685, Adjusted R-squared:  0.3799
## F-statistic: 5.288 on 3 and 18 DF,  p-value: 0.00861
```

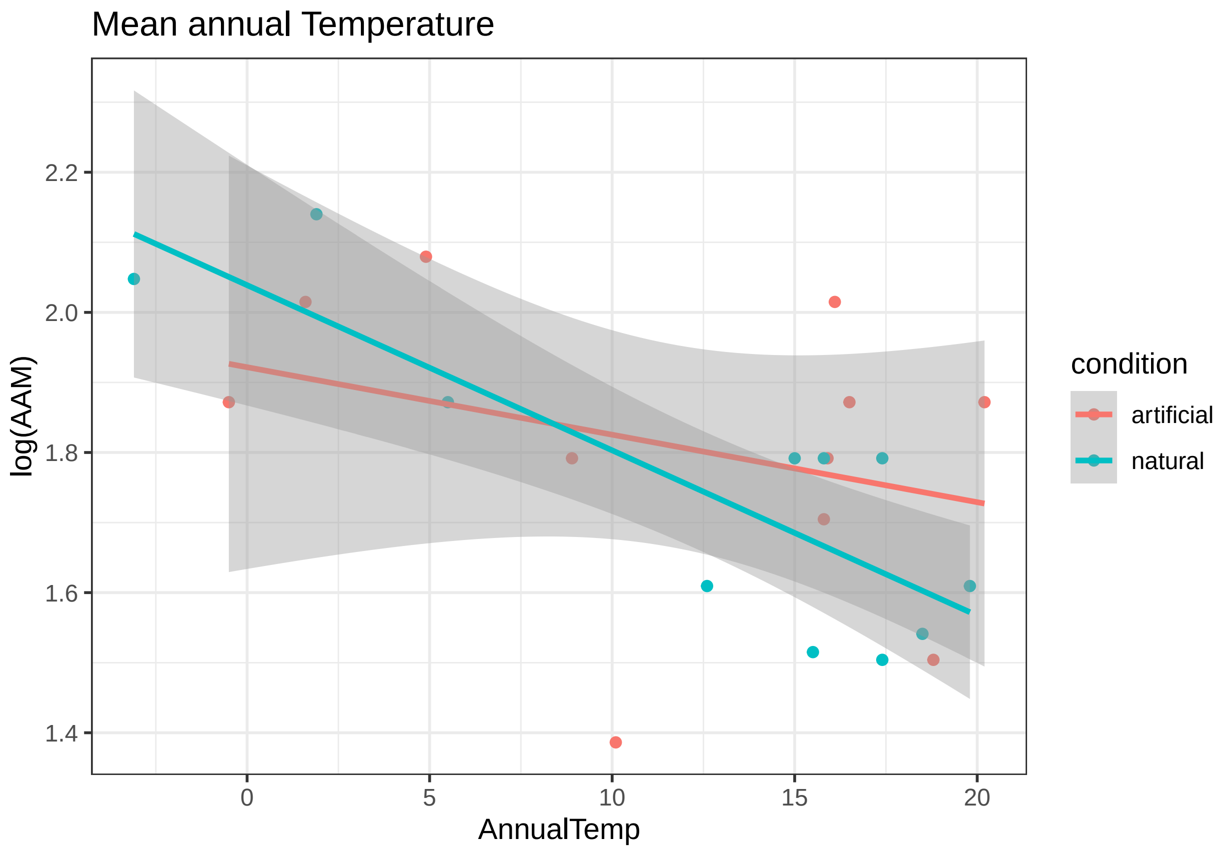
- No significant preference among the four models. Therefore, the simple linear model is selected.
- Cold temperature in general gives better predictions (lower AICc and higher R2).

### Black carp graphs with two conditions separated

We separated the black carp dataset into two based on conditions. Since there was no preference over the four models, we used the simple linear model on each set of the data.

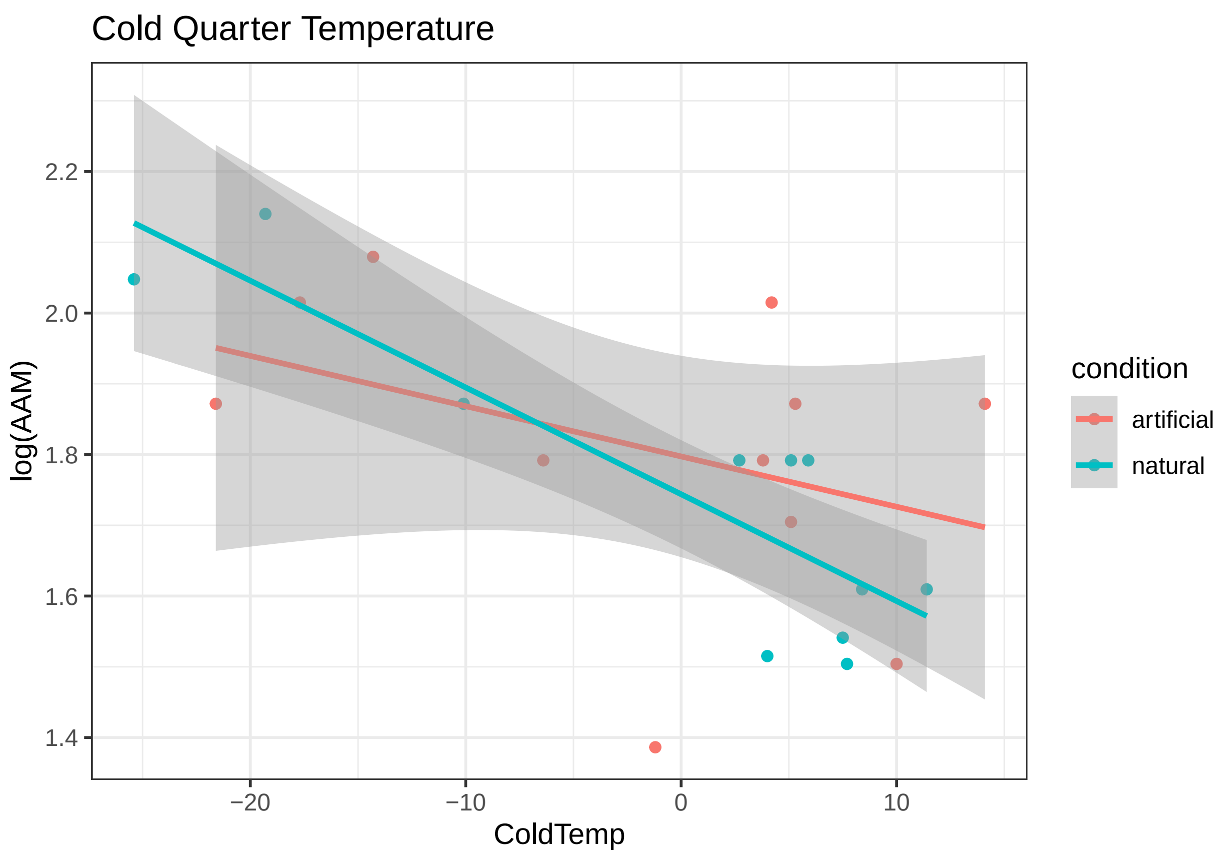
```r
## Annual temperature
ggplot(black.clean, aes(x = AnnualTemp, y = log(AAM), color = condition))+
  geom_point()+
  geom_smooth(method = "lm")+
  theme_bw()+
  labs(title = "Mean annual Temperature")
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```r
## Cold temperature
ggplot(black.clean, aes(x = ColdTemp, y = log(AAM), color = condition))+
  geom_point()+
  geom_smooth(method = "lm")+
  theme_bw()+
  labs(title = "Cold Quarter Temperature")
```

```
## `geom_smooth()` using formula 'y ~ x'
```




Now that we have seen that the artificial condition data seems to have a larger spread, we would like to run the simple linear model to take a look.

```r
## Separate into two data sets
black.natural <- black.clean[black.clean$condition == "natural",]
black.artificial <- black.clean[black.clean$condition == "artificial",]


## Run the models
black.annual.n <- lm(log(AAM)~AnnualTemp, data = black.natural)
black.cold.n <- lm(log(AAM)~ColdTemp, data = black.natural)

black.annual.a <- lm(log(AAM)~AnnualTemp, data = black.artificial)
black.cold.a <- lm(log(AAM)~ColdTemp, data = black.artificial)


## Compare the AIC scores
AIC(black.annual.n, black.annual.a) #for annual temperature
```

```
##                df         AIC
## black.annual.n  3 -10.4922037
## black.annual.a  3   0.9200476
```

```r
AIC(black.cold.n, black.cold.a) #for cold temperature
```

```
##              df        AIC
## black.cold.n  3 -13.121053
## black.cold.a  3   0.288321
```

```r
## Compare the model parameters
summary(black.annual.n)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp, data = black.natural)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.15831 -0.09440 -0.03738  0.11596  0.16311
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.038839   0.075970  26.838  6.7e-10 ***
## AnnualTemp  -0.023574   0.005304  -4.445  0.00161 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1264 on 9 degrees of freedom
## Multiple R-squared:  0.687,  Adjusted R-squared:  0.6523
## F-statistic: 19.76 on 1 and 9 DF,  p-value: 0.001612
```

```r
summary(black.annual.a)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp, data = black.artificial)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.43816 -0.05978  0.02318  0.12682  0.24825
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.921742   0.127277  15.099 1.07e-07 ***
## AnnualTemp  -0.009633   0.009431  -1.021    0.334
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2124 on 9 degrees of freedom
## Multiple R-squared:  0.1039, Adjusted R-squared:  0.004303
## F-statistic: 1.043 on 1 and 9 DF,  p-value: 0.3337
```

```r
summary(black.cold.n)
```

```
##
## Call:
## lm(formula = log(AAM) ~ ColdTemp, data = black.natural)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.168342 -0.084535 -0.007609  0.096764  0.136973
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)  1.743853   0.033825  51.555 1.95e-12 ***
## ColdTemp    -0.015096   0.002878  -5.246 0.000531 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1122 on 9 degrees of freedom
## Multiple R-squared:  0.7536, Adjusted R-squared:  0.7262
## F-statistic: 27.52 on 1 and 9 DF,  p-value: 0.0005305
```

```
summary(black.cold.a)
```

```
##
## Call:
## lm(formula = log(AAM) ~ ColdTemp, data = black.artificial)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.41954 -0.06766  0.02146  0.14343  0.24744
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.797309   0.062927  28.562 3.85e-10 ***
## ColdTemp    -0.007107   0.005555  -1.279    0.233
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2063 on 9 degrees of freedom
## Multiple R-squared:  0.1539, Adjusted R-squared:  0.05987
## F-statistic: 1.637 on 1 and 9 DF,  p-value: 0.2328
```

- It turned put that after separating out the artificial condition, the model performed much better. While the artificial model alone did not even have a significant relationship.

## Asian carp

### Define the models and check the R2

```
## Look at the spatial codes for the current asian carp data
asian.carp.clean <- asian.carp %>%
  filter(Condition %in% c("natural", "artificial"))

table(asian.carp.clean$Code)
```

```
##
##  A AA AB AC AD AE AF AG AH AI AL AM AN AO AP  B  C  D  E  F  G  H  J  K  L  M
##  7  1  1  1  1  1  3  3  1  1  2  1  1  1  2  6  3  4  3  1  2  1  2  1  1  3
##  N  O  Q  S  X  Y  Z
##  1  3  1  3  1  1  3
```

```
## Subsampling for 1000 times (define all four models in one iteration!)
# Create the matrices to store the results
asian.linear.results <- matrix(NA,1000,4)
asian.add.results <- matrix(NA,1000,4)
asian.int.results <- matrix(NA,1000,4)
asian.group.results <- matrix(NA,1000,4)
```

```r
# Slopes for all models


# Iteration (put both annual and cold inside one iteration)
for(i in 1:1000){
  sub <- asian.carp.clean %>% group_by(Code) %>% sample_n(size=1)

  # annual
  reg.linear.annual <- lm(log(AAM)~AnnualTemp, data = sub)
  reg.add.annual <- lm(log(AAM)~AnnualTemp+Condition, data = sub)
  reg.int.annual <- lm(log(AAM)~AnnualTemp:Condition, data = sub)
  reg.group.annual <- lm(log(AAM)~AnnualTemp*Condition, data = sub)

  # AICs for annual
  asian.linear.results[i,1]<-as.numeric(AICc(reg.linear.annual))
  asian.add.results[i,1]<-as.numeric(AICc(reg.add.annual))
  asian.int.results[i,1]<-as.numeric(AICc(reg.int.annual))
  asian.group.results[i,1]<-as.numeric(AICc(reg.group.annual))

  # R2 for annual
  asian.linear.results[i,2]<-summary(reg.linear.annual)$adj.r.squared
  asian.add.results[i,2]<-summary(reg.add.annual)$adj.r.squared
  asian.int.results[i,2]<-summary(reg.int.annual)$adj.r.squared
  asian.group.results[i,2]<-summary(reg.group.annual)$adj.r.squared

  # cold
  reg.linear.cold <- lm(log(AAM)~ColdTemp, data = sub)
  reg.add.cold <- lm(log(AAM)~ColdTemp+Condition, data = sub)
  reg.int.cold <- lm(log(AAM)~ColdTemp:Condition, data = sub)
  reg.group.cold <- lm(log(AAM)~ColdTemp*Condition, data = sub)

  # AICs for cold
  asian.linear.results[i,3]<-as.numeric(AICc(reg.linear.cold))
  asian.add.results[i,3]<-as.numeric(AICc(reg.add.cold))
  asian.int.results[i,3]<-as.numeric(AICc(reg.int.cold))
  asian.group.results[i,3]<-as.numeric(AICc(reg.group.cold))

  # R2 for cold
  asian.linear.results[i,4]<-summary(reg.linear.cold)$adj.r.squared
  asian.add.results[i,4]<-summary(reg.add.cold)$adj.r.squared
  asian.int.results[i,4]<-summary(reg.int.cold)$adj.r.squared
  asian.group.results[i,4]<-summary(reg.group.cold)$adj.r.squared
}


## R^2 values for the four models
# annual
r2annual <- data.frame(
  Model = c("Simple linear", "Linear additive", "Interaction", "Grouped"),
  R2 = c(mean(unique(asian.linear.results[,2])),
         mean(unique(asian.add.results[,2])),
         mean(unique(asian.int.results[,2])),
         mean(unique(asian.group.results[,2])))
```

```r
)
kable(r2annual)
```

| Model | R2 |
|---|---|
| Simple linear | 0.5497954 |
| Linear additive | 0.5410123 |
| Interaction | 0.5503061 |
| Grouped | 0.5389925 |

```r
# cold
r2cold <- data.frame(
  Model = c("Simple linear", "Linear additive", "Interaction", "Grouped"),
  R2 = c(mean(unique(asian.linear.results[,4])),
         mean(unique(asian.add.results[,4])),
         mean(unique(asian.int.results[,4])),
         mean(unique(asian.group.results[,4])))
)
kable(r2cold)
```

| Model | R2 |
|---|---|
| Simple linear | 0.5313820 |
| Linear additive | 0.5336307 |
| Interaction | 0.5190048 |
| Grouped | 0.5268630 |

**Check the slopes for all Asian carp models**

```r
summary(reg.linear.annual)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.74857 -0.19597  0.05018  0.20283  0.50338
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.965710   0.086902  22.620  < 2e-16 ***
## AnnualTemp  -0.040620   0.006345  -6.402 3.93e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2842 on 31 degrees of freedom
## Multiple R-squared:  0.5694, Adjusted R-squared:  0.5555
## F-statistic: 40.99 on 1 and 31 DF,  p-value: 3.928e-07
```

```r
summary(reg.add.annual)
```

```
##
## Call:
```

```
## lm(formula = log(AAM) ~ AnnualTemp + Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.68582 -0.19451  0.06315  0.20283  0.45260
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2.035308   0.113592  17.918  < 2e-16 ***
## AnnualTemp       -0.042944   0.006805  -6.310 5.88e-07 ***
## Conditionnatural -0.102369   0.107373  -0.953    0.348
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2846 on 30 degrees of freedom
## Multiple R-squared:  0.582,  Adjusted R-squared:  0.5542
## F-statistic: 20.89 on 2 and 30 DF,  p-value: 2.075e-06
```

```
summary(reg.int.annual)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp:Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.64645 -0.18332  0.09536  0.16756  0.46919
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     1.992617   0.088820  22.434  < 2e-16 ***
## AnnualTemp:Conditionartificial -0.039721   0.006331  -6.274 6.50e-07 ***
## AnnualTemp:Conditionnatural    -0.050622   0.010209  -4.959 2.62e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2817 on 30 degrees of freedom
## Multiple R-squared:  0.5905, Adjusted R-squared:  0.5632
## F-statistic: 21.63 on 2 and 30 DF,  p-value: 1.528e-06
```

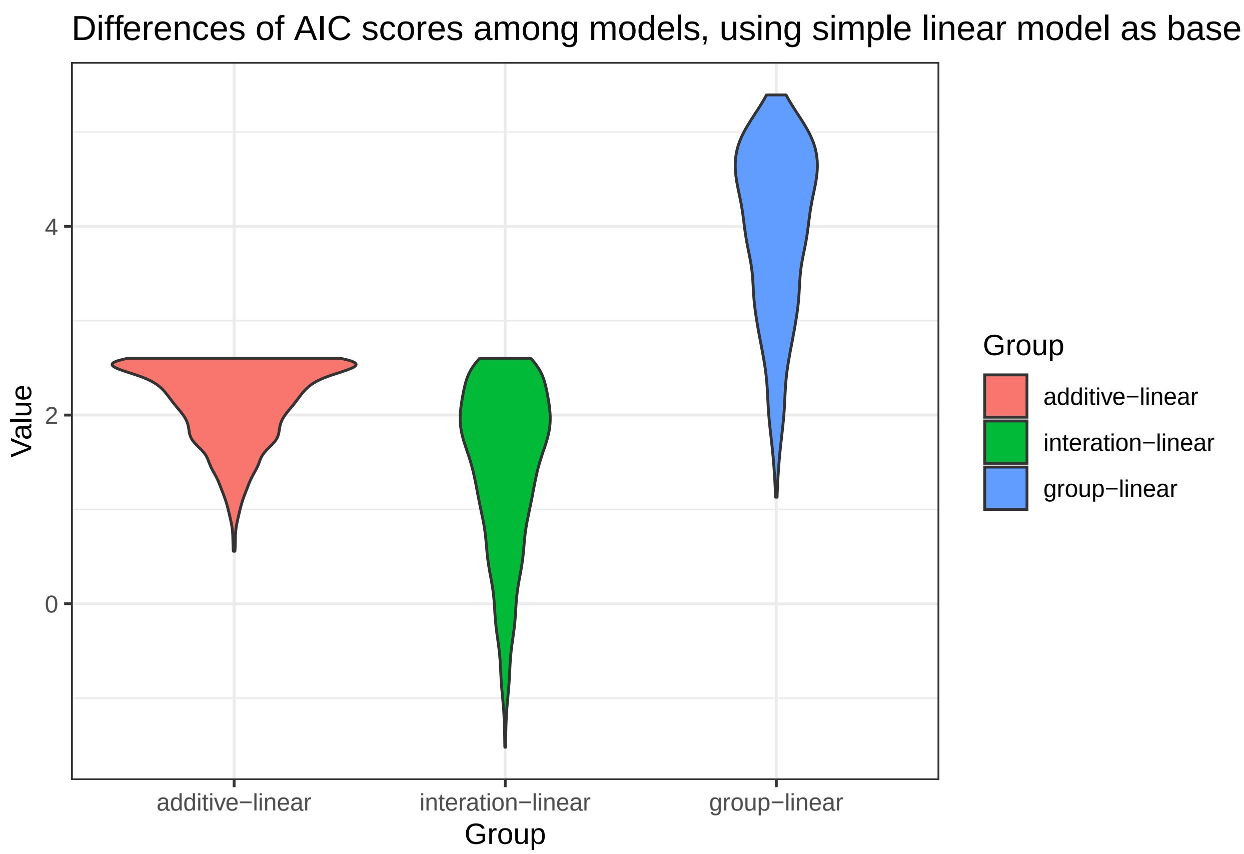
```
summary(reg.group.annual)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp * Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.64639 -0.18607  0.09639  0.16738  0.47208
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  1.987434   0.129943  15.295 2.03e-15 ***
## AnnualTemp                  -0.039438   0.008207  -4.805 4.35e-05 ***
## Conditionnatural             0.010030   0.180769   0.055    0.956
```

```
## AnnualTemp:Conditionnatural -0.011564   0.014906  -0.776    0.444
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2865 on 29 degrees of freedom
## Multiple R-squared:  0.5905, Adjusted R-squared:  0.5482
## F-statistic: 13.94 on 3 and 29 DF,  p-value: 8.248e-06
```

**Compare AICs for annual**

```
## Look at the distribution of the differences between AIC scores
# Calculate the differences of AIC values
aic.asian <- matrix(NA,1000,3) # store the differences in AIC values
aic.asian[,1] <- asian.add.results[,1] - asian.linear.results[,1]
aic.asian[,2] <- asian.int.results[,1] - asian.linear.results[,1]
aic.asian[,3] <- asian.group.results[,1] - asian.linear.results[,1]

# Create a data frame
data <- as.data.frame(aic.asian)
colnames(data) <- c("additive-linear","interation-linear","group-linear")

# Convert to long data format
data_long <- data %>%
  pivot_longer(cols = everything(), names_to = "Group", values_to = "Value")

# Define the desired order of groups
desired_order <- c("additive-linear","interation-linear","group-linear")

# Convert "Group" to a factor with desired order
data_long$Group <- factor(data_long$Group, levels = desired_order)

# Violin plot
ggplot(data_long, aes(x = Group, y = Value, fill = Group))+
  geom_violin()+
  labs(title = "Differences of AIC scores among models, using simple linear model as base. Annual Temp")
  theme_bw()
```




```r
## Check the AICc scores and akaike weights in 1000 iterations
weight.matrix <- matrix(NA, 1000, 4)
count <- numeric(0)

for (i in 1:1000) {
  # Create a list of the aic values of the current iteration
  aic_value <- c(asian.linear.results[i,1], asian.add.results[i,1],
                 asian.int.results[i,1], asian.group.results[i,1])

  ## check the akaike weights
  weight <- compute_akaike_weights(aic_value)
  weight.matrix[i,c(1,2,3,4)] <- round(weight[c(1,2,3,4)],3)

  ## check the AICc scores
  indexing <- compare_aic_scores(aic_value)
  if (indexing != -999) {
    count <- c(count, indexing)
  }
}

summary(weight.matrix)
```

```
##        V1               V2               V3               V4
##  Min.   :0.2270   Min.   :0.1190   Min.   :0.1560   Min.   :0.04200
##  1st Qu.:0.4537   1st Qu.:0.1590   1st Qu.:0.1980   1st Qu.:0.05400
##  Median :0.5275   Median :0.1690   Median :0.2310   Median :0.06700
```

```
##  Mean   :0.5061   Mean   :0.1712   Mean   :0.2498   Mean   :0.07297
##  3rd Qu.:0.5743   3rd Qu.:0.1820   3rd Qu.:0.2880   3rd Qu.:0.08600
##  Max.   :0.6200   Max.   :0.2450   Max.   :0.4860   Max.   :0.16800
```

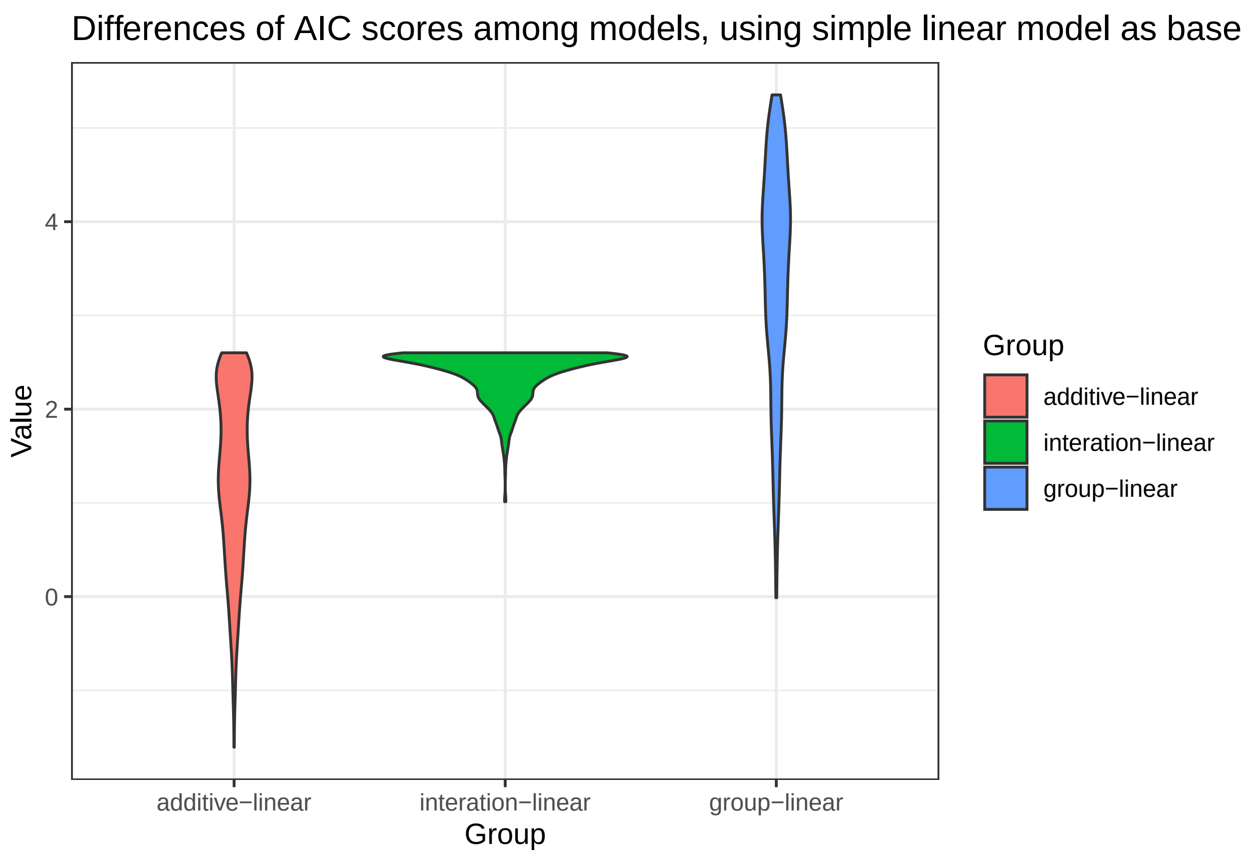
```
table(count)
```

```
## count
##   1
## 319
```

- When looking at each iteration, we saw that around 35% of the times the simple linear model is the best.
- In general, the linear model had the smallest AIC values.

### Compare AICs for the cold

```
# Calculate the differences of AIC values
aic.asian <- matrix(NA,1000,3) # store the differences in AIC values
aic.asian[,1] <- asian.add.results[,3] - asian.linear.results[,3]
aic.asian[,2] <- asian.int.results[,3] - asian.linear.results[,3]
aic.asian[,3] <- asian.group.results[,3] - asian.linear.results[,3]

# Look at the distribution of differences
# Create a data frame
data <- as.data.frame(aic.asian)
colnames(data) <- c("additive-linear","interation-linear","group-linear")

# Convert to long data format
data_long <- data %>%
  pivot_longer(cols = everything(), names_to = "Group", values_to = "Value")

# Define the desired order of groups
desired_order <- c("additive-linear","interation-linear","group-linear")

# Convert "Group" to a factor with desired order
data_long$Group <- factor(data_long$Group, levels = desired_order)

# Violin plot
ggplot(data_long, aes(x = Group, y = Value, fill = Group))+
  geom_violin()+
  labs(title = "Differences of AIC scores among models, using simple linear model as base. Cold Temp")+
  theme_bw()
```




```r
## Check the AICc scores and akaike weights in 1000 iterations
weight.matrix <- matrix(NA, 1000, 4)
count <- numeric(0)

for (i in 1:1000) {
  # Create a list of the aic values of the current iteration
  aic_value <- c(asian.linear.results[i,3], asian.add.results[i,3],
                 asian.int.results[i,3], asian.group.results[i,3])

  ## check the akaike weights
  weight <- compute_akaike_weights(aic_value)
  weight.matrix[i,c(1,2,3,4)] <- round(weight[c(1,2,3,4)],3)

  ## check the AICc scores
  indexing <- compare_aic_scores(aic_value)
  if (indexing != -999) {
    count <- c(count, indexing)
  }
}

summary(weight.matrix)
```

```
##        V1               V2               V3              V4
##  Min.   :0.2340   Min.   :0.1340   Min.   :0.064   Min.   :0.04300
##  1st Qu.:0.4410   1st Qu.:0.1928   1st Qu.:0.130   1st Qu.:0.06300
##  Median :0.5100   Median :0.2525   Median :0.151   Median :0.08150
```

```
##  Mean   :0.4941   Mean   :0.2599   Mean   :0.152   Mean   :0.09406
##  3rd Qu.:0.5583   3rd Qu.:0.3073   3rd Qu.:0.173   3rd Qu.:0.11200
##  Max.   :0.6180   Max.   :0.5230   Max.   :0.287   Max.   :0.27200
```

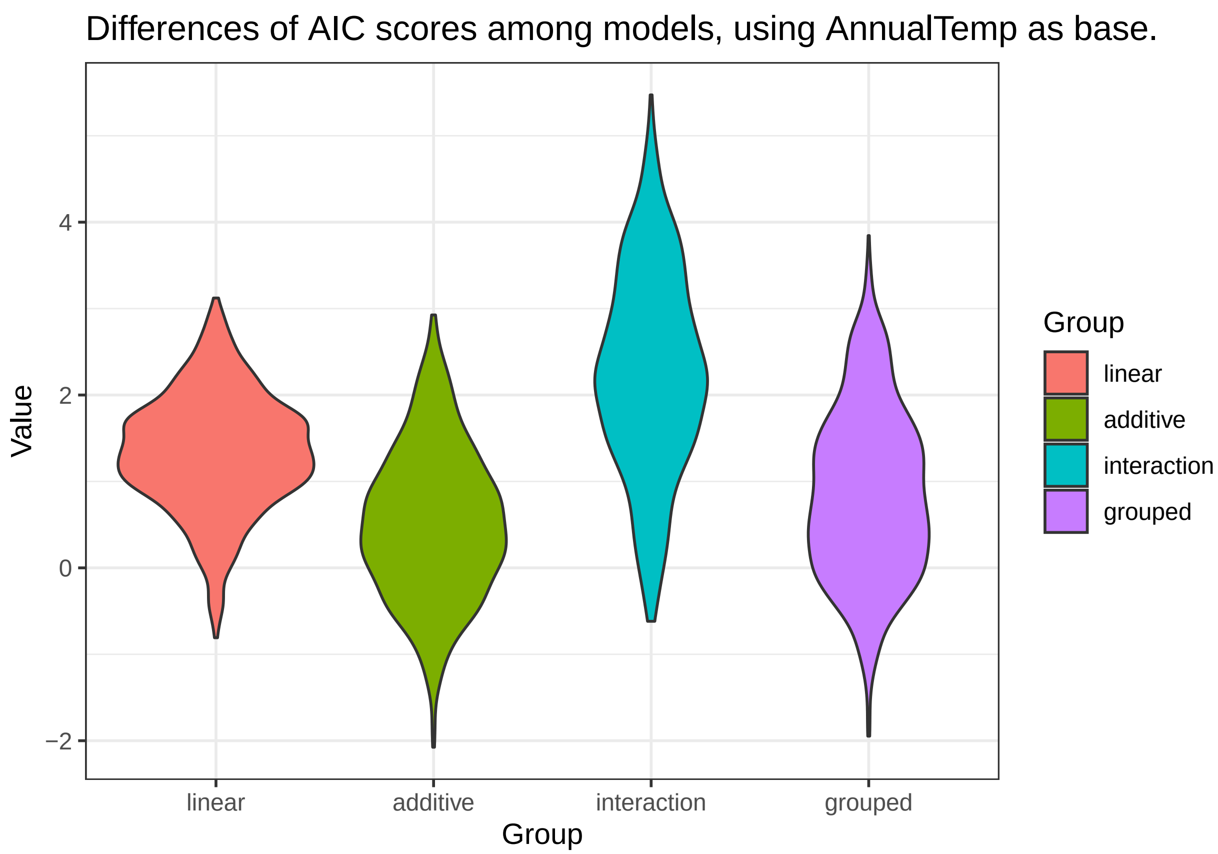
```r
table(count)
```

```
## count
##   1
## 243
```

- When looking at each iteration, we saw that around 28% of the times the simple linear model is the best.
- In general, the linear model had the smallest AIC values.

### Compare between annual and cold

```r
# Calculate the differences of AIC values
aic.asian <- matrix(NA,1000,4) # store the differences in AIC values
aic.asian[,1] <- asian.linear.results[,3] - asian.linear.results[,1]
aic.asian[,2] <- asian.add.results[,3] - asian.add.results[,1]
aic.asian[,3] <- asian.int.results[,3] - asian.int.results[,1]
aic.asian[,4] <- asian.group.results[,3] - asian.group.results[,1]

# Look at the distribution of differences
# Create a data frame
data <- as.data.frame(aic.asian)
colnames(data) <- c("linear","additive","interaction","grouped")

# Convert to long data format
data_long <- data %>%
  pivot_longer(cols = everything(), names_to = "Group", values_to = "Value")

# Define the desired order of groups
desired_order <- c("linear","additive","interaction","grouped")

# Convert "Group" to a factor with desired order
data_long$Group <- factor(data_long$Group, levels = desired_order)

# Violin plot
ggplot(data_long, aes(x = Group, y = Value, fill = Group))+
  geom_violin()+
  labs(title = "Differences of AIC scores among models, using AnnualTemp as base.")+
  theme_bw()
```




```
## Checking AIC values in each iteration
count <- NA

for (i in 1:1000) {
  # Create a list of the aic values of the current iteration
  aic_value <- c(asian.linear.results[i,1], asian.linear.results[i,3])
  smallest_aic <- min(aic_value)
  # Determining if the smallest value is 2 units smaller than the others
  is_smaller_by_two <- all(smallest_aic + 2 <= aic_value[aic_value != smallest_aic])

  # Append the index of the current list if smaller than 2 units
  if (is_smaller_by_two) {
    count <- c(count, which(aic_value == smallest_aic))
  }
}

count
```

```
##   [1] NA  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##  [26]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##  [51]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##  [76]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
## [101]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
## [126]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
## [151]  1  1  1  1  1  1  1  1  1  1  1  1  1
```

```r
length(count)-1
```

```
## [1] 162
```

- With the simple linear model, around 16% of the time when using Annualtemp is preferred over using ColdTemp.

## Grass carp

### Define and models and check the R2

```r
Grass.clean <- Grass %>%
  filter(Condition %in% c("natural", "artificial"))

table(Grass.clean$Code)
```

```
##
##   A AA AB AE AF AG AH AL AM AN AO AP  B  C  D  E  F  G  J  K  L  M  O  Q  S  X
##   3  1  1  1  1  1  1  1  1  1  1  1  3  2  2  1  1  2  1  1  1  1  1  1  1  1
##   Y  Z
##   1  1
```

```r
## Subsampling for 1000 times (define all four models in one iteration!)
# Create the matrices to store the results
grass.linear.results <- matrix(NA,1000,4)
grass.add.results <- matrix(NA,1000,4)
grass.int.results <- matrix(NA,1000,4)
grass.group.results <- matrix(NA,1000,4)

# Iteration (put both annual and cold inside one iteration)
for(i in 1:1000){
  sub <- Grass.clean %>% group_by(Code) %>% sample_n(size=1)

  # annual
  reg.linear.annual <- lm(log(AAM)~AnnualTemp, data = sub)
  reg.add.annual <- lm(log(AAM)~AnnualTemp+Condition, data = sub)
  reg.int.annual <- lm(log(AAM)~AnnualTemp:Condition, data = sub)
  reg.group.annual <- lm(log(AAM)~AnnualTemp*Condition, data = sub)

  # AICs for annual
  grass.linear.results[i,1]<-as.numeric(AICc(reg.linear.annual))
  grass.add.results[i,1]<-as.numeric(AICc(reg.add.annual))
  grass.int.results[i,1]<-as.numeric(AICc(reg.int.annual))
  grass.group.results[i,1]<-as.numeric(AICc(reg.group.annual))

  # R2 for annual
  grass.linear.results[i,2]<-summary(reg.linear.annual)$adj.r.squared
  grass.add.results[i,2]<-summary(reg.add.annual)$adj.r.squared
  grass.int.results[i,2]<-summary(reg.int.annual)$adj.r.squared
  grass.group.results[i,2]<-summary(reg.group.annual)$adj.r.squared

  # cold
  reg.linear.cold <- lm(log(AAM)~ColdTemp, data = sub)
  reg.add.cold <- lm(log(AAM)~ColdTemp+Condition, data = sub)
  reg.int.cold <- lm(log(AAM)~ColdTemp:Condition, data = sub)
```

```r
  reg.group.cold <- lm(log(AAM)~ColdTemp*Condition, data = sub)

  # AICs for cold
  grass.linear.results[i,3]<-as.numeric(AICc(reg.linear.cold))
  grass.add.results[i,3]<-as.numeric(AICc(reg.add.cold))
  grass.int.results[i,3]<-as.numeric(AICc(reg.int.cold))
  grass.group.results[i,3]<-as.numeric(AICc(reg.group.cold))

  # R2 for cold
  grass.linear.results[i,4]<-summary(reg.linear.cold)$adj.r.squared
  grass.add.results[i,4]<-summary(reg.add.cold)$adj.r.squared
  grass.int.results[i,4]<-summary(reg.int.cold)$adj.r.squared
  grass.group.results[i,4]<-summary(reg.group.cold)$adj.r.squared
}


## R^2 values for the four models
# annual
r2annual <- data.frame(
  Model = c("Simple linear", "Linear additive", "Interaction", "Grouped"),
  R2 = c(mean(unique(grass.linear.results[,2])),
         mean(unique(grass.add.results[,2])),
         mean(unique(grass.int.results[,2])),
         mean(unique(grass.group.results[,2])))
)
kable(r2annual)
```

| Model | R2 |
|---|---|
| Simple linear | 0.6346557 |
| Linear additive | 0.6299637 |
| Interaction | 0.6274822 |
| Grouped | 0.6156102 |

```r
# cold
r2cold <- data.frame(
  Model = c("Simple linear", "Linear additive", "Interaction", "Grouped"),
  R2 = c(mean(unique(grass.linear.results[,4])),
         mean(unique(grass.add.results[,4])),
         mean(unique(grass.int.results[,4])),
         mean(unique(grass.group.results[,4])))
)
kable(r2cold)
```

| Model | R2 |
|---|---|
| Simple linear | 0.6399651 |
| Linear additive | 0.6283568 |
| Interaction | 0.6277322 |
| Grouped | 0.6137069 |

### Check the slopes for all Grass carp models

```r
summary(reg.linear.annual)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.59965 -0.18402  0.03802  0.18490  0.48882
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.047990   0.090701  22.580  < 2e-16 ***
## AnnualTemp  -0.041613   0.006525  -6.377 9.37e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2873 on 26 degrees of freedom
## Multiple R-squared:   0.61,  Adjusted R-squared:  0.595
## F-statistic: 40.67 on 1 and 26 DF,  p-value: 9.366e-07
```

```r
summary(reg.add.annual)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp + Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.53479 -0.17433  0.04409  0.13403  0.52290
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.972581   0.117169  16.835 3.76e-15 ***
## AnnualTemp       -0.039367   0.006886  -5.717 5.91e-06 ***
## Conditionnatural  0.117585   0.115766   1.016    0.319
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2871 on 25 degrees of freedom
## Multiple R-squared:  0.6255, Adjusted R-squared:  0.5955
## F-statistic: 20.88 on 2 and 25 DF,  p-value: 4.659e-06
```

```r
summary(reg.int.annual)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp:Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.57866 -0.15055  0.03687  0.15563  0.47980
##
## Coefficients:
```

```
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     2.034406   0.091690  22.188  < 2e-16 ***
## AnnualTemp:Conditionartificial -0.043188   0.006710  -6.436  9.7e-07 ***
## AnnualTemp:Conditionnatural    -0.034166   0.009878  -3.459  0.00196 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2872 on 25 degrees of freedom
## Multiple R-squared:  0.6251, Adjusted R-squared:  0.5952
## F-statistic: 20.85 on 2 and 25 DF,  p-value: 4.712e-06
```

```
summary(reg.group.annual)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp * Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.55042 -0.15444  0.05241  0.13098  0.50763
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  1.996596   0.139183  14.345 2.86e-13 ***
## AnnualTemp                  -0.041152   0.008807  -4.673 9.55e-05 ***
## Conditionnatural             0.068689   0.187598   0.366    0.717
## AnnualTemp:Conditionnatural  0.004876   0.014554   0.335    0.741
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2923 on 24 degrees of freedom
## Multiple R-squared:  0.6272, Adjusted R-squared:  0.5806
## F-statistic: 13.46 on 3 and 24 DF,  p-value: 2.349e-05
```

```
summary(reg.linear.cold)
```

```
##
## Call:
## lm(formula = log(AAM) ~ ColdTemp, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.54843 -0.21522  0.04773  0.15672  0.55030
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.548930   0.053555  28.922  < 2e-16 ***
## ColdTemp    -0.025741   0.003916  -6.574  5.7e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2819 on 26 degrees of freedom
## Multiple R-squared:  0.6244, Adjusted R-squared:  0.6099
## F-statistic: 43.22 on 1 and 26 DF,  p-value: 5.697e-07
```

```r
summary(reg.add.cold)
```

```
##
## Call:
## lm(formula = log(AAM) ~ ColdTemp + Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.52150 -0.20735  0.06304  0.14808  0.57585
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.529097   0.073196  20.890  < 2e-16 ***
## ColdTemp         -0.025017   0.004362  -5.735 5.65e-06 ***
## Conditionnatural  0.048618   0.119945   0.405    0.689
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2866 on 25 degrees of freedom
## Multiple R-squared:  0.6268, Adjusted R-squared:  0.597
## F-statistic:    21 on 2 and 25 DF,  p-value: 4.457e-06
```

```r
summary(reg.int.cold)
```

```
##
## Call:
## lm(formula = log(AAM) ~ ColdTemp:Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.56397 -0.20953  0.04385  0.14638  0.53658
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  1.555077   0.059590  26.096  < 2e-16 ***
## ColdTemp:Conditionartificial -0.026700   0.005470  -4.881 5.07e-05 ***
## ColdTemp:Conditionnatural    -0.024468   0.006371  -3.841 0.000745 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2871 on 25 degrees of freedom
## Multiple R-squared:  0.6253, Adjusted R-squared:  0.5954
## F-statistic: 20.86 on 2 and 25 DF,  p-value: 4.682e-06
```

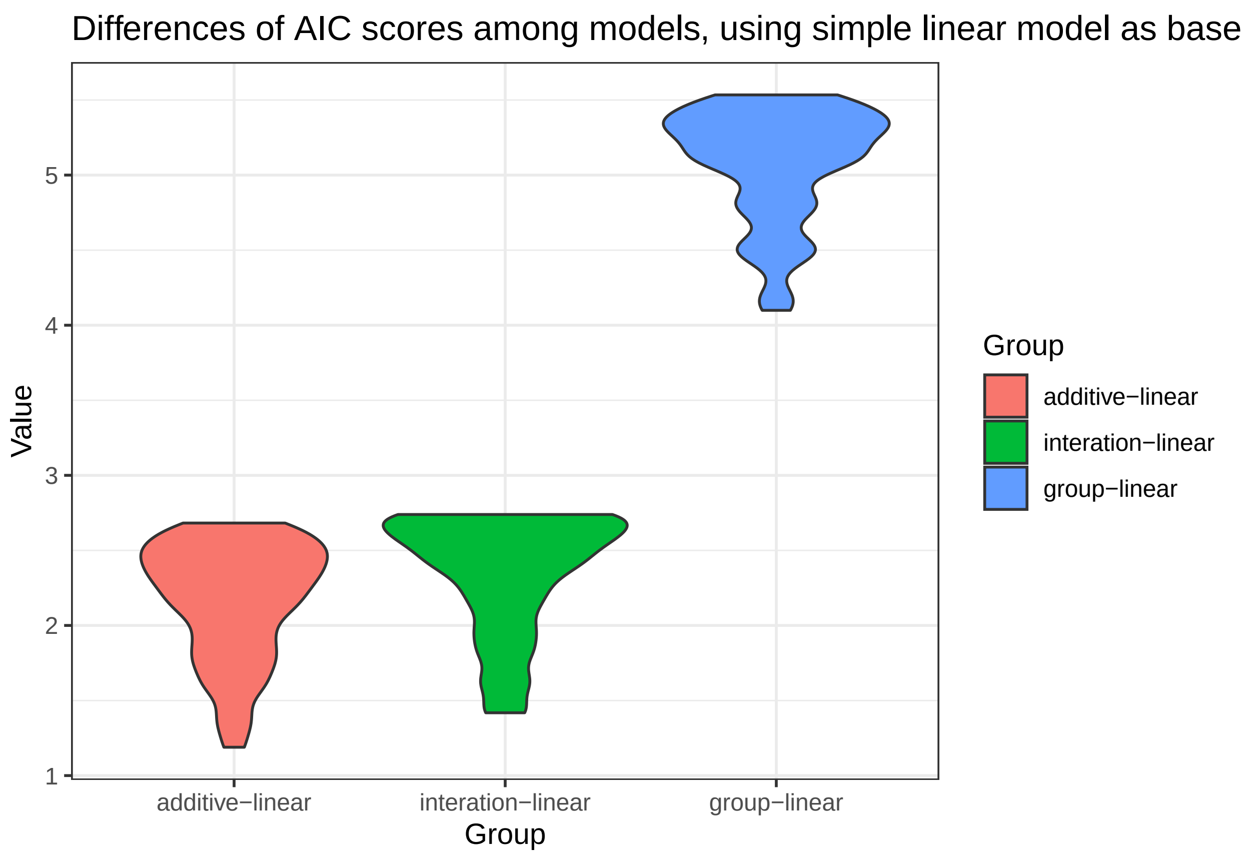
```r
summary(reg.group.cold)
```

```
##
## Call:
## lm(formula = log(AAM) ~ ColdTemp * Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.53831 -0.19326  0.06205  0.16387  0.56146
##
## Coefficients:
```

```
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                1.533462   0.075433  20.329  < 2e-16 ***
## ColdTemp                  -0.026287   0.005622  -4.675 9.48e-05 ***
## Conditionnatural           0.060694   0.126405   0.480    0.635
## ColdTemp:Conditionnatural  0.003373   0.009164   0.368    0.716
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2917 on 24 degrees of freedom
## Multiple R-squared:  0.6289, Adjusted R-squared:  0.5825
## F-statistic: 13.56 on 3 and 24 DF,  p-value: 2.228e-05
```

### Compare AICs for annual

```
# Calculate the differences of AIC values
aic.grass <- matrix(NA,1000,3) # store the differences in AIC values
aic.grass[,1] <- grass.add.results[,1] - grass.linear.results[,1]
aic.grass[,2] <- grass.int.results[,1] - grass.linear.results[,1]
aic.grass[,3] <- grass.group.results[,1] - grass.linear.results[,1]

# Look at the distribution of differences
# Create a data frame
data <- as.data.frame(aic.grass)
colnames(data) <- c("additive-linear","interation-linear","group-linear")

# Convert to long data format
data_long <- data %>%
  pivot_longer(cols = everything(), names_to = "Group", values_to = "Value")

# Define the desired order of groups
desired_order <- c("additive-linear","interation-linear","group-linear")

# Convert "Group" to a factor with desired order
data_long$Group <- factor(data_long$Group, levels = desired_order)

# Violin plot
ggplot(data_long, aes(x = Group, y = Value, fill = Group))+
  geom_violin()+
  labs(title = "Differences of AIC scores among models, using simple linear model as base. Annual Temp")
  theme_bw()
```




```
## Check the AICc scores and akaike weights in 1000 iterations
weight.matrix <- matrix(NA, 1000, 4)
count <- numeric(0)

for (i in 1:1000) {
  # Create a list of the aic values of the current iteration
  aic_value <- c(grass.linear.results[i,1], grass.add.results[i,1],
                 grass.int.results[i,1], grass.group.results[i,1])

  ## check the akaike weights
  weight <- compute_akaike_weights(aic_value)
  weight.matrix[i,c(1,2,3,4)] <- round(weight[c(1,2,3,4)],3)

  ## check the AICc scores
  indexing <- compare_aic_scores(aic_value)
  if (indexing != -999) {
    count <- c(count, indexing)
  }
}

summary(weight.matrix)
```

```
##        V1               V2               V3               V4
##  Min.   :0.4620   Min.   :0.1660   Min.   :0.1560   Min.   :0.03900
##  1st Qu.:0.5430   1st Qu.:0.1780   1st Qu.:0.1640   1st Qu.:0.04200
##  Median :0.5960   Median :0.1920   Median :0.1730   Median :0.04500
```

```
##  Mean   :0.5783   Mean   :0.1963   Mean   :0.1794   Mean   :0.04583
##  3rd Qu.:0.6160   3rd Qu.:0.2130   3rd Qu.:0.1880   3rd Qu.:0.04900
##  Max.   :0.6330   Max.   :0.2550   Max.   :0.2310   Max.   :0.05900
```

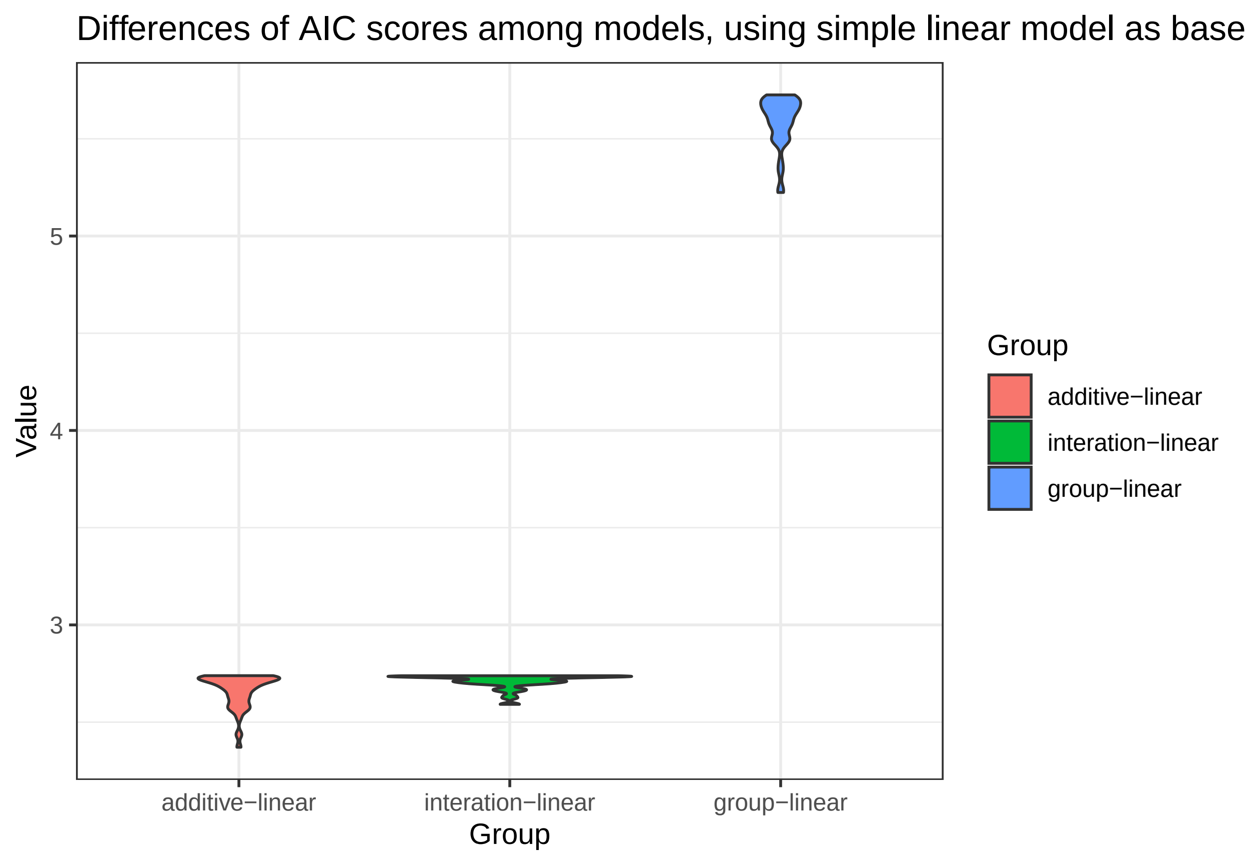
```r
table(count)
```

```
## count
##   1
## 689
```

- For grass carp, AIC for simple linear model was always smaller than the additive and interaction model, but within two units, and significantly smaller than the grouped-specific model (greater than 2 units).
- 68% of the times when the simple linear model perfroms better.

**Compare AICs for the cold**

```r
# Calculate the differences of AIC values
aic.grass <- matrix(NA,1000,3) # store the differences in AIC values
aic.grass[,1] <- grass.add.results[,3] - grass.linear.results[,3]
aic.grass[,2] <- grass.int.results[,3] - grass.linear.results[,3]
aic.grass[,3] <- grass.group.results[,3] - grass.linear.results[,3]

# Look at the distribution of differences
# Create a data frame
data <- as.data.frame(aic.grass)
colnames(data) <- c("additive-linear","interation-linear","group-linear")

# Convert to long data format
data_long <- data %>%
  pivot_longer(cols = everything(), names_to = "Group", values_to = "Value")

# Define the desired order of groups
desired_order <- c("additive-linear","interation-linear","group-linear")

# Convert "Group" to a factor with desired order
data_long$Group <- factor(data_long$Group, levels = desired_order)

# Violin plot
ggplot(data_long, aes(x = Group, y = Value, fill = Group))+
  geom_violin()+
  labs(title = "Differences of AIC scores among models, using simple linear model as base. Cold Temp")+
  theme_bw()
```




```r
## Check the AICc scores and akaike weights in 1000 iterations
weight.matrix <- matrix(NA, 1000, 4)
count <- numeric(0)

for (i in 1:1000) {
  # Create a list of the aic values of the current iteration
  aic_value <- c(grass.linear.results[i,3], grass.add.results[i,3],
                 grass.int.results[i,3], grass.group.results[i,3])

  ## check the akaike weights
  weight <- compute_akaike_weights(aic_value)
  weight.matrix[i,c(1,2,3,4)] <- round(weight[c(1,2,3,4)],3)

  ## check the AICc scores
  indexing <- compare_aic_scores(aic_value)
  if (indexing != -999) {
    count <- c(count, indexing)
  }
}

summary(weight.matrix)
```

```
##        V1               V2               V3               V4
##  Min.   :0.6110   Min.   :0.1600   Min.   :0.158   Min.   :0.0360
##  1st Qu.:0.6290   1st Qu.:0.1630   1st Qu.:0.162   1st Qu.:0.0370
##  Median :0.6340   Median :0.1650   Median :0.162   Median :0.0380
```

```
## Mean   :0.6315   Mean   :0.1667   Mean   :0.163   Mean   :0.0387
## 3rd Qu.:0.6360   3rd Qu.:0.1690   3rd Qu.:0.164   3rd Qu.:0.0400
## Max.   :0.6390   Max.   :0.1870   Max.   :0.170   Max.   :0.0450
```

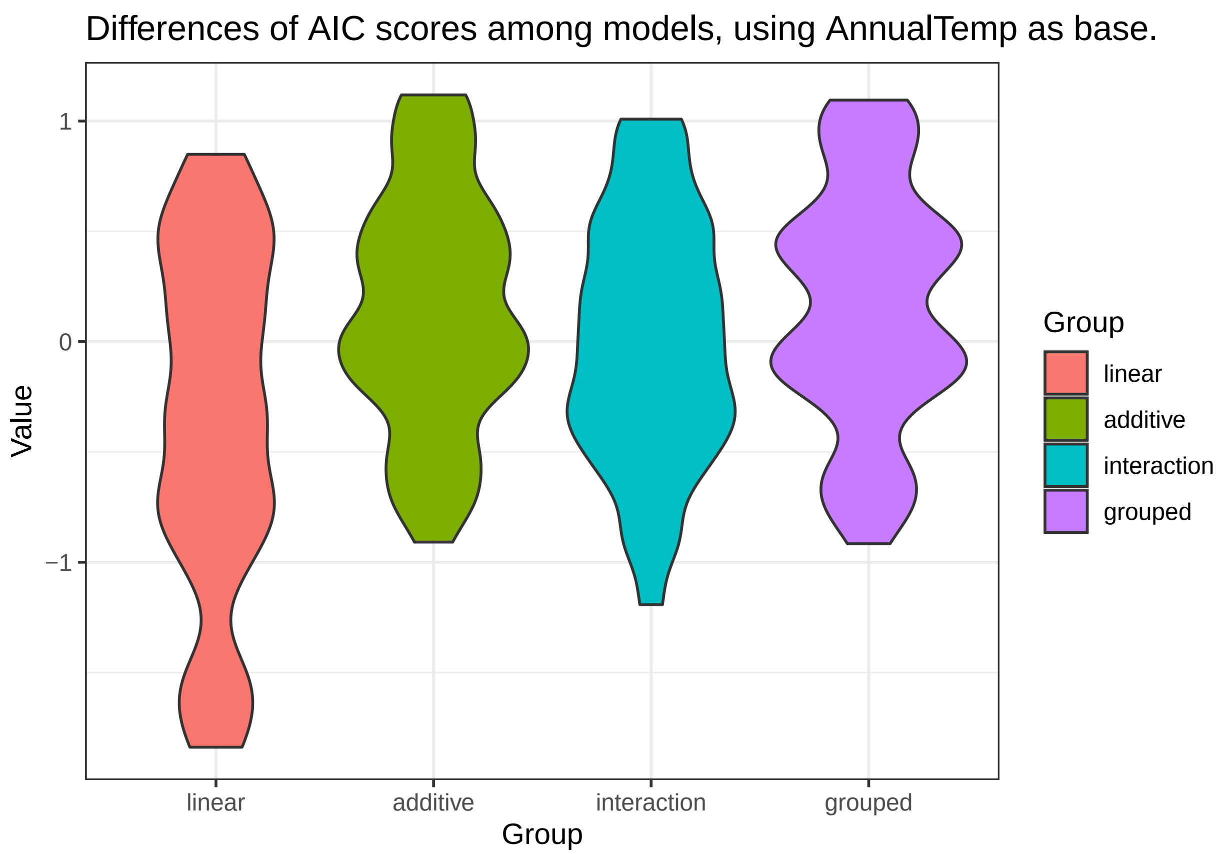
```
table(count)
```

```
## count
##    1
## 1000
```

- Same conclusion as AnnualTemp. 100% of the times when the simple model performs better.

#### Compare between annual and cold

```
# Calculate the differences of AIC values
aic.grass <- matrix(NA,1000,4) # store the differences in AIC values
aic.grass[,1] <- grass.linear.results[,3] - grass.linear.results[,1]
aic.grass[,2] <- grass.add.results[,3] - grass.add.results[,1]
aic.grass[,3] <- grass.int.results[,3] - grass.int.results[,1]
aic.grass[,4] <- grass.group.results[,3] - grass.group.results[,1]

# Look at the distribution of differences
# Create a data frame
data <- as.data.frame(aic.grass)
colnames(data) <- c("linear","additive","interaction","grouped")

# Convert to long data format
data_long <- data %>%
  pivot_longer(cols = everything(), names_to = "Group", values_to = "Value")

# Define the desired order of groups
desired_order <- c("linear","additive","interaction","grouped")

# Convert "Group" to a factor with desired order
data_long$Group <- factor(data_long$Group, levels = desired_order)

# Violin plot
ggplot(data_long, aes(x = Group, y = Value, fill = Group))+
  geom_violin()+
  labs(title = "Differences of AIC scores among models, using AnnualTemp as base.")+
  theme_bw()
```




```
## Checking AIC values in each iteration
count <- NA

for (i in 1:1000) {
  # Create a list of the aic values of the current iteration
  aic_value <- c(grass.linear.results[i,1], grass.linear.results[i,3])
  smallest_aic <- min(aic_value)
  # Determining if the smallest value is 2 units smaller than the others
  is_smaller_by_two <- all(smallest_aic + 2 <= aic_value[aic_value != smallest_aic])

  # Append the index of the current list if smaller than 2 units
  if (is_smaller_by_two) {
    count <- c(count, which(aic_value == smallest_aic))
  }
}
count
```

```
## [1] NA
```

```
summary(count)
```

```
##    Mode    NA's
## logical       1
```

- Cold temperature did not show any preference over Annual temperature. No significant differences.

## Bighead and silver carp

### Define the models and compare the R2

```
Big.sil.clean <- Big.sil %>%
  filter(Condition %in% c("natural", "artificial"))

table(Big.sil.clean$Code)
```

```
##
##  A AC AD AF AG AI AL AP  B  C  D  E  H  J  M  N  O  S  Z
##  4  1  1  2  2  1  1  1  3  1  2  2  1  1  2  1  2  2  2
```

```
## Subsampling for 1000 times (define all four models in one iteration!)
# Create the matrices to store the results
bs.linear.results <- matrix(NA,1000,4)
bs.add.results <- matrix(NA,1000,4)
bs.int.results <- matrix(NA,1000,4)
bs.group.results <- matrix(NA,1000,4)

# Iteration (put both annual and cold inside one iteration)
for(i in 1:1000){
  sub <- Big.sil.clean %>% group_by(Code) %>% sample_n(size=1)

  # annual
  reg.linear.annual <- lm(log(AAM)~AnnualTemp, data = sub)
  reg.add.annual <- lm(log(AAM)~AnnualTemp+Condition, data = sub)
  reg.int.annual <- lm(log(AAM)~AnnualTemp:Condition, data = sub)
  reg.group.annual <- lm(log(AAM)~AnnualTemp*Condition, data = sub)

  # AICs for annual
  bs.linear.results[i,1]<-as.numeric(AICc(reg.linear.annual))
  bs.add.results[i,1]<-as.numeric(AICc(reg.add.annual))
  bs.int.results[i,1]<-as.numeric(AICc(reg.int.annual))
  bs.group.results[i,1]<-as.numeric(AICc(reg.group.annual))

  # R2 for annual
  bs.linear.results[i,2]<-summary(reg.linear.annual)$adj.r.squared
  bs.add.results[i,2]<-summary(reg.add.annual)$adj.r.squared
  bs.int.results[i,2]<-summary(reg.int.annual)$adj.r.squared
  bs.group.results[i,2]<-summary(reg.group.annual)$adj.r.squared

  # cold
  reg.linear.cold <- lm(log(AAM)~ColdTemp, data = sub)
  reg.add.cold <- lm(log(AAM)~ColdTemp+Condition, data = sub)
  reg.int.cold <- lm(log(AAM)~ColdTemp:Condition, data = sub)
  reg.group.cold <- lm(log(AAM)~ColdTemp*Condition, data = sub)

  # AICs for cold
  bs.linear.results[i,3]<-as.numeric(AICc(reg.linear.cold))
  bs.add.results[i,3]<-as.numeric(AICc(reg.add.cold))
  bs.int.results[i,3]<-as.numeric(AICc(reg.int.cold))
  bs.group.results[i,3]<-as.numeric(AICc(reg.group.cold))

  # R2 for cold
```

```
  bs.linear.results[i,4]<-summary(reg.linear.cold)$adj.r.squared
  bs.add.results[i,4]<-summary(reg.add.cold)$adj.r.squared
  bs.int.results[i,4]<-summary(reg.int.cold)$adj.r.squared
  bs.group.results[i,4]<-summary(reg.group.cold)$adj.r.squared
}


## R^2 values for the four models
# annual
r2annual <- data.frame(
  Model = c("Simple linear", "Linear additive", "Interaction", "Grouped"),
  R2 = c(mean(unique(bs.linear.results[,2])),
         mean(unique(bs.add.results[,2])),
         mean(unique(bs.int.results[,2])),
         mean(unique(bs.group.results[,2])))
)
kable(r2annual)
```

| Model | R2 |
|---|---|
| Simple linear | 0.3378739 |
| Linear additive | 0.4199328 |
| Interaction | 0.3857888 |
| Grouped | 0.3841013 |

```
# cold
r2cold <- data.frame(
  Model = c("Simple linear", "Linear additive", "Interaction", "Grouped"),
  R2 = c(mean(unique(bs.linear.results[,4])),
         mean(unique(bs.add.results[,4])),
         mean(unique(bs.int.results[,4])),
         mean(unique(bs.group.results[,4])))
)
kable(r2cold)
```

| Model | R2 |
|---|---|
| Simple linear | 0.2306353 |
| Linear additive | 0.3125189 |
| Interaction | 0.1917873 |
| Grouped | 0.2684299 |

### Check the slopes for all bighead and silver carp models

```
summary(reg.linear.annual)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.75893 -0.14377 -0.01913  0.17074  0.56344
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.91498    0.14168  13.516  1.6e-10 ***
## AnnualTemp  -0.03588    0.01048  -3.424  0.00323 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3115 on 17 degrees of freedom
## Multiple R-squared:  0.4082, Adjusted R-squared:  0.3734
## F-statistic: 11.72 on 1 and 17 DF,  p-value: 0.003235
```

```
summary(reg.add.annual)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp + Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.62195 -0.18609  0.02824  0.13009  0.44765
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2.024288   0.141854  14.270 1.61e-10 ***
## AnnualTemp       -0.034562   0.009696  -3.565  0.00259 **
## Conditionnatural -0.263345   0.132420  -1.989  0.06412 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2875 on 16 degrees of freedom
## Multiple R-squared:  0.5255, Adjusted R-squared:  0.4661
## F-statistic: 8.859 on 2 and 16 DF,  p-value: 0.002571
```

```
summary(reg.int.annual)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp:Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.66218 -0.19242 -0.01966  0.11558  0.52482
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      1.92012    0.13751  13.963 2.23e-10 ***
## AnnualTemp:Conditionartificial -0.02905    0.01123  -2.588  0.01983 *
## AnnualTemp:Conditionnatural    -0.04378    0.01156  -3.787  0.00162 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3022 on 16 degrees of freedom
## Multiple R-squared:  0.4757, Adjusted R-squared:  0.4101
## F-statistic: 7.257 on 2 and 16 DF,  p-value: 0.005714
```

```r
summary(reg.group.annual)
```

```
##
## Call:
## lm(formula = log(AAM) ~ AnnualTemp * Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.62579 -0.18904  0.03627  0.12625  0.43764
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   2.064291   0.172292  11.981 4.42e-09 ***
## AnnualTemp                   -0.038124   0.012897  -2.956  0.00981 **
## Conditionnatural             -0.367125   0.274941  -1.335  0.20169
## AnnualTemp:Conditionnatural   0.008804   0.020275   0.434  0.67031
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2951 on 15 degrees of freedom
## Multiple R-squared:  0.5314, Adjusted R-squared:  0.4376
## F-statistic: 5.669 on 3 and 15 DF,  p-value: 0.008431
```

```r
summary(reg.linear.cold)
```

```
##
## Call:
## lm(formula = log(AAM) ~ ColdTemp, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.76574 -0.13647 -0.05495  0.11096  0.67209
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.471003   0.078173  18.817 8.08e-13 ***
## ColdTemp    -0.020189   0.007448  -2.711   0.0148 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3384 on 17 degrees of freedom
## Multiple R-squared:  0.3018, Adjusted R-squared:  0.2607
## F-statistic: 7.348 on 1 and 17 DF,  p-value: 0.01484
```

```r
summary(reg.add.cold)
```

```
##
## Call:
## lm(formula = log(AAM) ~ ColdTemp + Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.61644 -0.21597 -0.00317  0.16811  0.53950
##
## Coefficients:
```

```
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.60641    0.09934  16.171 2.46e-11 ***
## ColdTemp         -0.01983    0.00688  -2.883   0.0108 *
## Conditionnatural -0.28492    0.14361  -1.984   0.0647 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3124 on 16 degrees of freedom
## Multiple R-squared:  0.4396, Adjusted R-squared:  0.3696
## F-statistic: 6.277 on 2 and 16 DF,  p-value: 0.009721
```

```r
summary(reg.int.cold)
```

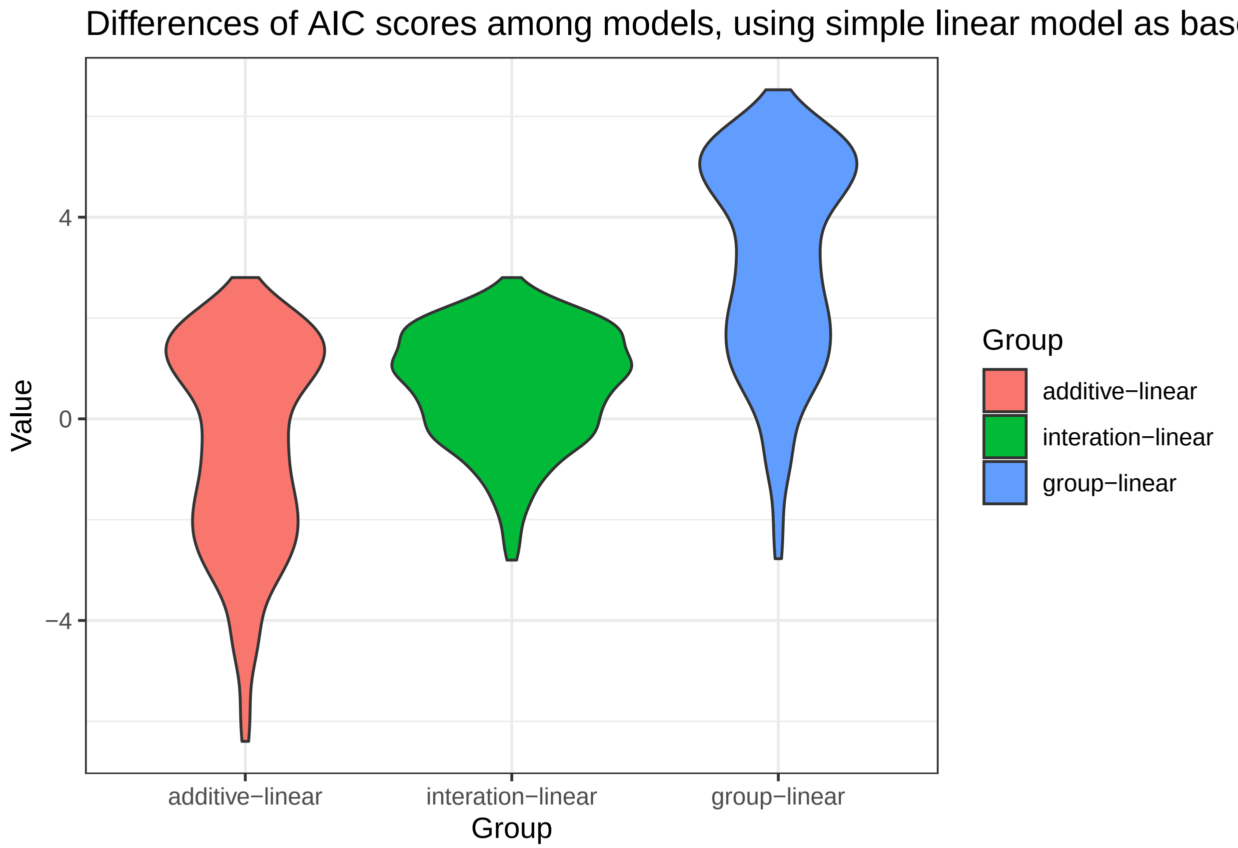
```
##
## Call:
## lm(formula = log(AAM) ~ ColdTemp:Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.76758 -0.15309 -0.02387  0.15727  0.64685
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  1.47032    0.07999  18.381  3.5e-12 ***
## ColdTemp:Conditionartificial -0.02347    0.01014  -2.315   0.0342 *
## ColdTemp:Conditionnatural    -0.01598    0.01147  -1.394   0.1825
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3462 on 16 degrees of freedom
## Multiple R-squared:  0.3121, Adjusted R-squared:  0.2262
## F-statistic:  3.63 on 2 and 16 DF,  p-value: 0.05012
```

```r
summary(reg.group.cold)
```

```
##
## Call:
## lm(formula = log(AAM) ~ ColdTemp * Condition, data = sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.62029 -0.22810  0.00946  0.16279  0.52707
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)               1.603494   0.102705  15.613  1.1e-10 ***
## ColdTemp                 -0.021775   0.009463  -2.301   0.0361 *
## Conditionnatural         -0.279643   0.148825  -1.879   0.0798 .
## ColdTemp:Conditionnatural  0.004416   0.014270   0.309   0.7612
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3217 on 15 degrees of freedom
## Multiple R-squared:  0.4432, Adjusted R-squared:  0.3318
## F-statistic:  3.98 on 3 and 15 DF,  p-value: 0.02858
```

**Compare AICs for annual**

```r
# Calculate the differences of AIC values
aic.bs <- matrix(NA,1000,3) # store the differences in AIC values
aic.bs[,1] <- bs.add.results[,1] - bs.linear.results[,1]
aic.bs[,2] <- bs.int.results[,1] - bs.linear.results[,1]
aic.bs[,3] <- bs.group.results[,1] - bs.linear.results[,1]

# Look at the distribution of differences
# Create a data frame
data <- as.data.frame(aic.bs)
colnames(data) <- c("additive-linear","interation-linear","group-linear")

# Convert to long data format
data_long <- data %>%
  pivot_longer(cols = everything(), names_to = "Group", values_to = "Value")

# Define the desired order of groups
desired_order <- c("additive-linear","interation-linear","group-linear")

# Convert "Group" to a factor with desired order
data_long$Group <- factor(data_long$Group, levels = desired_order)

# Violin plot
ggplot(data_long, aes(x = Group, y = Value, fill = Group))+
  geom_violin()+
  labs(title = "Differences of AIC scores among models, using simple linear model as base. Annual Temp")
  theme_bw()
```




```r
## Check the AICc scores and akaike weights in 1000 iterations
weight.matrix <- matrix(NA, 1000, 4)
count <- numeric(0)

for (i in 1:1000) {
  # Create a list of the aic values of the current iteration
  aic_value <- c(bs.linear.results[i,1], bs.add.results[i,1],
                 bs.int.results[i,1], bs.group.results[i,1])

  ## check the akaike weights
  weight <- compute_akaike_weights(aic_value)
  weight.matrix[i,c(1,2,3,4)] <- round(weight[c(1,2,3,4)],3)

  ## check the AICc scores
  indexing <- compare_aic_scores(aic_value)
  if (indexing != -999) {
    count <- c(count, indexing)
  }
}

summary(weight.matrix)
```

```
##        V1               V2               V3               V4
##  Min.   :0.0300   Min.   :0.1610   Min.   :0.1000   Min.   :0.02500
##  1st Qu.:0.1840   1st Qu.:0.2540   1st Qu.:0.1830   1st Qu.:0.04000
##  Median :0.3280   Median :0.3850   Median :0.2055   Median :0.06100
```

```
##  Mean   :0.3309   Mean   :0.3984   Mean   :0.2067   Mean   :0.06403
##  3rd Qu.:0.4770   3rd Qu.:0.5300   3rd Qu.:0.2290   3rd Qu.:0.08525
##  Max.   :0.6530   Max.   :0.7430   Max.   :0.3070   Max.   :0.12600
```

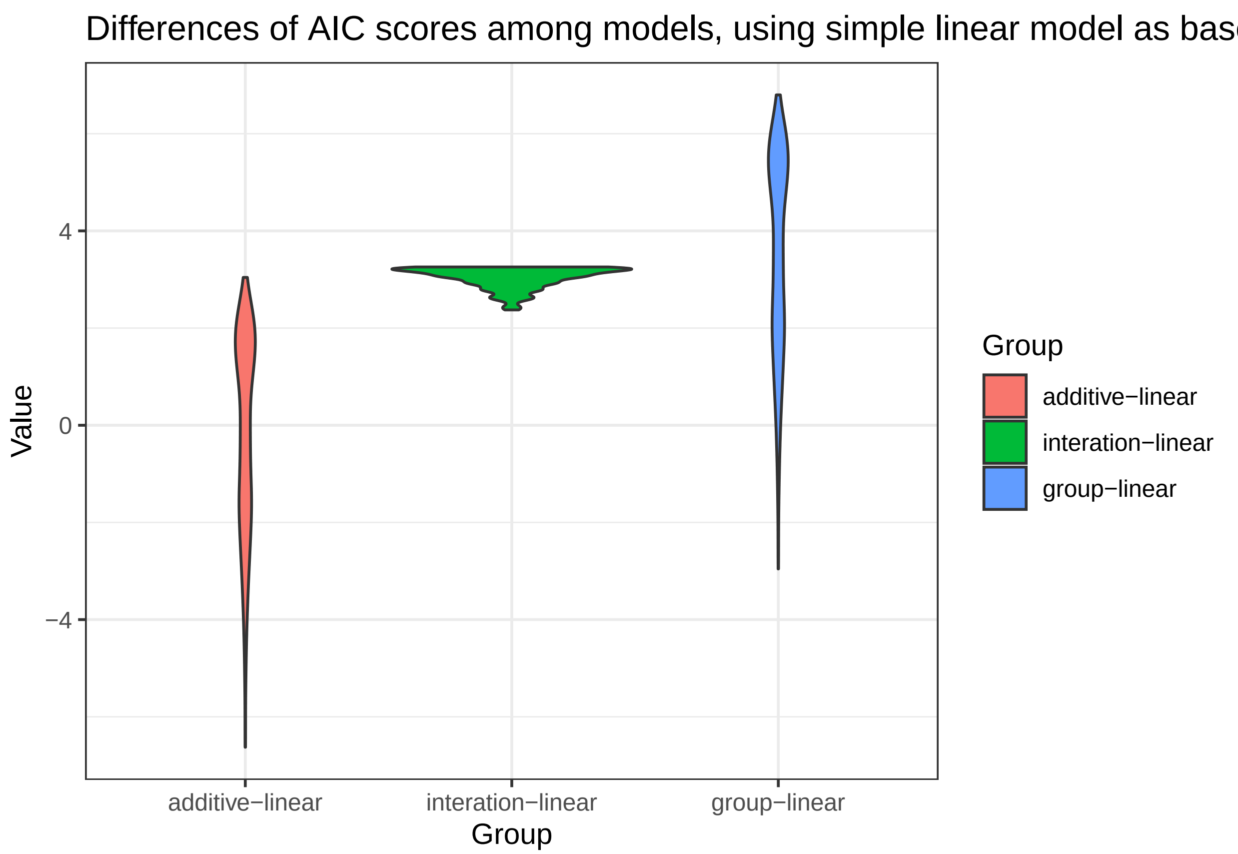
```r
table(count)
```

```
## count
##   1   2
##  61 208
```

- We saw a large range in the difference of AIC values due to a larger number of combinations for subsampling sets.
- 5.8% times simple model. 22% times linear additive model.

**Compare AICs for the cold**

```r
# Calculate the differences of AIC values
aic.bs <- matrix(NA,1000,3) # store the differences in AIC values
aic.bs[,1] <- bs.add.results[,3] - bs.linear.results[,3]
aic.bs[,2] <- bs.int.results[,3] - bs.linear.results[,3]
aic.bs[,3] <- bs.group.results[,3] - bs.linear.results[,3]

# Look at the distribution of differences
# Create a data frame
data <- as.data.frame(aic.bs)
colnames(data) <- c("additive-linear","interation-linear","group-linear")

# Convert to long data format
data_long <- data %>%
  pivot_longer(cols = everything(), names_to = "Group", values_to = "Value")

# Define the desired order of groups
desired_order <- c("additive-linear","interation-linear","group-linear")

# Convert "Group" to a factor with desired order
data_long$Group <- factor(data_long$Group, levels = desired_order)

# Violin plot
ggplot(data_long, aes(x = Group, y = Value, fill = Group))+
  geom_violin()+
  labs(title = "Differences of AIC scores among models, using simple linear model as base. Cold Temp")+
  theme_bw()
```




```r
## Check the AICc scores and akaike weights in 1000 iterations
weight.matrix <- matrix(NA, 1000, 4)
count <- numeric(0)

for (i in 1:1000) {
  # Create a list of the aic values of the current iteration
  aic_value <- c(bs.linear.results[i,3], bs.add.results[i,3],
                 bs.int.results[i,3], bs.group.results[i,3])

  ## check the akaike weights
  weight <- compute_akaike_weights(aic_value)
  weight.matrix[i,c(1,2,3,4)] <- round(weight[c(1,2,3,4)],3)

  ## check the AICc scores
  indexing <- compare_aic_scores(aic_value)
  if (indexing != -999) {
    count <- c(count, indexing)
  }
}

summary(weight.matrix)
```

```
##        V1               V2                V3                  V4
##  Min.   :0.030   Min.   :0.1510   Min.   :0.00700   Min.   :0.02300
##  1st Qu.:0.243   1st Qu.:0.2570   1st Qu.:0.05700   1st Qu.:0.04000
##  Median :0.421   Median :0.4135   Median :0.09500   Median :0.06500
```

```
##  Mean   :0.409   Mean   :0.4351   Mean   :0.08739   Mean   :0.06858
##  3rd Qu.:0.584   3rd Qu.:0.6010   3rd Qu.:0.12000   3rd Qu.:0.09500
##  Max.   :0.690   Max.   :0.8300   Max.   :0.14100   Max.   :0.13200
```

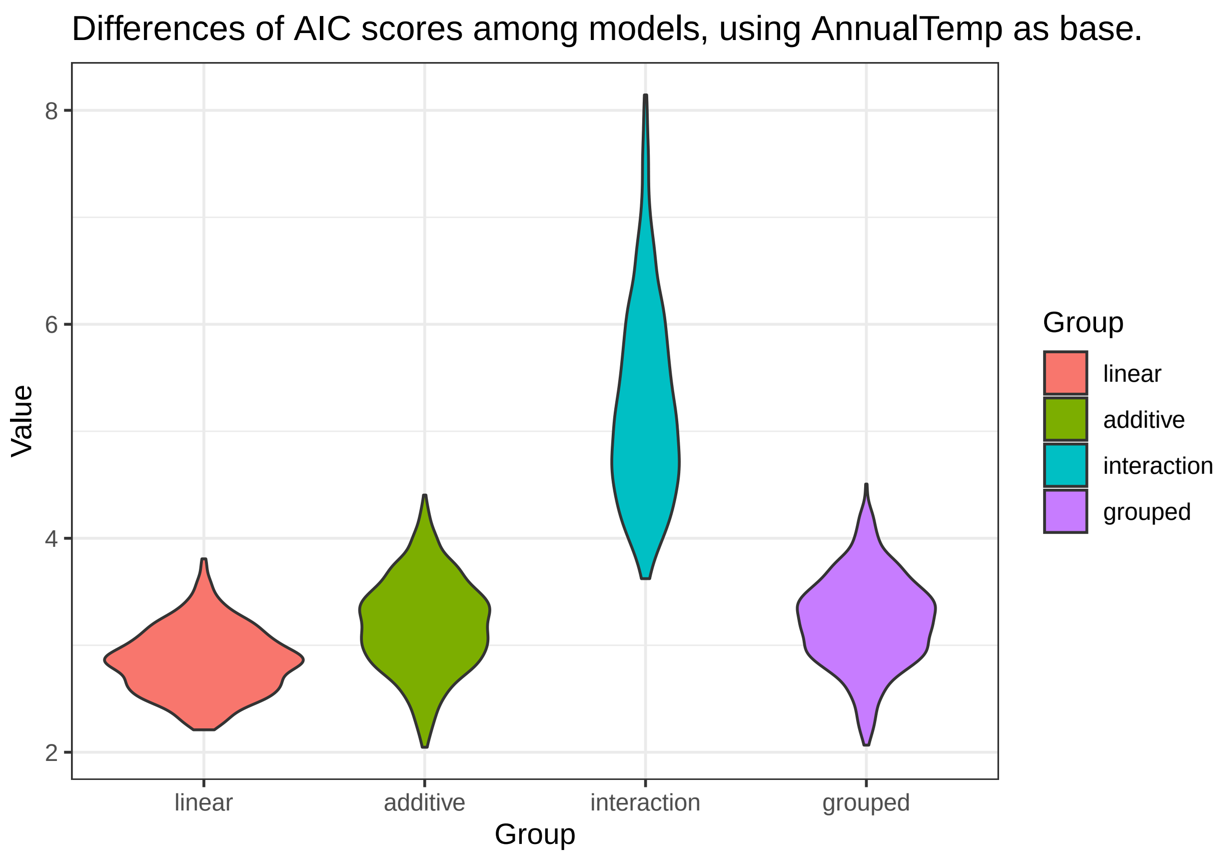
```r
table(count)
```

```
## count
##   1   2
## 149 217
```

- 47% simple linear model, 6.6% linear additive model.

**Compare between annual and cold**

```r
# Calculate the differences of AIC values
aic.bs <- matrix(NA,1000,4) # store the differences in AIC values
aic.bs[,1] <- bs.linear.results[,3] - bs.linear.results[,1]
aic.bs[,2] <- bs.add.results[,3] - bs.add.results[,1]
aic.bs[,3] <- bs.int.results[,3] - bs.int.results[,1]
aic.bs[,4] <- bs.group.results[,3] - bs.group.results[,1]

# Look at the distribution of differences
# Create a data frame
data <- as.data.frame(aic.bs)
colnames(data) <- c("linear","additive","interaction","grouped")

# Convert to long data format
data_long <- data %>%
  pivot_longer(cols = everything(), names_to = "Group", values_to = "Value")

# Define the desired order of groups
desired_order <- c("linear","additive","interaction","grouped")

# Convert "Group" to a factor with desired order
data_long$Group <- factor(data_long$Group, levels = desired_order)

# Violin plot
ggplot(data_long, aes(x = Group, y = Value, fill = Group))+
  geom_violin()+
  labs(title = "Differences of AIC scores among models, using AnnualTemp as base.")+
  theme_bw()
```




```r
## Checking AIC values in each iteration
count <- NA

for (i in 1:1000) {
  # Create a list of the aic values of the current iteration
  aic_value <- c(bs.linear.results[i,1], bs.linear.results[i,3])
  smallest_aic <- min(aic_value)
  # Determining if the smallest value is 2 units smaller than the others
  is_smaller_by_two <- all(smallest_aic + 2 <= aic_value[aic_value != smallest_aic])

  # Append the index of the current list if smaller than 2 units
  if (is_smaller_by_two) {
    count <- c(count, which(aic_value == smallest_aic))
  }
}

count
```

```
##    [1] NA  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##   [25]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##   [49]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##   [73]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##   [97]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##  [121]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##  [145]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##  [169]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##  [193]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
```

```
##  [217] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [241] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [265] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [289] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [313] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [337] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [361] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [385] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [409] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [433] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [457] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [481] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [505] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [529] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [553] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [577] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [601] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [625] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [649] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [673] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [697] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [721] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [745] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [769] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [793] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [817] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [841] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [865] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [889] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [913] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [937] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [961] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [985] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```r
table(count)
```

```
## count
##    1
## 1000
```

- For bs, using annual temperature is always better than using the cold temperature (difference in AIC > 2) for all models. This was also explained by lower R^2 values for the cold temperature models.
- For bighead and silver carp, there were fewer data points (32 datapoints in total), but more subsample sets (10 sets of subsamples. This gave us 19 data points after subsampling with a much larger variation (due to a larger number of combinations).
- At extremes, we would have 13 artificial and 6 natural (if all subseting choose artificial); or 10 natural and 9 artificial (if all subsetting choose natural).

## Concluding points

1. Black carp: using cold temperature have a better fit (higher R2). No preference over the four types of models.
2. Black carp: When separate the two conditions, we see a large increase in the R2 for the natural condition. The artificial condition alone did not have a significant relationship between log AAM and temperature.

3. Aisan carp: No preference over the four models.