Data science report:
Predictively modeling salmon louse counts in fish farming

DATA3800 – Introduction to data science with scripting

Project group 9 (individual submission)

Table of contents

1 Introduction

1.1 Biology of the salmon louse

The salmon louse (*Lepeophtheirus salmonis*) is a species of copepod which exists naturally in all ocean areas of the northern hemisphere (Håstein et al., 2022). It is an ectoparasite of several marine fish species, primarily salmonoid fishes, notably the Atlantic salmon (*Salmo salar*) (hereafter simply referred to as salmon), rainbow trout (*Oncorhynchus mykiss*), and sea trout (*Salmo trutta* morpha *trutta*) (Institute of Marine Research, 2020).

In its later, parasitic life stages, the salmon louse adheres to the skin or the gills of its host, where it feeds on mucus and blood, as well as skin and subcutaneous tissues (Håstein et al., 2022). Preadult and adult individuals cause the most serious harm, and in large numbers, infections can lead to emaciation, open wounds, and secondary fungal or bacterial infections. Untreated, the infection often results in the death of the fish host.

While salmon lice exhibit limited means of self propulsion, they are dispersed via ocean currents during their planktonic life stages, or by the movement of their hosts once attached. Individual salmon lice larvae may be transported several tens of kilometers away before reaching an infectious life stage (Asplin et al., 2014; Gillibrand & Willis, 2007; Johnsen et al., 2016).

Both in the industry and in the literature, the term *sea lice* is sometimes used to refer to salmon lice. While the salmon louse *is* a type of sea louse, the name sea lice is in fact a shared name for several parasitic species of the family Caligidae, primarily those within the genera *Lepeophtheirus* and *Caligus* (Revie et al., 2009).

Although the salmon louse is endemic to Norwegian coastal waters, the introduction of commercial salmon farming has led to a substantial increase in the number and density of suitable host populations (Institute of Marine Research, 2020). This, in turn, has resulted in an immense increase in the number of salmon lice, as well as changes to the spatial distribution of populations.

1.2 Concerns and controversies

The salmon louse issue is arguably *the* major challenge facing Norwegian aquaculture, as well as the raison d'être of a major public controversy surrounding the industry.

From the point of view of the fish farmer, high concentrations of salmon lice increases fish mortality and negatively affects salmon growth. In addition to lost profits, treatments and countermeasures against salmon lice infestations costs the Norwegian aquaculture industry an estimated 5 billion Norwegian kroner annually (Johansen, 2017).

Moreover, salmon lice outbreaks, as well as a general population increase due to intensive fish farming activity, constitutes a major environmental issue. At the end of 2022, 989 salt water farm sites were in operation, with an estimated live stock of 457 749 000 salmon, rainbow trout, and sea trout (Norwegian Directorate of Fisheries, 2022). Fish farming on such a massive scale has provided excellent conditions for parasite growth and transmission compared with natural conditions due to the unnaturally high abundance of suitable hosts (Torrissen et al., 2013). This can have an extremely detrimental effect on vulnerable wild salmonoid populations (Marty et al., 2010).

In recent years, the sea louse species *Caligus elongatus* has increasingly become an issue in salmon farming as well (Imsland et al., 2019). Unlike the salmon louse, *C. elongatus* is a generalist which can attack a broad range of species across several taxa, including e.g. salmonoids, gadiforms, herring, etc. (Hemmingsen et al., 2020). Because of this, farmed fish populations can both contract the parasites from and transmit them to a variety of wild fish populations. Another cause of concern is that *C. elongatus* is known to parasitise lumpfish (*Cyclopterus lumpus*) (Heuch et al., 2007), a fish species which is commonly used for biological control of sea lice in the fish farming industry.

For the time being, Norwegian aquaculture regulations concerning sea lice refer to salmon lice exclusively. This means that it is not mandatory for Norwegian fish farmers to count or report numbers of *C. Elongatus* (or other potential sea lice species of concern, e.g. *C. rogercresseyi*, which is the most important sea louse species affecting the Chilean fish farming industry), though it is not unlikely that individuals mistaken for salmon lice are included in lice counts. As a result, there is insufficient available data to include *C. Elongatus* in this analysis.

1.3 Counts and countermeasures

Every operative salmon, rainbow trout, and sea trout farming facility in Norway is required by Norwegian Law to register lice counts at specified intervals (Forskrift om lakselusbekjempelse, 2012). Lice counts are registered with The Norwegian Food Safety Authority through the Altinn platform and made publicly available through the ocean management information system BarentsWatch.

Lice counts are performed at weekly intervals when the sea temperature, measured at a depth of three meters, is 4°C or above, or at two-week intervals when the sea temperature is below 4°C. A random sample of individuals is taken from each fish cage, and the number of attached salmon lice are counted in three categories, namely adult female lice, mobile lice, and sessile (i.e. stationary) lice. The sample size varies by production area and time of year, with either ten or twenty fish included in the sample.

If the mean number of salmon lice per fish across every cage in the facility matches or exceeds the specified upper limit, countermeasures (e.g. mechanical delousing or medicinal treatment) must be implemented in order to bring the number of salmon lice down. Failure to comply may result in

forced slaughter of the entire stock. The upper limit is 0.2 or 0.5 adult female lice per fish respectively, determined by production area and time of year.

Broadly speaking, reactive countermeasures against salmon lice can be subdivided into three categories. These are *a)* mechanical removal, *b)* medicinal treatment, and *c)* biological control using cleanerfish. Each one of these categories is surrounded by its own host of controversies. Medicinal treatment has been proven especially controversial, as the chemical therapeutants can also affect non-target crustaceans (Bechmann et al., 2019). Objections to mechanical removal methods typically cite animal welfare concerns, as the methods utilised can be pretty harsh.

1.4 Project aim – So what?

The aim of the is to produce a model which is able to predict salmon louse counts based on easily available data. This may prove useful in developing new strategies of salmon louse control.

The way salmon lice are combatted today is inherently reactive in nature, i.e. countermeasures are implemented only after a legal threshold has been reached. This may result in overly intensive treatment regimens. By moving from a reactive to a proactive strategy for combatting salmon lice, we can hopefully limit the detrimental effects of both the salmon lice outbreaks themselves and the countermeasures by acting earlier.

2 Methods

2.1 Introducing the dataset

The dataset was retrieved from BarentsWatch. The dataset can be downloaded for free from their website, in its entirety or filtered by single localities and/or by start date and end date. The dataset is updated weekly as new registrations come in, with records going as far back as January 2012.

The dataset consists of three tables, but only two of the tables are relevant to this project. These are the tables concerning salmon lice counts and lice treatments. The third table, which is not relevant to the problem I aim to explore, contains records for farm sites with suspected or confimed status for notifiable fish diseases, these being Pancreatic Disease (PD) and Infectious Salmon Anemia (ISA).

More detailed information about the dataset can be found in an article published on the BarentsWatch website (2016).

2.2 Statistical analysis

2.2.1 Packages and dependencies

All diagrams and statistical anylyses are produced in the `R` programming environment, version 4.3.2 (R Core Team, 2023). The model was created using the function `lmer()` from the `lme4` package, version 1.1-35.1 (Bates et al., 2015). See the reference list for the full list of necessary `R` packages.

2.2.2 Choosing a model

My response variable is the average number of adult female salmon lice per salmon, calculated from a sample of ten or twenty fish per cage depending on the geographical region and time of year (see Section 1.3).

The data follow a nested hierarchical structure and are therefore not independent. Measurements are taken repeatedly from the same subjects (i.e. fish farms) over time, which means that the data are longitudinal. Additionally, individual farm sites are nested within municipalities, production areas, and counties. This means that there are several levels of nesting within the dataset.

We can expect that observations from the same farm site are more similar to each other than to observations from other sites, and that they are dependent on previous measurements from the same site. It is also plausible that observations are more similar within than between nested groups.

Because the data data are not independent, I will use a mixed model. This allows me to interpret predictor variables as either fixed or random effects (Henderson Jr, 1982). There is likely a lot of variation between individual farm sites which is not captured in the dataset, e.g. sea currents, wild fish and sea lice populations, production scale, etc. I account for this variation by including farm site as a random effect.

One assumption of a Linear Mixed Model (LMM) is that the residuals (i.e. the differences between observed and predicted values) are normally distributed. In an LMM, the residuals should also be homoscedastic, i.e. the variance of the residuals should remain constant as the predictors change. In this case, both of these assumptions are violated.

The distribution of are strongly right-skewed To avoid violating these assumptions, I can use a Generalised Linear Mixed Model (GLMM) instead. GLMMs are commonly used when the residuals are inherently non-normally distributed, e.g. when modelling count data.

2.2.3 Distribution family

Deciding on which family of probability distributions to use for my model was tricky. Two commonly used distribution families in the GLMM are the binomial or Poisson distributions. An assumption of these distribution families is that the variance is equal to the nominal mean. This is often not the case with biological or medical data. In contrast, these data tend to be overdispersed, i.e. the variance exceeds the mean (Ver Hoef & Boveng, 2007).

For overdispersed data, the Negative Binomial Distribution is useful, but this distribution family is typically reserved for count data. For continuous data with right-skewed distribution characteristics, either the Gamma distribution or the Inverse Gaussian Distribution could be a good fit, but both of these distribution families assume independence between observations.

Both the Negative Binomial Distribution, the Gamma distribution, and the Inverse Gaussian Distribution are defined for strictly positive (i.e. positive, non-zero) values. There are several ways of dealing with this (e.g. using a zero-inflated probability distribution), but the easiest method is to simply add a small constant (e.g. 0.01) to every value.

Another possible solution is to address the skewness of the data with a log transformation (used in combination with the added constant value, since $\log 0$ is undefined). For example, if a variable x follows a log-normal distribution, then $\ln x$ will follow a normal distribution. In such a case, we can fit an LMM despite the fact that the the residuals are not normally distributed. Alternatively, we can fit a GLMM with a log link function, but this still requires that the response variable follows a particular distribution (e.g. Negative Binomial) and adheres to the assumptions of that distribution family.

After exploring several of these options, I was not able to find a distribution family for the GLMM which accommodated my data, mainly due of the continuous nature of the response variable (i.e. an average) and the lack of independence between observations. Therefore, I decided on fitting an LMM to my data using a log transformation of the average number of adult female salmon lice as the response.

2.3 Model assessment

For a predictive model, the sensible measure of model quality is its ability to accurately predict the response variable. This means that, unlike in a descriptive model, we are not really interested in identifying which predictor variables are significant. The predictor variables which do not contribute to an accurate prediction will fall away during the model selection process.

I used three different measures to evaluate the candidate models. The Root Mean Squared Error (RMSE) and the Mean Absolute Error (MAE) were calculated using a set of functions from the `Metrics` package (Hamner & Frasco, 2018). The approximated coefficient of determination (R^2) was calculated using the

function `r.squaredGLMM()` from the `MuMIn` package (Bartoń, 2023).

RMSE and MAE are two common metrics used to measure the accuracy of continuous variables in regression analyses, and are interpreted in similar ways. Both methods are used to quantify the magnitude of the error made by the model in predicting the response (i.e. how far data points fall from the regression line), but in slightly different ways (i.e. absolute vs. average). I included both measures in the evaluation mostly for the sake of my own curiosity, but that is really not necessary.

R^2 is a common statistical measure to estimate the goodness-of-fit of a model, i.e. how well the model fits the data. Calculating the R^2 for an LMM is not as straightforward as for a traditional Linear Model (LM), since there is no universally agreed-upon measure of R^2 for mixed models. However, several approximations exist. The function used in this project is based on Nakagawa & Schielzeth (2013), and provides approximations of both the marginal R^2 (i.e. the variance explained by fixed effects only) and conditional R^2 (i.e. the variance explained by both fixed and random effects).

2.4 Predictor variables

2.4.1 Week number

The time of year is represented in the dataset by week number. However, week number is recorded as a linear variable ranging from 1 to 52. If week number were fitted as a predictor variable in a GLMM, the response would would increase/decrease linearly as a function of week number. In this case, the greatest difference in the response would be between week 1 and week 52.

In reality, the time of year is cyclic, i.e. week 52 and week 1 are neighboring values. To represent this, I created a predictor variable for cyclic time, represented by two oscillatory transformations of week number (Fernandez et al., 2014). The transformations were made using the single frequency cosine (cos 1) and sine (sin 1) functions

$$(\sin 1 \vee \cos 1) = (2 \times \pi \times \sin \vee \cos \times x) \div 52,$$

where x equals the frequency of the function (i.e. one, since one year represents one full cycle). Finally, I multiplied the sine and cosine values by -1 in order to flip them vertically (Fig. 1). This was done to make the relationship with the response variable more intuitive (i.e. positively correlated), since lice counts are typically higher in the warmer months.

Some years have 53 weeks instead of 52. This happens if there are at least four days left in the year after week 52. Since most years only have 52 weeks, the inclusion of week 53 will make week 52 and week 1 appear artificially further apart. To avoid this issue, and to avoid losing the pertinent data, I changed the week number to 52 for each of the affected rows.

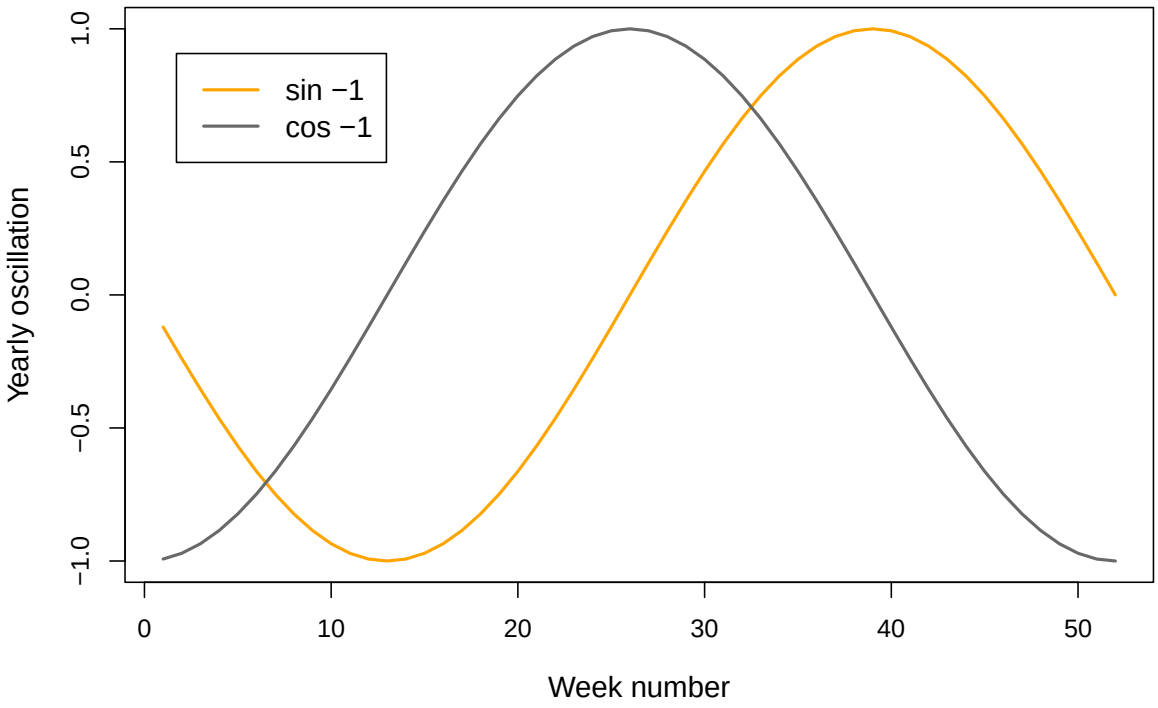

Figure 1: *Visualisation of the predictor variable for cyclic time. Yearly oscillation (cos −1) peaks mid summer. Yearly oscillation (sin −1) is identical to yearly oscillation (cos −1) but is offset by ¼ cycle.*

The year variable is not relevant for this project since we are not interested in modelling general trends.

2.4.2 Interaction effects

Sometimes, a response is better explained by the interaction between two predictor variables than by their additive effect. I have identified four possible interaction effcts in this dataset. These are:

1. *Temperature* × *Week number*
2. *Temperature* × *Latitude*
3. *Week number* × *Latitude*
4. *Medicinal treatment* × *Year*

The first three have to do with seasonality and climatic zones. Seasonal variations may depend on more than just temperature. It is therefore possible that an equal increase in temperature will have an unequal effect depending on these other factors. It is also possible that seasonal changes occur earlier or later at different latitudes.

The fourth hypothesised interaction effect is between medicinal treatment against salmon lice and year. Medicinal treatment has become somewhat less common over time due to the development of resistance in the salmon louse population (Coates et al., 2021). It is possible that the effect of treatment decreases over time.

2.4.3 Incomplete data

The records concerning cleaner fish were discontinued in 2018 due to changes in the surrounding legislation (Endr. i forskrifter om akvakultur – transporttilpasning, oppbevaring, bruk og produksjon av rensefisk, 2018). This unfortunately disqualifies the cleaner fish data from inclusion in the model.

3 Going through the code

> **Note**
>
> The Jupyter notebook contains some additional code used for data exploration and plot generation which is not repeated in the code walkthrough below.

3.1 Loading packages and sourcing data

I will start by loading the necessary `R` packages. Both `dplyr` and `tidyr` include several useful functions for data manipulation (Wickham, François, et al., 2023; Wickham, Vaughan, et al., 2023). `stringr` (Wickham, 2023) provides a cohesive set of functions for working with strings, which is especially useful for data cleaning tasks. I will also set a seed for the pseudorandom number generation. This will ensure the reproducibility of my exact results across sessions:

```
# 0. Load libraries and set seed: -----------------------------------------------------
library(dplyr)
library(tidyr)
library(stringr)
library(lme4)

# Set seed to ensure reproducibility:
set.seed(3800)
```

Additionally, I will sometimes utilize functions from a package without wanting to load the package in its entirety, e.g. when sourcing the data files (see next code chunk). This is achieved by using the following syntax: `package_name::function_name()`.

The next step is sourcing the data files:

```
# 1. Source data files ----------------------------------------------------------------
lice_per_fish  <- readr::read_csv("data/lakselus_per_fisk.csv")
lice_treatment <- readr::read_csv("data/tiltak_mot_lakselus.csv")
```

3.2 Data wrangling

Before we can use the data, it must first be manipulated into a usable working data frame. The first thing I do is to change the column names to conform to the `tidyverse` styleguide:

```
# Replace spaces with underscores in all column names and make lowercase:
list(lice_per_fish = lice_per_fish,
     lice_treatment = lice_treatment) %>%
  lapply(function(df) {
    rename_all(df, ~tolower(.)) %>%                # Change to lowercase.
    rename_all(.,  ~str_replace_all(., " ", "_"))   # Spaces to underscores.
  }) %>%
  list2env(envir = .GlobalEnv)  # Save to global environment.
```

In preparation for joining the two data frames together, I create a subset of the `lice_treatment` data frame to match the main data frame, i.e. `lice_per_fish`. I also split the column describing treatment into distinct boolean columns, in order to remove duplicate observations with respect to the triad of identifying variables (i.e. year, week number, and farm site):

```
# Prepare data frame 'lice_treatment' for joining:
lice_treatment <- lice_treatment %>%
  select(uke, år, lokalitetsnavn, tiltak) %>%

  # Remove duplicate rows:
  distinct() %>%

  # Give each of the treatment categories its own boolean column:
  mutate(value = TRUE) %>%
  pivot_wider(
    names_from = tiltak,  # Column to be split.
    values_from = value,
    values_fill = FALSE)
```

To create my working data frame, I join the two data frames `lice_per_fish` and `lice_treatment` together based on identical values in shared columns. Records without lice counts are discarded, and the cyclic time variables are added. I also perform several operations to clean up the data, e.g. changing the week number of week 53 (see Section 2.4.1) and filling in missing (i.e. `NA`) values:

```
# Join the dataframe 'lice_treatment' to 'lice_per_fish':
working_df <- left_join(lice_per_fish,
                        lice_treatment) %>%

  # Remove rows with no salmon lice counts:
  subset(har_telt_lakselus == "Ja") %>%

  # Fill NAs in the newly added boolean columns with FALSE:
  mutate_if(is.logical, coalesce, FALSE) %>%
```

```
  # Change week 53 to week 52 for all records:
  mutate(uke = if_else(uke == 53, 52, uke)) %>%

  # Add variables for cyclic time:
  {
    # Assign local variable 'angle':
    angle = (2 * pi * .$uke) / 52

    # Calculate flipped sine and cosine transformed values:
    mutate(., sin = -sin(angle),
              cos = -cos(angle))
  } %>%

  # Create the log transformed response variable:
  {
    const <- 0.01  # Add small constant to handle zeros.

    # NB! log() in R refers to the natural logarithm.
    mutate(., log_response = log(voksne_hunnlus + const))
  } %>%

  # Arrange rows chronologically:
  arrange(år, uke, lokalitetsnavn)
```

To back-transform the log transformed response variable later, use the command `exp(x) - 0.01`, where `x` is the column or the vector containing the fitted response.

Even after cleaning the dataset, some rows contain incomplete data. Only an insignificant portion of the total data was affected, or 248 rows in total. Still, before these rows were removed from the working data frame, I inspected the affected rows in order to avoid losing any usable data. The same nine columns were affected in each case, namely the columns containing geographic (i.e. unchanging) information about the site (e.g. longitude & latitude, county, site name, etc.). Luckily, site ID (i.e. `lokalitetsnummer`) was intact in each case. I was able to match the site ID and fill in the missing values from complete observations. After this procedure, only 167 rows which did not share their site ID with any known observations were discarded:

```
# Identify all rows containing NA values:
incomplete <- working_df[!complete.cases(working_df),]

# Fill NAs with correct values based on shared site IDs (i.e. 'lokalitetsnummer'):
working_df <- working_df %>%
  group_by(lokalitetsnummer) %>%

  {
    # Local variable containing the names of affected columns:
    incomplete_cols <- colnames(.)[colSums(is.na(.)) > 0]
```

```
    # Fill NA values where possible:
    mutate(., across(all_of(incomplete_cols), ~ first(na.omit(.))))
  } %>%
  ungroup()

# Remove rows which still contain NA values:
working_df <- working_df %>% na.omit()
```

3.3 Analysis

I began by splitting the dataset into a training set and a test set. The training set contains the largest subset of the data, and is used to fit the model. The test set is used to evaluate the model's performance and to assess its generalisation to new data:

```
# Split the data into test and training sets:
sample_index <- sample(seq_len(nrow(working_df)),
                       size = floor(0.8 * nrow(working_df)),  # 80 / 20 split.
                       replace = FALSE)

train <- working_df[ sample_index,]  # Include given rows.
test  <- working_df[-sample_index,]  # Exclude given rows.
```

This is the general syntax to fit a model with `lmer()`. The following syntax is used to specify `lokalitetsnavn` as a random effect `(1 | lokalitetsnavn)` specifies the random effect. In this example, and for the sake of illustration, I have included each of the considered fixed and interaction effects. That said, this does not necessarily represent the best model:

```
# Fit the Linear Mixed Model:
model <- lmer(log_response ~ sjøtemperatur +
                             lat +
                             cos + sin +
                             sjøtemperatur * cos +
                             sjøtemperatur * sin +
                             sjøtemperatur * lat +
                             cos * lat +
                             sin * lat +
                             (1 | lokalitetsnavn),
              data = train)
```

Next, I run the model on the test set and save the fitted response to a new column. I also save the back-transformed fitted response to a new column. This is not necessary to evaluate model performance, but it is a useful step to gain a grasp of the actual size of the difference between predicted and observed values (The back-transformed values are obviously needed for the final model as well, so that the predictions are readable and actionable):

```r
# Apply model to the test set:
test <- test %>%
  # Save the fitted response to a new column:
  mutate(fitted_log_response = predict(model,
                                       newdata = test,
                                       allow.new.levels = TRUE),

         # Back-transform the fitted response to avg. count:
         fitted_voksne_hunnlus = exp(fitted_log_response) - 0.01)
```

After running the model on the test set, I can evaluate the model performance using my chosen metrics:

```r
# Calculate metrics for model performance evaluation:
rmse_value <- Metrics::rmse(test$fitted_log_response,  # Root Mean Square Error.
                            test$log_response)
mae_value  <- Metrics::mae(test$fitted_log_response,   # Mean Absolute Error.
                           test$log_response)
r_squared  <- MuMIn::r.squaredGLMM(model)               # Approximated R squared.
```

Now that we have seen how this is done for one model, this step must be repeated for every candidate model (i.e. for each combination of fixed effects). It is not feasible to perform this manually given the number of possible fixed effects (including interaction effects).

To achieve this, I first generated a list of all possible combinations of predictor variables. This gave me a total of 255 possible models. In reality, this number is highly inflated, since any model which includes an interaction effect between a fixed effect A and a fixed effect B automatically includes both A and B *and* their interaction. This means that the list very likely contains several essentially duplicate combinations of fixed effects:

```r
# Define variable fixed effects:
# NB! Sea temperature is always included.
predictors <- c(
  "lat",
  "cos",
  "sin",
  "sjøtemperatur * lat",
  "sjøtemperatur * cos",
  "sjøtemperatur * sin",
  "cos * lat",
  "sin * lat"
)

# Generate combinations of predictors:
predictor_combinations <-
  lapply(1:length(predictors),
         function(n) {
           combinations <- gtools::combinations(length(predictors),
```

```
                                                n,
                                                as.character(predictors))
          lapply(1:nrow(combinations),
                 function(i) combinations[i, ])})

# Simplify the data structure:
predictor_combinations <- unlist(predictor_combinations,
                                 recursive = FALSE)
```

To avoid having to go through every model in the inflated list, I run the entire list through a function which finds and removes list entries which are themselves substrings of another entry in the same list. Once that is done, I simply remove every combination which is not unique. Using this method, 197 "possibilities" were discarded, and only 58 unique sets of fixed effects remained:

```
# Function to remove strings that are substrings of another string:
remove_substrings <- function(lst) {
  lst[!vapply(seq_along(lst),
              function(i) any(grepl(lst[i],
                                    lst[-i])),
              logical(1))]
  }

# Apply the function to the combinations of predictors:
predictor_combinations <- predictor_combinations %>%

  # Remove substrings:
  lapply(remove_substrings) %>%

  # Remove duplicate list entries:
  unique()
```

Now that I have compiled a list of possible fixed effects for my candidate models, I run through the list and save each model to the list `models`:

```
# Create a list to store model results:
models <- list()

# Iterate through the combinations list and fit models for each combination:
# Grab a cup of coffee, this will take a little while!
for (combination in predictor_combinations) {

  # Specify the formula based on the list of combinations:
  formula <- as.formula(paste("log_response ~ sjøtemperatur +
                                              (1 | lokalitetsnavn) +",
                              paste(combination, collapse = " + ")))

  # Run the model:
  model <- lmer(formula = formula, data = train)

```

```
  # Save the model to the list we created:
  models[[paste(c("sjøtemperatur",
                  combination), collapse = " + ")]] <- model
}
```

Finally, each of those models is run through the test set, and the evaluation metrics are saved to a data frame:

```
# Create the data frame to store the evaluation metrics:
model_evaluation <- data.frame(model = names(models),
                               rmse  = NA,
                               mae   = NA,
                               r2m   = NA,
                               r2c   = NA)

for (i in seq_along(models)) {

  # Run the model through the test set:
  prediction <- predict(models[[i]],
                        newdata = test,
                        allow.new.levels = TRUE)

  # Calculate RMSE:
  model_evaluation[i, "rmse"] <- Metrics::rmse(prediction,
                                               test$log_response)
  # Calculate MAE:
  model_evaluation[i, "mae"] <- Metrics::mae(prediction,
                                             test$log_response)
  # Calculate R^2:
  model_evaluation[i, c("r2m", "r2c")] <- MuMIn::r.squaredGLMM(models[[i]]) %>%
    as.vector()
}
```

4 Results

In order to identify the best model among the candidate models, we have to look at the evaluation metrics. Both RMSE and MAE measure an error, so for these two measures, we want the value to be as low as possible. For the marginal and conditional R^2. we want the value to be high (though this can be a pitfall, as R^2 does not punish for adding additional predictor variables, even if this does not help the model's predictive ability).

```
# Find best model according to all evaluation metrics:
model_evaluation %>%
  filter(rmse == min(rmse))  # Find minimum RMSE value.

model_evaluation %>%
  filter(mae == min(mae))    # Find minimum MAE value.
```

```r

model_evaluation %>%
  filter(r2m == max(r2m))    # Find maximum R^2m value.

model_evaluation %>%
  filter(r2c == max(r2c))    # Find maximum R^2c value.
```

All the evaluation metrics pointed to the same best model, given by the formula `log_response ~ + sjøtemperatur * lat + sjøtemperatur * cos + sjøtemperatur * sin + lat * cos + lat * sin + (1 | lokalitetsnavn)`. The difference between the best and worst model was small according to both the RMSE, MAE, and R^2 evaluation metrics (Table 1). The conditional R^2 was much higher than the marginal R^2. This indicates that the random effect helped the model accuracy of the model.

Table 1: Values from the best and worst model according to the RMSE, MAE, and R^2 evaluation metrics.

Evaluation metric	Best model	Worst model	Difference
Root Mean Square Error (RMSE)	2.2031	2.2437	0.0406
Mean Absolute Error (MAE)	1.8578	1.8994	0.0416
Marginal R^2	0.0397	0.0069	0.0328
Conditional R^2	0.1635	0.1249	0.0387

5 Discussion and conclusion

5.1 Bimodal distribution

The distribution of the residuals is bimodal (Figure 2). This is an objective violation of the model assumptions of an LMM, which assumes a normal distribution of residuals. However, this does not necessarily matter too much to accurate estimation, as LMMs can be remarkably robust (Knief & Forstmeier, 2021; Schielzeth et al., 2020; Warrington et al., 2014). Schielzeth et al. (2020) found that *"the effect of violations of distributional assumptions of random effect variances and residuals [was] surprisingly small"*, even when when the distributions were substantially skewed, bimodal, and heteroscedastic.

Bimodal distributions often occur when there are two distinct clusters in the data. For example, in biology, bimodal distributions often occur becuase of differences between sexes, i.e. one peak represents the distribution among the male cluster and the other peak among the female cluster.

In this case, there is clustering within the dataset which is not adequately accounted for by the model. I have not identified the cause of the clustering, but the bimodal distribution indicates that the situation is in some way more complex than assumed.

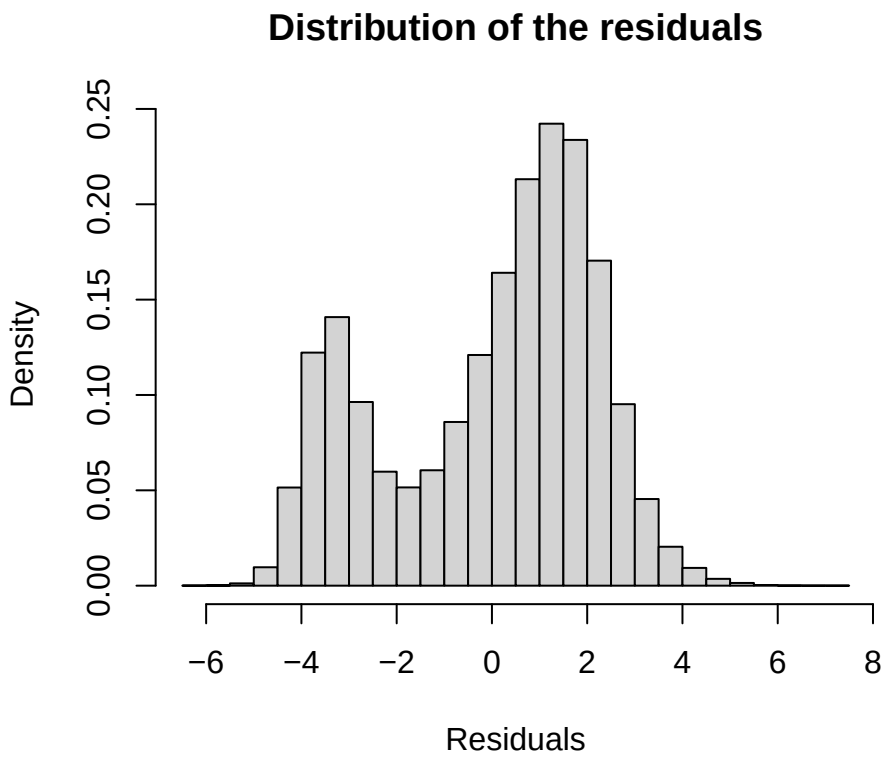

Figure 2: *The residuals follow a bimodal distribution.*

5.2 Limitations

I initially wanted to place a heavier emphasis on treatment history. However, this problem proved to be too complex for the scope of this project. In addition to the treatment category (i.e. mechanical removal or medicinal treatment), which are boolean variables, the dataset goes into a lot more detail regarding the specifics of the treatment. For example, medicine can be administered either through bath treatments or via feed, and treatments may be implemented across entire farm sites or affect just part of the site (and in the latter case, the portion of affected cages is unspecified).

There is also an issue of causality when considering treatment events, since treatments are reactively implemented in response to the lice count. Since farmers are obliged to take action whenever lice counts exceed the legally defined limit, the presense or absense of a treatment event would likely appear to be a very good predictor. However, this gets the causality in reverse and is not very useful. Creating a new time corrected variable would be trivial (e.g. No. of weeks since last treatment or a simple boolean value specifying whether a treatment event has occured within a specified time-frame). However, I am not familiar with the statistical implications of this, especially considering the varying time intervals between observations.

I had high hopes for the third treatment category, i.e. the presence or absence of cleanerfish (species and

number of individuals is also recorded in the dataset, but this would be less useful due to the varying and unknown scale of production per farm site). Needless to say, I was very dissappointed to learn that this variable had been discontinued (see Section 2.4.3).

Unfortunately, measurements of salinity level were not included in the dataset. While salinity data are available from other sources, the imprecise unit of time (i.e. week number) and the unevenly distributed distance between farm sites and the nearest measuring station made me decide against pursuing this any further.

5.3 Conclusion

None of the candidate models that I created were very accurate, but one was identified as at least a little bit better than the others. Despite having access to a great deal of individual observations, several of the potential predictor variables were either incomplete, ambiguous, or far too complex to include in the project.

However, I was able to select a statistical model which I believe could handle the task if it had access to more and better organised data.

REFERENCES

Asplin, L., Johnsen, I. A., Sandvik, A. D., Albretsen, J., Sundfjord, V., Aure, J., & Boxaspen, K. K. (2014). Dispersion of salmon lice in the hardangerfjord. *Marine Biology Research*, *10*(3), 216–225. https://doi.org/10.1080/17451000.2013.810755

BarentsWatch. (2016). Charting farmed fish health. In *BarentsWatch*. https://www.barentswatch.no/en/articles/fish-health/

Bartoń, K. (2023). *MuMIn: Multi-model inference*. R package version 1.47.5, https://CRAN.R-project.org/package=MuMIn

Bates, D., Mächler, M., Bolker, B., & Walker, S. (2015). Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, *67*(1), 1–48. https://doi.org/10.18637/jss.v067.i01

Bechmann, R. K., Arnberg, M., Gomiero, A., Westerlund, S., Lyng, E., Berry, M., Agustsson, T., Jager, T., & Burridge, L. E. (2019). Gill damage and delayed mortality of northern shrimp (*Pandalus borealis) after short time exposure to anti-parasitic veterinary medicine containing hydrogen peroxide. Ecotoxicology and Environmental Safety*, *180*, 473–482. https://doi.org/10.1016/j.ecoenv.2019.05.045

Coates, A., Phillips, B. L., Bui, S., Oppedal, F., Robinson, N. A., & Dempster, T. (2021). Evolution of salmon lice in response to management strategies: A review. *Reviews in Aquaculture*, *13*(3), 1397–1422. https://doi.org/https://doi.org/10.1111/raq.12528

Endr. i forskrifter om akvakultur – transporttilpasning, oppbevaring, bruk og produksjon av rensefisk. (2018). Forskrift om endring i forskrifter om akvakultur for deres tilpasning til transport, oppbevaring, bruk og produksjon av rensefisk (FOR-2018-04-19-674). Retrieved from: https://lovdata.no/LTI/forskrift/2018-04-19-674

Fernandez, M. A. L., Gelaye, B., Vander Weele, T., Hernandez-Diaz, S., Williams, M. A., Ananth, C., Qui, C., Sanchez, S., Cynthia Ferre, A. M. S.-R., Holzman, C., et al. (2014). *Modelling time as a circular scale*. https://scholar.harvard.edu/files/malf/files/circularttime-maluque_0.pdf

Forskrift om lakselusbekjempelse. (2012). Forskrift om bekjempelse av lakselus i akvakulturanlegg (FOR-2012-12-05-1140). Retrieved from: https://lovdata.no/forskrift/2012-12-05-1140

Gillibrand, P. A., & Willis, K. J. (2007). Dispersal of sea louse larvae from salmon farms: Modelling the influence of environmental conditions and larval behaviour. *Aquatic Biology*, *1*(1), 63–75. https://doi.org/10.3354/ab00006

Hamner, B., & Frasco, M. (2018). *Metrics: Evaluation metrics for machine learning*. R package version 0.1.4, https://github.com/mfrasco/Metrics, https://cran.r-project.org/package=Metrics

Håstein, T., Sømme, L. S., Halleråker, J. H., & Vøllestad, leif A. (2022). Lakselus. In *Store norske leksikon*. https://snl.no/lakselus (Accessed on 25/11/2023)

Hemmingsen, W., MacKenzie, K., Sagerup, K., Remen, M., Bloch-Hansen, K., & Imsland, A. K. D. (2020). Caligus elongatus and other sea lice of the genus *Caligus* as parasites of farmed salmonids: A review. *Aquaculture*, *522*, 735160. https://doi.org/10.1016/j.aquaculture.2020.735160

Henderson Jr, C. R. (1982). Analysis of covariance in the mixed model: Higher-level, nonhomogeneous, and random regressions. *Biometrics*, 623–640.

Heuch, P., Øines, Ø., Knutsen, J. A., & Schram, T. A. (2007). Infection of wild fishes by the parasitic copepod *Caligus elongatus* on the south east coast of Norway. *Diseases of Aquatic Organisms*, *77*(2), 149–158. https://doi.org/10.3354/dao01833

Imsland, A. K. D., Sagerup, K., Remen, M., Block-Hansen, K., Hemmingsen, W., & Myklebust, E. A. (2019). Kunnskaps- og erfaringskartlegging av skottelus. *Akvaplan-Niva Rapport: APN-60795*. https://www.fhf.no/prosjekter/prosjektbasen/901539/

Institute of Marine Research. (2020). *Topic: Sea lice*. https://www.hi.no/en/hi/temasider/species/sea-lice (Accessed on 25/11/2023)

Johansen, A.-M. (2017). *Dyr lus og dyrere fôr*. Nofima. https://nofima.no/resultater/dyr-lus-og-dyrere-for/ (Accessed on 23/11/2023)

Johnsen, I. A., Asplin, L., Sandvik, A. D., & Serra-Llinares, R. M. (2016). Salmon lice dispersion in a northern norwegian fjord system and the impact of vertical movements. *Aquaculture Environment Interactions*, *8*, 99–116. https://doi.org/10.3354/aei00162

Knief, U., & Forstmeier, W. (2021). Violating the normality assumption may be the lesser of two evils. *Behavior Research Methods*, *53*(6), 2576–2590. https://doi.org/10.3758/s13428-021-01587-5

Marty, G. D., Saksida, S. M., & Quinn, T. J. (2010). Relationship of farm salmon, sea lice, and wild salmon populations. *Proceedings of the National Academy of Sciences*, *107*(52), 22599–22604. https://doi.org/10.1073/pnas.1009573108

Nakagawa, S., & Schielzeth, H. (2013). A general and simple method for obtaining R^2 from generalized linear mixed-effects models. *Methods in Ecology and Evolution*, *4*(2), 133–142. https://doi.org/10.1111/j.2041-210x.2012.00261.x

Norwegian Directorate of Fisheries. (2022). *Atlantic salmon, rainbow trout and trout - grow out production.* Retrieved from: https://www.fiskeridir.no/English/Aquaculture/Statistics/Atlantic-salmon-and-rainbow-trout/grow-out-production

R Core Team. (2023). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. https://www.R-project.org/

Revie, C., Dill, L., Finstad, B., & Todd, C. (2009). *Sea lice working group report.* Norwegian Institute for Nature Research (NINA). ISBN: 978-82-426-2048-4. https://brage.nina.no/nina-xmlui/handle/11250/2725197

Schielzeth, H., Dingemanse, N. J., Nakagawa, S., Westneat, D. F., Allegue, H., Teplitsky, C., Réale, D., Dochtermann, N. A., Garamszegi, L. Z., & Araya-Ajoy, Y. G. (2020). Robustness of linear mixed-effects models to violations of distributional assumptions. *Methods in Ecology and Evolution*, *11*(9), 1141–1152. https://doi.org/10.1111/2041-210X.13434

Torrissen, O., Jones, S., Asche, F., Guttormsen, A., Skilbrei, O. T., Nilsen, F., Horsberg, T. E., & Jackson, D. (2013). Salmon lice–impact on wild salmonids and salmon aquaculture. *Journal of Fish Diseases*, *36*(3), 171–194. https://doi.org/10.1073/pnas.1009573108

Venables, W. N., & Ripley, B. D. (2002). *Modern applied statistics with S* (Fourth). Springer. ISBN: 0-387-95457-0. https://www.stats.ox.ac.uk/pub/MASS4/

Ver Hoef, J. M., & Boveng, P. L. (2007). Quasi-poisson vs. Negative binomial regression: How should we model overdispersed count data? *Ecology*, *88*(11), 2766–2772. https://doi.org/10.1890/07-0043.1

Warnes, G. R., Bolker, B., Lumley, T., Magnusson, A., Venables, B., Ryodan, G., & Moeller, S. (2023). *Gtools: Various R programming tools.* https://github.com/r-gregmisc/gtools R package version 3.9.5

Warrington, N. M., Tilling, K., Howe, L. D., Paternoster, L., Pennell, C. E., Wu, Y. Y., & Briollais, L. (2014). Robustness of the linear mixed effects model to error distribution assumptions and the consequences for genome-wide association studies. *Statistical Applications in Genetics and Molecular Biology*, *13*(5), 567–587. https://doi.org/10.1515/sagmb-2013-0066

Wickham, H. (2023). *Stringr: Simple, consistent wrappers for common string operations.* R package version 1.5.1, https://github.com/tidyverse/stringr, https://stringr.tidyverse.org

Wickham, H., François, R., Henry, L., Müller, K., & Vaughan, D. (2023). *Dplyr: A grammar of data manipulation.* R package version 1.1.4, https://github.com/tidyverse/dplyr, https://dplyr.tidyverse.org

Wickham, H., Hester, J., & Bryan, J. (2023). *Readr: Read rectangular text data.* R package version 2.1.4, https://github.com/tidyverse/readr, https://readr.tidyverse.org

Wickham, H., Vaughan, D., & Girlich, M. (2023). *Tidyr: Tidy messy data.* R package version 1.3.0, https://github.com/tidyverse/tidyr, https://tidyr.tidyverse.org