Exploring Ecological Data

Dr Ellen Bell

2023-03-05

Contents

Chapter 1

Background

Today we will be using the `and_vertebrates` data set from the lterdatasampler package. This data set is based on a real longitudinal dataset collected from populations of West Slope cutthroat trout (*Onchorhyncus clarki clarki*, monitored since 1987) and Pacific Giant Salamander (*Dicamptodon tenebrosus*, monitored since 1993) from two reaches of Mack Creek in the H.J. Andrews Experimental Forest, Oregon, USA. The two reaches are in a section of clearcut forest (ca. 1963) and an upstream 500 year old coniferous forest. Sampling was performed with 2-pass electrofishing and all captured vertebrates were measured and weighed. Additional classifications of immediate habitat type were also recorded.

Hopefully you all had a chance to read the paper that accompanies this workshop here. What did you think of it?

> Note down some of your observations, interpretations or thoughts, we will have a class discussion around these in the last 20 minutes of the workshop.

1.1 Learning objectives

Ecological data is notoriously complex to analyse and interpret, mainly because there are so many variables that need to be accounted for and patterns change in both space and time. This workshop will;

- Give you practice in tidying up and wrangling a messy, real, ecological data set
- Help you develop data exploration and plotting skills
- Allow you to practice your R coding skills
- Get you thinking about how best to make sense of and interpret ecological data

Disclaimer - This workshop assumes prior introduction and use of posit Cloud as a platform for using RStudio.

1.2 The data format

This data set is formatted as a tibble with 32,209 rows and 16 variables:

- year - a number indicating the observation year
- sitecode - a character denoting the coded name of sample area
- section - a character denoting the section in Mack Creek (CC = clear cut forest, OG = upstream old growth coniferous forest)
- reach - a character denoting the reach sampled from each section; L = lower reach (0 - 50 meters), M = middle reach (50 - 100 meters), U = upper reach (100 - 150 meters))
- pass - a number denoting the electroshocking pass number, either 1 or 2
- unitnum - a number denoting the channel unit number
- unittype - a character denoting the channel unit classification type (C = cascade, I = riffle, IP = isolated pool (not connected to channel), P = pool, R = rapid, S = step (small falls), SC = side channel, NA = not sampled by unit)
- vert_index - a number denoting the unique index for each vertebrate
- pitnumber - a number denoting the unique tag number embedded into vertebrate (tagging started in 2007)
- species - a character denoting species measured
- length_1_mm - a number denoting vertebrate length in millimeters; total or snout-fork length for trout, and snout-vent length for salamanders
- length_2_mm - a number denoting snout-tail length in millimeters (for Coastal giant salamander only)
- weight_g - a number denoting vertebrate mass in grams
- clip - a character denoting the fin clip type for cutthroat trout, ended in 2006 (LV = left ventrical fin; LVRV = left and right ventrical fins; RV = right ventrical fin; NONE = no ventral fin clip)
- sampledate - a date denoting the date of observation
- notes - a character denoting additional comments

1.3 Setting up your project

We will be working on a new data set, generally I recommend starting a new R Project every time you start working on new and unrelated data sets. Go to posit Cloud, open a **New R Studio Project** and name it `mack_creek`. Spend a few minutes setting up your work space. Remember you will need to create sensible places to save scripts, data and figures. You will also need to freshly install any required packages.

For this workshop you will need to install the following packages:

- tidyverse
- lterdatasampler

1.3.1 Script set up

Its fairly safe to say we will be creating some new scripts, so open a new **R script** and set it up with a sensible title, your name and date, commented out. Remember to use `library()` to load your freshly installed packages. You then use the following line to load our dataset;

```
vertebrates <- and_vertebrates
```

Chapter 2

Checking the data

Once you have imported and loaded your `vertebrates` dataset into your new R project you will need to spend a few minutes checking it over to make sure that R has correctly identified and formatted your variables.

2.1 Make sure your data have imported correctly and are easy to manipulate

1) First of all make sure your data have imported correctly, do you have the expected numbers of rows and columns?

   Hint - consider using functions such as `ncol()`, `nrow()`, `colnames()` to check this.

2) Then check that R has correctly identified the data types for each variable. Do you need to adjust any variables to factors?

   Hint - consider using functions such as `glimpse()` to check.

3) Now you can check your variable names, are they nice and concise, do you want to change them?

   Hint - you can use the `rename()` function to change variable names if you wish to.

2.2 Checking for errors

Now we know our that R has read our data correctly and we have all of our variables named as we would like we can run a few further checks. Frequently you will find that your data have originally been manually entered into a spreadsheet, maybe from hand drafted notes in a field or lab notebook. Every time something

is copied there is the opportunity for error. Maybe a row has been accidentally duplicated, or there is a typo or maybe some data is missed out altogether. There are a few tricks we can use to check for each of these and to make sure we are confident in the fidelity of our dataset.

1) Checking for duplicates - When you are manually entering data into a spreadsheet, it is very easy to accidentally enter the same row twice. With very large data sets this is something that is very hard to pick out by eye. Thankfully R has some very useful functions to check for this. Try running;

```
vertebrates %>%
  duplicated() # check for duplicated rows
```

> Note that %>% used here is just another notation for piping, rather like the + that ggplot uses. This means that the output from one function is fed directly into the next function.

This chunk of code will spit out a long list of TRUE (row is duplicated) or FALSE (row is not duplicated) statements. Again not very human readable, especially if we have a very large data set. Try amending the code to;

```
vertebrates %>%
  duplicated() %>% # check for duplicated rows
  sum() # Sums any TRUE statements in the list
```

Do we have any duplicated rows?

2) Checking for typos - As with duplicates it is very easy to enter a typo when manually entering data into a spreadsheet. Generally if you have been collecting continuous data you will have an idea of what a sensible upper and lower bound within your data set should be. We can use the `summarise()` function to see what these are within this data set, as shown below;

```
vertebrates %>%
  group_by(species) %>%
  summarise(min=min(length_1_mm, na.rm=TRUE), # reports the minimum value in the lengt
            max=max(length_1_mm, na.rm=TRUE)) # reports the maximum value in the lengt
```

Try manipulating the above chunk to report the upper and lower bounds for the weight variable in your data set. Do these values all look reasonable to you?

But how can we check for typos in categorical data? We can use the `distinct()` function to identify all of the options stored under a categorical variable name. Try using the following chunk;

```
vertebrates %>%
  distinct(species) # reports the categories stored under species
```

So do you think there are any potential typos in your data set? Are there any

further categorical variables you may wish to check?

3) Checking for missing data - finally sometimes when entering data manually, you may miss or delete a spreadsheet cell by mistake, leaving it empty. Again this is really difficult to spot by eye in a large data set. Try running the following chunk of code, can you work out what each line does? Try adding comments to it in your script yourself.

```
vertebrates %>%
  is.na() %>%
  sum()
```

Click-me to check your code interpretation

So here you are piping your initial data set `vertebrates` into the `is.na()` function which is then looking for cells containing N/A and reporting a TRUE/FALSE data frame (where TRUE indicates N/A). We are then piping that output straight into the `sum()` function which is summing the number of TRUE values.

Do we have any missing data? Do you think its reasonable that there may be some missing data? You can investigate individual variables using;

```
sum(is.na(vertebrates$species))
```

2.3 Tidying up

So from the previous exercises that the data set contains four categories in the `species` variable; two species of salamander, one species of trout and some NA's. It would be good to know what the representation is of each of these groups in this data set.

Try running the following;

```
vertebrates %>%
  group_by(species) %>%
  count()
```

What do you think this piece of code has done? From this information, do you think it may be worth removing some categories from the data set?

Click-me to check your analysis

We can see that three species in this data set and a category that are unidentified (NA). The Cascade torrent salamander has only 15 data entries and 3 entries have an unknown species (NA), compared to thousands of entries for the other species. It therefore makes good analytical sense to remove those entries. Try using the following to filter them out, for now.

```
vertebrates <- vertebrates %>%
  filter(species != "Cascade torrent salamander") %>%
  filter(species != "NA")
```

You can check this worked by running the last piece of code again.

Now we are ready to start exploring the data set from an ecological point of view.

Chapter 3

Exploring the data

We are interested in exploring the potential impact of different land management strategies and habitats on abundance and biomass of cutthroat trout and coastal giant salamander.

3.1 Plotting abundance through time and space

To begin with it would be good to know how abundance of both species has changed through time.

Use glimpse() to look at your data set once more. If we wanted to look at abundance how might we go about it?

> Hint - we can use the function count() to derive abundances.

Click-me to check your analysis

This data set contains a lot of individual information, which is great and we will come back to that later. But we want to be able to look at species abundance to start with so we will need to do a little bit of data wrangling. Try running the following on your data set;

```
vert_counts <- vertebrates %>%
  count(year, species, section, reach, unittype)
```

Use glimpse() to look at your new data frame, do you understand what has been calculated? use the # to comment on your new piece of code.

Now that we have some abundance data lets have a look at how these have changed through time. Run the following piece of code;

```
vert_counts %>% # Pipe vert_counts into ggplot
  ggplot(aes(x = year, y = n)) + # Plot year against abundance (n)
```

```
  geom_point() + # Produce a scatter plot
  theme_bw() + # Set the theme to a basic black and white theme
  facet_wrap(section~species) # produce a separate plot for different sections and spe
```

Look at what you have produced, is it very meaningful? There is some information in these plots about the land management for each reach of creek but no further spatial information has been included, each reach is 150 meters long and contains several different habitat types. We may have clumped too much data into these figures. But we can separate this out a little further, try this;

```
vert_counts %>% # Pipe vert_counts into ggplot
  ggplot(aes(x = year, y = n, colour = reach)) + # Plot year against abundance (n), co
  geom_point() + # Produce a scatter plot
  theme_bw() + # Set the theme to a basic black and white theme
  facet_wrap(section~species) # produce a separate plot for different sections and spe
```

Now we have added a little more resolution to our figures by splitting the data into 50 meter sections, but do we think it might be more informative to have habitat information presented instead of which reach division the abundance data were collected from? See if you can manipulate the last chunk of code to produce the following figure (Figure 3.1).

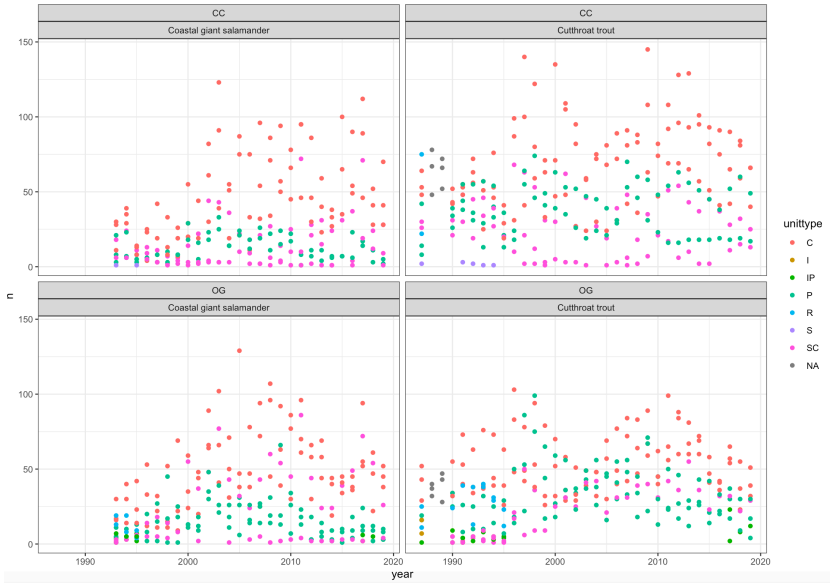

Figure 3.1: Abundance by year, coloured by habitat

This analytical process might have given us some idea of the abundances across year, reach and habitat type but its still pretty tricky to pull out much of a pattern from these data. So we need to do some more data wrangling to produce

a plot that is more meaningful and easier to interpret. Look again at your plots, do you think that there are multiple data points per year and unittype? Could these be summarised more effectively by some measures of central tenancy and spread?

Try running the following to make a summary tibble;

```
summary <- vert_counts %>%
  group_by(year, species, section, unittype) %>%
  summarise(mean=mean(n),
            sd=sd(n),
            n=n())
summary
```

Have a look at the tibble produced, do you understand what each line of the above code has done? Try adding comments `#` to your code.

It would also be useful to know the standard error of the mean as well. We can use the `mutate` function to add additional columns onto our summary stats table. Remember we can calculate the standard error of the mean using;

$$SEM = \frac{SD}{\sqrt{n}}$$

Add the following line of code to the end of the previous chunk, don't forget to include the piping syntax `>%>`;

```
mutate(sem = sd/sqrt(n))
```

Your summary tibble should now look like this (note that this is just an exert, yours will be larger);

```
> summary
# A tibble: 386 × 8
# Groups:   year, species, section [120]
    year species         section unittype  mean    sd     n   sem
   <dbl> <chr>           <chr>   <chr>    <dbl> <dbl> <int> <dbl>
 1  1987 Cutthroat trout CC      C         55    8.19     3  4.73
 2  1987 Cutthroat trout CC      P         21.3 18.1      3 10.5
 3  1987 Cutthroat trout CC      R         48.5 37.5      2 26.5
 4  1987 Cutthroat trout CC      S          2   NA        1  NA
 5  1987 Cutthroat trout CC      SC        28    2.83     2  2
 6  1987 Cutthroat trout OG      C         41.3 11.6      3  6.69
 7  1987 Cutthroat trout OG      I         11.5  6.36     2  4.5
```

You might notice that there are some `NA` values stored here. This is because where there is only one data value per category combination (e.g. 1987, Cutthroat trout, CC, S) we cant calculate a standard deviation or a standard error of the mean. We don't want these `NA` values to cause future errors so we are just

going to replace them with `0`. Add the this line to your last chunk (remember to pipe).

```
mutate(sem = replace_na(sem, 0))
```

Now we can use this

```
summary %>%
  ggplot(aes(x = year, y = mean, colour = unittype)) +
  geom_line() +
  geom_pointrange(aes(ymax = mean+sem, ymin = mean-sem)) +
  theme_bw() +
  facet_wrap(section~species)
```

To produce plots that look like this;

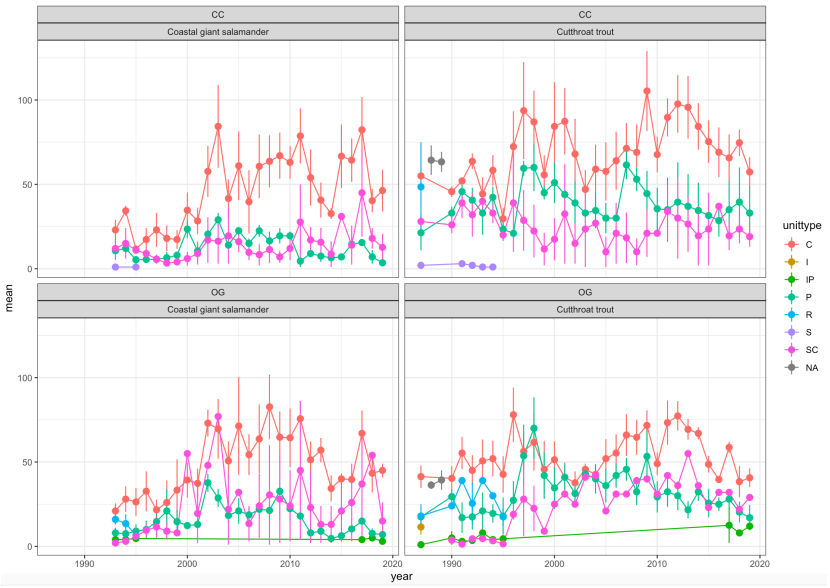

Figure 3.2: Mean abundance by year, coloured by habitat

Once again make sure you understand what each line of code has done, try adding comments `#` to the script. Spend some time looking at these plots, what do they tell you? Remember that you have plotted mean and standard error for abundance across year, habitat, land management strategy and two species here, which gives you multiple questions you could ask of the data. It may be of interest to know that Mack Creek flooded in 1996. What impact (if any) did this have on our abundances? Remember to look at the variation in the data, don't just focus on the mean.

> Note down some of your observations, interpretations or thoughts, we will have a class discussion around these in the last 20 minutes

of the workshop.

3.2 Comparing habitat and land management strategy impacts on abundance

We may wish to directly compare abundances between habitat and land management strategy (removing time as a variable). How do you think you might go about this? See if you can manipulate some of the above scripts to produce an appropriate plot;

> Hint - I would use geom_boxplot

What do these new plots tell you about differences in abundance in either trout or salamander species between different habitats/land management strategy?

> Note down some of your observations, interpretations or thoughts, we will have a class discussion around these in the last 20 minutes of the workshop.

3.3 Bonus Challenge

Our original data set also contained data on salamander and trout standard length and weight. Have a go at exploring these data, do you think habitat or land management has any impact on the biomass of trout or salamanders? Think about the most appropriate plots you could produce to explore these questions.

> Note down some of your observations, interpretations or thoughts, we will have a class discussion around these in the last 20 minutes of the workshop.

3.4 Discussion

The last 20 minutes of this session will be used for a class discussion around the observations you have made from the plots produced. Think about any major patterns that have emerged, can you link these patterns back to any one single variable?

3.5 Before you leave

Please log out of posit Cloud before you leave and make sure you save your script, we will come back to it in future workshops.

Chapter 4

References

Gregory, S.V. and Arismendi, I. (2020). Aquatic Vertebrate Population Study in Mack Creek, Andrews Experimental Forest, 1987 to present ver 14. Environmental Data Initiative. https://doi.org/10.6073/pasta/7c78d662e847cdbe33584add8f809165 Horst A., Brun J. (2023). lterdatasampler: Educational dataset examples from the Long Term Ecological Research program. R package version 0.1.1, https://github.com/lter/lterdatasampler. Kaylor. M. and Warren, D. (2017) Linking riparian shade and the legacies of forest management to fish and vertebrate biomass in forested streams. Ecosphere, 8(6), e01845. https://doi.org/10.1002/ecs2.1845 Wickham, H., Averick, M., Bryan, j., Chang, W., D'Agostino McGowan , L., François, R., Grolemund, G. (2019). "Welcome to the tidyverse." Journal of Open Source Software 4 (43): 1686. https://doi.org/10.21105/joss.01686.