functions-after

Azul Carrillo

2022-11-10

A function for the Portal data set

1. loading the needed data into r with the read.csv func and asigning it a name

```
surveys <- read.csv(file = "../raw-data/surveys.csv")
species <- read.csv(file = "../raw-data/species.csv")
plots <- read.csv(file = "../raw-data/plots.csv")
```

2. first loading that package that has the inner join function so that it can be used combined the survey and species tables into a single data frame using the inner_join function joined them by species

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
surveys_species <- inner_join(surveys, species, by = "species_id")
```

need the col names in order to add the correct argument

```
colnames(surveys_species)
```

```
##  [1] "record_id"       "month"           "day"             "year"
##  [5] "plot_id"         "species_id"      "sex"             "hindfoot_length"
##  [9] "weight"          "genus"           "species"         "taxa"
```

3. Write a function that: Takes three arguments - a data frame (the combined table created before), a genus name, and a species name, Uses functions from the dplyr package to produce a data frame with two columns: year and count, where count is the number of individuals (i.e., the number of rows) for the species indicated by genus and species in that year, Returns the resulting data frame. Note that this data frame is a time series by year.

gives info for all genus adn all years but addind the filter it changes it

```
library(dplyr)
func_1 <- function(surveys_species, genus , species){
results <- surveys_species %>%
  group_by(year, genus, species) %>%
  summarise(count = n())
  return(results[ ,c("year", "count")])
}

results <- surveys_species %>%
  group_by(year, genus, species) %>%
  summarise(count = n())
```

```
## ‘summarise()‘ has grouped output by ’year’, ’genus’. You can override using the
## ‘.groups‘ argument.
```

```
results[ ,c("year", "count")]
```

```
## # A tibble: 509 x 2
## # Groups:   year [26]
##     year count
##    <int> <int>
##  1  1977     7
##  2  1977   264
##  3  1977    12
##  4  1977    98
##  5  1977    31
##  6  1977    10
##  7  1977     7
##  8  1977    17
##  9  1977    31
## 10  1977     6
## # ... with 499 more rows
```

#try 2 added the filter to the function to specify what we want from the genus column and the species column removed the subsetting at the return function because it was cousing an issue in the output gruped only by year using the group_by

```
library(dplyr)
func_2 <- function(combined, genus_name , species_name){
results <- combined %>%
  filter(genus == genus_name , species == species_name) %>%
  group_by(year) %>%
  summarise(count = n())
  return(results)
}

head(surveys_species)
```

```
##   record_id month day year plot_id species_id sex hindfoot_length weight
## 1         1     7  16 1977       2         NL   M              32     NA
## 2         2     7  16 1977       3         NL    M              33      NA
## 3         3     7  16 1977       2         DM    F              37      NA
## 4         4     7  16 1977       7         DM    M              36      NA
## 5         5     7  16 1977       3         DM    M              35      NA
## 6         6     7  16 1977       1         PF    M              14      NA
##         genus  species   taxa
## 1     Neotoma albigula Rodent
## 2     Neotoma albigula Rodent
## 3   Dipodomys merriami Rodent
## 4   Dipodomys merriami Rodent
## 5   Dipodomys merriami Rodent
## 6 Perognathus   flavus Rodent
```

running a pipe befor creating the function to see that our function is doing the correct command

```
results <- surveys_species %>%
  filter(genus == "Dipodomys", species == "merriami" ) %>%
  group_by(year) %>%
  summarise(count = n())
```

#4 creating a new object that uses our function to create a time series for the genus “Dipodomys” and species “merriami”

```
plot1 <- func_2(surveys_species, "Dipodomys", "merriami")
plot1
```

```
## # A tibble: 26 x 2
##     year count
##    <int> <int>
##  1  1977   264
##  2  1978   389
##  3  1979   209
##  4  1980   493
##  5  1981   559
##  6  1982   609
##  7  1983   528
##  8  1984   396
##  9  1985   667
## 10  1986   406
## # ... with 16 more rows
```

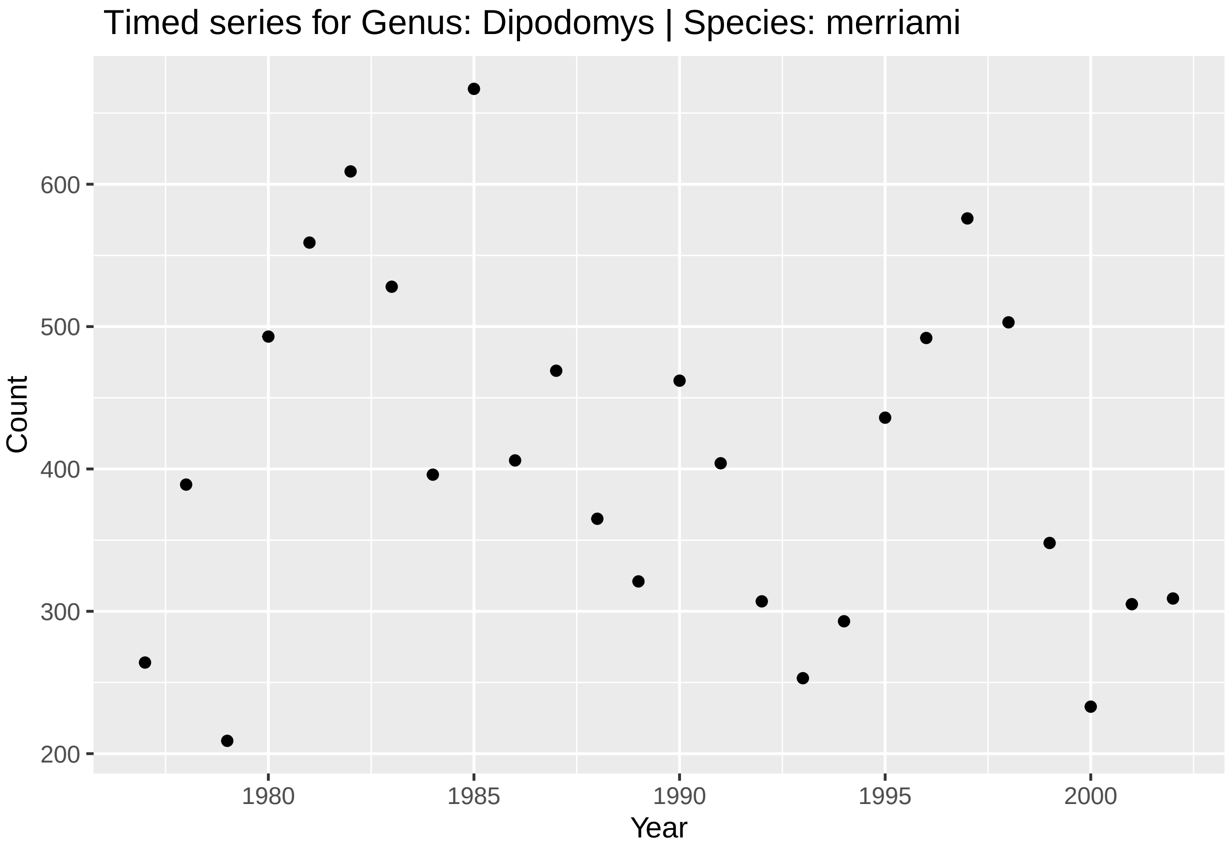
#5 creating a new object that uses our function to create a time series for the genus “Chaetodipus” species “penicillatus”

```
plot2 <- func_2(surveys_species, "Chaetodipus", "penicillatus")
plot2
```

```
## # A tibble: 26 x 2
##     year count
##    <int> <int>
##  1  1977     7
##  2  1978    24
##  3  1979    20
##  4  1980    17
##  5  1981    20
##  6  1982    32
##  7  1983    32
##  8  1984    26
##  9  1985    35
## 10  1986    26
## # ... with 16 more rows
```

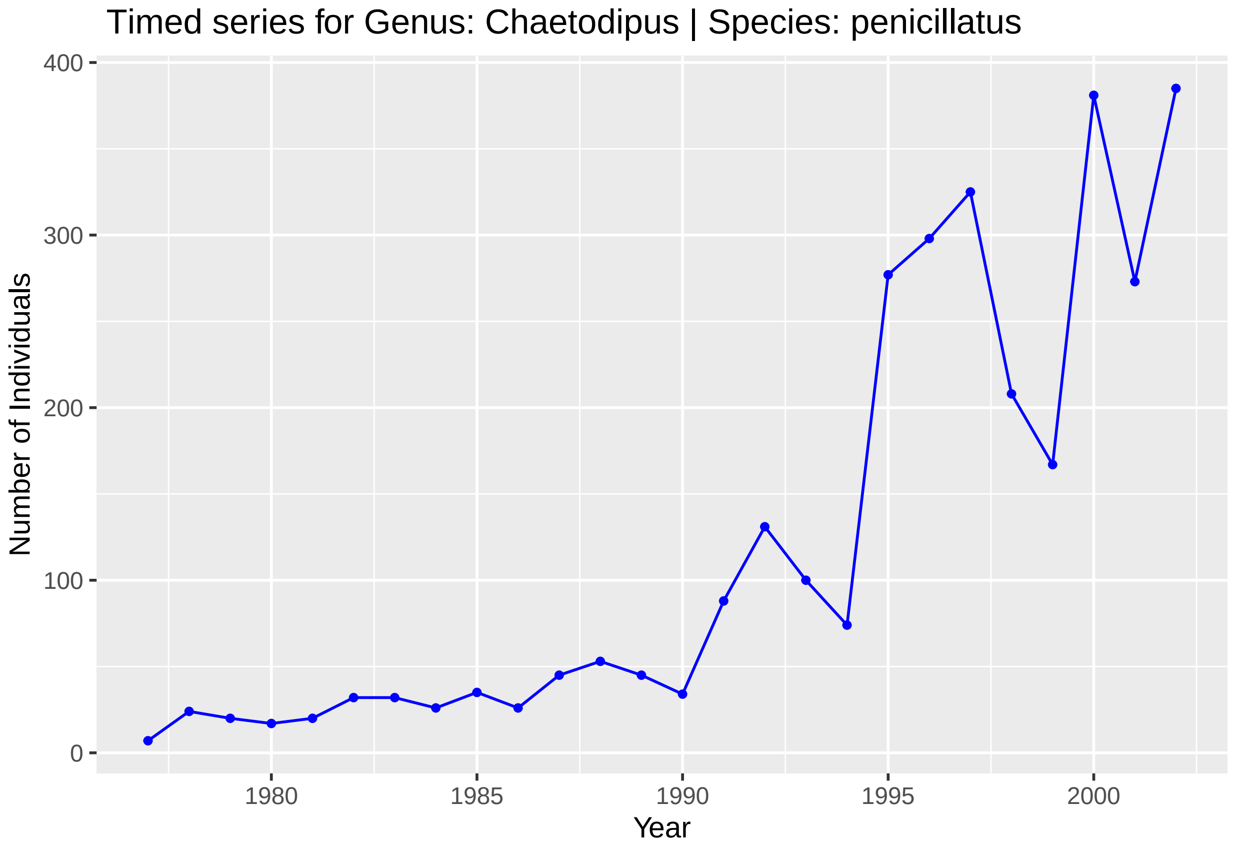
#4 time-series plot of genus Chaetodipus and species

```
library("ggplot2")
ggplot(data = plot1, mapping = aes(x = year, y = count)) +
  geom_point() +
  labs(x = "Year", y = "Count", title = " Timed series for Genus: Dipodomys | Species: merriami")
```



#5 time-series plot of genus Chaetodipus and species penicillatus added color to the plots using the color func and a size of 1 using the size func added labels using labs added a line that connects the plots using geom_ line func

```
library("ggplot2")
ggplot(data = plot2, mapping = aes(x = year, y = count)) +
  geom_point(color = "blue", size = 1 ) +
  geom_line(color="blue" ) +
  labs(x = "Year", y = "Number of Individuals", title = " Timed series for Genus: Chaetodipus | Species
```


Timed series for Genus: Chaetodipus | Species: penicillatus
Number of Individuals
400
300
200
100
0
1980
1985
1990
1995
2000
Year