

Week12_Assignment

Ellen Bledsoe

2024-04-08

Assignment

Purpose

The goal of this assignment is to practice writing and using for loops for iteration.

Task

Write R code to successfully answer each question below.

Criteria for Success

- Code is within the provided code chunks or new code chunks are created where necessary
- Code chunks run without errors
- Code chunks have brief comments indicating which code is answering which part of the question
- Code will be assessed as follows:
 - Produces the correct answer using the requested approach: 100%
 - Generally uses the right approach, but a minor mistake results in an incorrect answer: 90%
 - Attempts to solve the problem and makes some progress using the core concept, but returns the wrong answer and does not demonstrate comfort with the core concept: 50%
 - Answer demonstrates a lack of understanding of the core concept: 0%
- Any questions requiring written answers are answered with sufficient detail

Due Date

April 15 at midnight MST

Assignment Exercises

Set Up

Be sure to load the `tidyverse` to use later in the assignment.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
## v dplyr      1.1.4      v readr      2.1.5
## v forcats    1.0.0      v stringr   1.5.0
## v ggplot2    3.4.2      v tibble   3.2.1
## v lubridate  1.9.2      v tidyr    1.3.0
## v purrr      1.0.1
## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

1. For Loop Basics (30 pts)

Complete the following tasks.

I recommend copying the commented code and making changes to the copy. That way, if you have made edits that you haven't kept track of, you have the original code to start over as needed.

- The code below prints the numbers 1 through 5 one line at a time. Modify it to print each of these numbers multiplied by 3.

```
numbers <- c(1, 2, 3, 4, 5)
# for (number in numbers){
#   print(number)
# }
```

```
print("1a")
```

```
## [1] "1a"
```

```
numbers <- c(1, 2, 3, 4, 5)
for (number in numbers){
  print(3 * number)
}
```

```
## [1] 3
## [1] 6
## [1] 9
## [1] 12
## [1] 15
```

- Write a for loop that loops over the following vector and prints out the mass in kilograms (`mass_kg = 2.2 * mass_lb`)

```
mass_lbs <- c(2.2, 3.5, 9.6, 1.2)
```

```
print("1b")
```

```
## [1] "1b"
```

```
for (mass in mass_lbs) {
  mass_kgs <- 2.2 * mass
  print(mass_kgs)
}
```

```
## [1] 4.84
## [1] 7.7
## [1] 21.12
## [1] 2.64
```

c. Complete the code below so that it prints out the name of each bird one line at a time.

```
# birds = c("robin", "woodpecker", "blue jay", "sparrow")
# for (i in 1:length(____)){
#   print(birds[____])
# }
```

```
birds = c('robin', 'woodpecker', 'blue jay', 'sparrow')
for (i in 1:length(birds)){
  print(birds[i])
}
```

```
## [1] "robin"
## [1] "woodpecker"
## [1] "blue jay"
## [1] "sparrow"
```

d. Complete the code below so that it stores one area for each radius.

```
radius <- c(1.3, 2.1, 3.5)
# areas <- vector(_____ = "numeric", length = _____)
# for (i in 1:length(_____)){
#   areas[i] <- pi * radius[i] ^ 2
# }
# areas
```

```
print("1d")
```

```
## [1] "1d"
```

```
areas <- vector(mode = "numeric", length = length(radius))
for (i in 1:length(areas)){
  areas[i] <- pi * radius[i] ^ 2
}
areas
```

```
## [1] 5.309292 13.854424 38.484510
```

e. Complete the code below to calculate an area for each pair of **lengths** and **widths**, store the areas in a vector, and after they are all calculated print them out:

```
lengths = c(1.1, 2.2, 1.6)
widths = c(3.5, 2.4, 2.8)
# areas <- vector(length = _____)
# for (i in _____) {
#   areas[___] <- lengths[___] * widths[___]
# }
areas
```

```
## [1] 5.309292 13.854424 38.484510
```

```
print("1e")
```

```
## [1] "1e"
```

```
areas <- vector(length = length(lengths))
for (i in 1:length(lengths)) {
  areas[i] <- lengths[i] * widths[i]
}
areas
```

```
## [1] 3.85 5.28 4.48
```

2. Size Estimates by Name (30 pts)

This is a followup to “Size Estimates by Name” from last week.

Download the `dinosaur_lengths.csv` file from D2L and place it in the correct directory. Read the file into R.

Write a function `mass_from_length()` that uses the equation $\text{mass} \leftarrow a * \text{length}^b$ to estimate the size of a dinosaur from its length. This function should take two arguments, `length` and `species`. For each of the following inputs for `species`, so use the associated `a` and `b` values to estimate the species mass using these equations:

- *Stegosauria*: $\text{mass} = 10.95 * \text{length}^2.64$ (Seebacher 2001).
- *Theropoda*: $\text{mass} = 0.73 * \text{length}^3.63$ (Seebacher 2001).
- *Sauropoda*: $\text{mass} = 214.44 * \text{length}^1.46$ (Seebacher 2001).
- For any other value of `species`: $\text{mass} = 25.37 * \text{length}^2.49$

- Use this function and a for loop to calculate the estimated mass for each dinosaur, store the masses in a vector, and after all of the calculations are complete show the first few items in the vector using `head()`.
- Add the results in the vector back to the original data frame. Show the first few rows of the data frame using `head()`.
- Calculate the mean mass for each `species` using `dplyr`.

```
mass_from_length <- function(length, dino_group) {
  if (dino_group == 'Stegosauria') {
    mass <- 10.95 * length ^ 2.64
  } else if (dino_group == 'Theropoda') {
    mass <- 0.73 * length ^ 3.63
  }
}
```

```

    } else if (dino_group == 'Sauropoda') {
      mass <- 214.44 * length ^ 1.46
    } else {
      mass <- 25.37 * length ^ 2.49
    }
    return(mass)
  }
}

dino_data <- read_csv("dinosaur_lengths.csv")

```

```

## Rows: 500 Columns: 2
## -- Column specification -----
## Delimiter: ","
## chr (1): species
## dbl (1): lengths
##
## i Use 'spec()' to retrieve the full column specification for this data.
## i Specify the column types or set 'show_col_types = FALSE' to quiet this message.

```

```

# 2a
print("2a")

```

```
## [1] "2a"
```

```

lengths <- dino_data$lengths
species <- dino_data$species
masses <- vector(length = length(lengths))
for (i in 1:length(lengths)){
  masses[i] <- mass_from_length(lengths[i], species[i])
}
head(masses)

```

```
## [1] 24341.68 27017.90 67453.38 22114.19 53884.76 52026.34
```

```

# 2b
print("2b")

```

```
## [1] "2b"
```

```

dino_data$masses = masses
head(dino_data)

```

```

## # A tibble: 6 x 3
##   species      lengths masses
##   <chr>         <dbl>   <dbl>
## 1 Stegosauria    18.5 24342.
## 2 Ankylosauria   16.4 27018.
## 3 Ankylosauria   23.7 67453.
## 4 Sauropoda      23.9 22114.
## 5 Ankylosauria   21.7 53885.
## 6 Ankylosauria   21.4 52026.

```

```
# 2c
print("2c")
```

```
## [1] "2c"
```

```
dino_data %>%
  group_by(species) %>%
  summarize(avg_mass = mean(masses))
```

```
## # A tibble: 4 x 2
##   species      avg_mass
##   <chr>         <dbl>
## 1 Ankylosauria  46819.
## 2 Sauropoda    16104.
## 3 Stegosauria  31924.
## 4 Theropoda    45572.
```

3. Multi-file Analysis (40 pts)

You have satellite collars on a number of different individuals and want to be able to quickly look at all of their recent movements at once. The data is posted daily to a URL that contains one csv file for each individual: zip file

Start your solution by:

- If `individual-collar_data.zip` is not already in your working directory, download it from the link above and place the zip file in the correct directory.
 - Unzip it using `unzip()`
 - Obtain a list of all of the files with file names matching the pattern `"collar-data-*.txt"` (using `list.files()`)
- a. Use a loop to load each of these files into R and make a line plot (using `geom_path()`) for each file with `long` on the x axis and `lat` on the y axis.

Graphs, like other types of output, won't display inside a loop unless you explicitly display them, so you need put your `ggplot()` command inside a `print()` statement.

Include the name of the file in the graph as the graph title using the `ggtitle()`.

- b. Add code to the loop to calculate the minimum and maximum latitude in the file, and store these values, along with the name of the file, in a data frame.

Show the data frame as output.

```
library(ggplot2)

download.file("http://www.datacarpentry.org/semester-biology/data/individual-collar_data.zip",
             "individual-collar_data.zip")
unzip("individual-collar_data.zip")
collar_data_files = list.files(pattern = "collar-data-*.txt",
                              full.names = TRUE)
```

```

num_data_files = length(collared_data_files)
output = data.frame(file_name = character(num_data_files),
                    max_lat = numeric(num_data_files),
                    min_lat = numeric(num_data_files),
                    observations = numeric(num_data_files),
                    stringsAsFactors = FALSE)

print("3a")

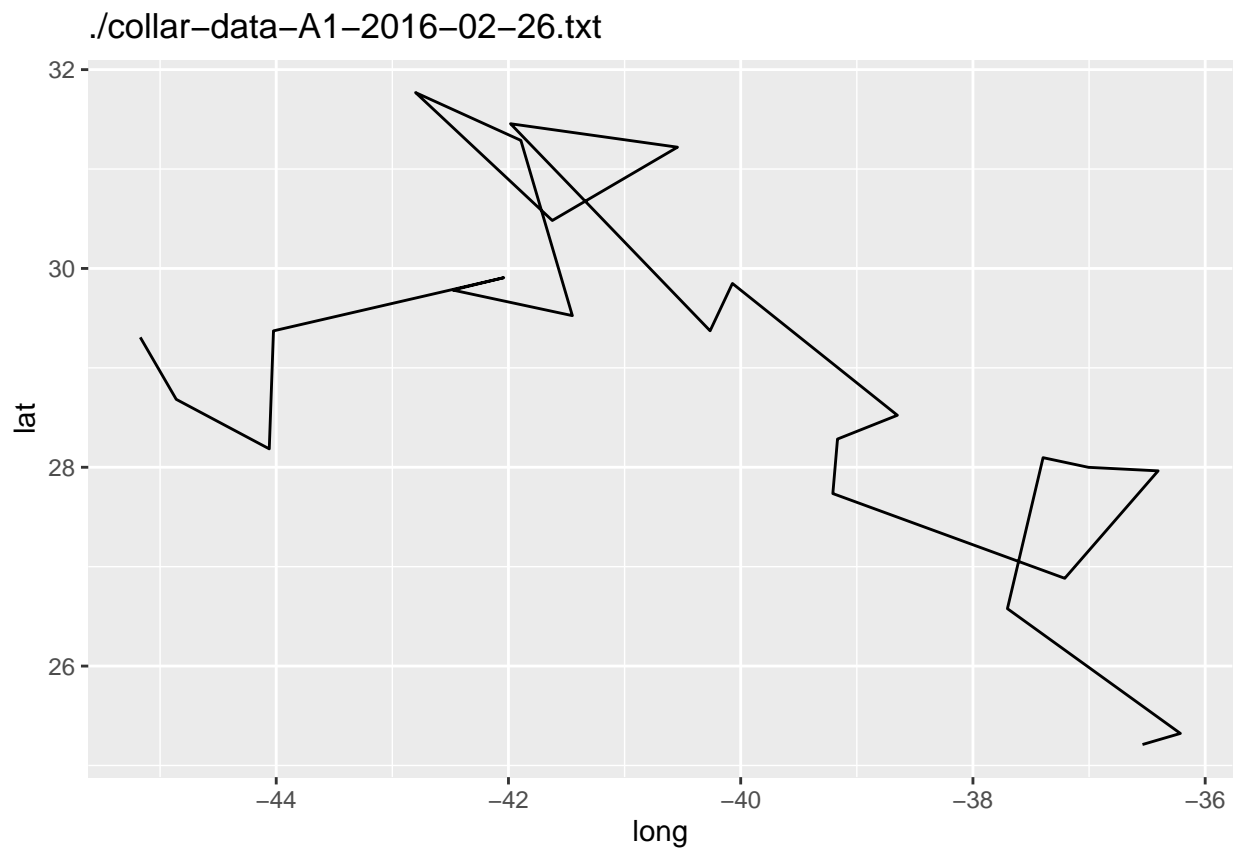
```

```
## [1] "3a"
```

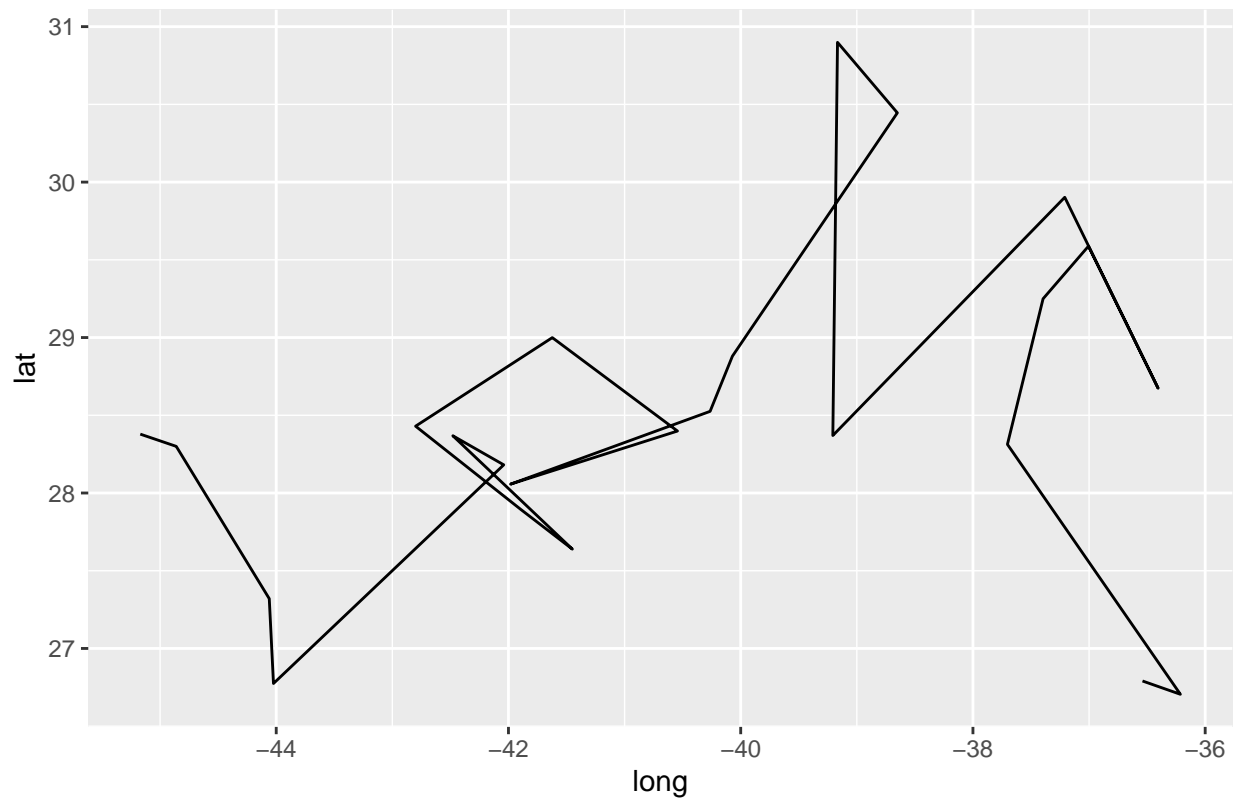
```

for (i in 1:length(collared_data_files)){
  data = read.csv(collared_data_files[i])
  p <- ggplot(data, aes(x = long, y = lat)) + geom_path() + labs(title = collared_data_files[i])
  print(p)
}

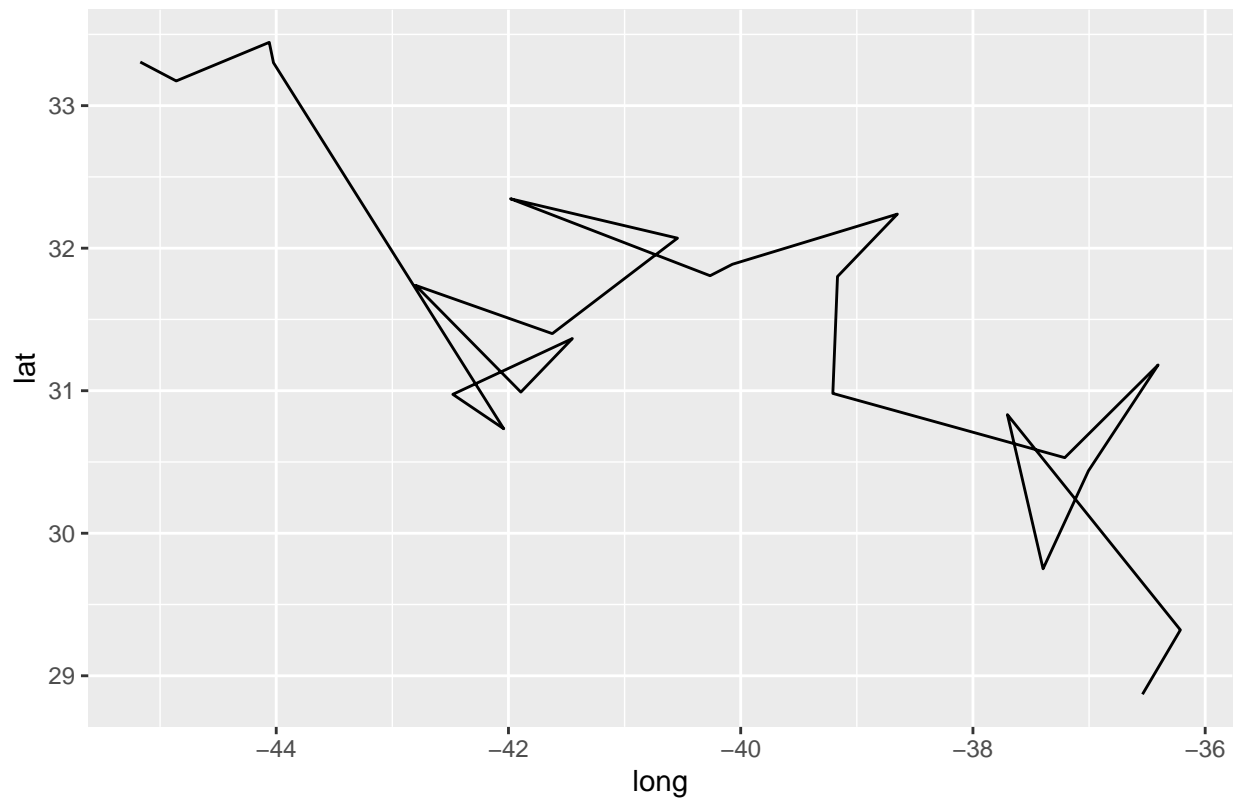
```



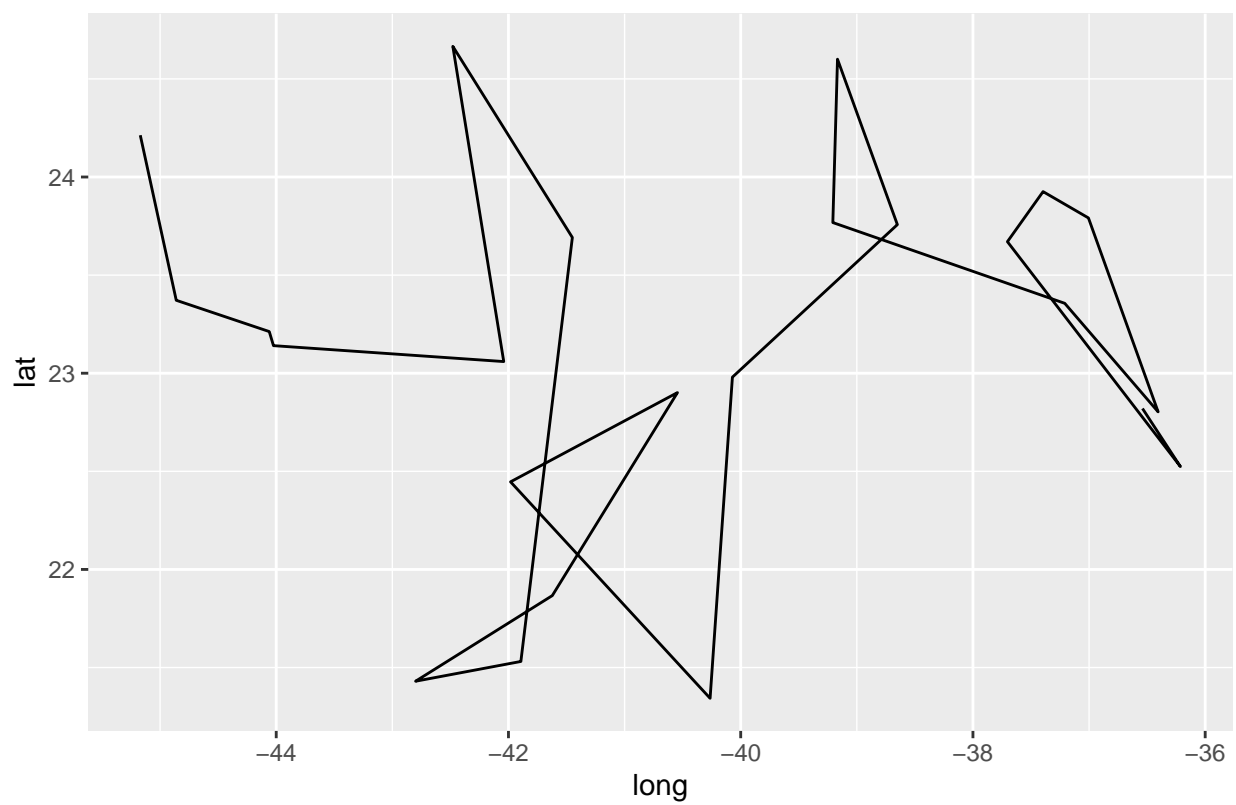
./collar-data-B2-2016-02-26.txt



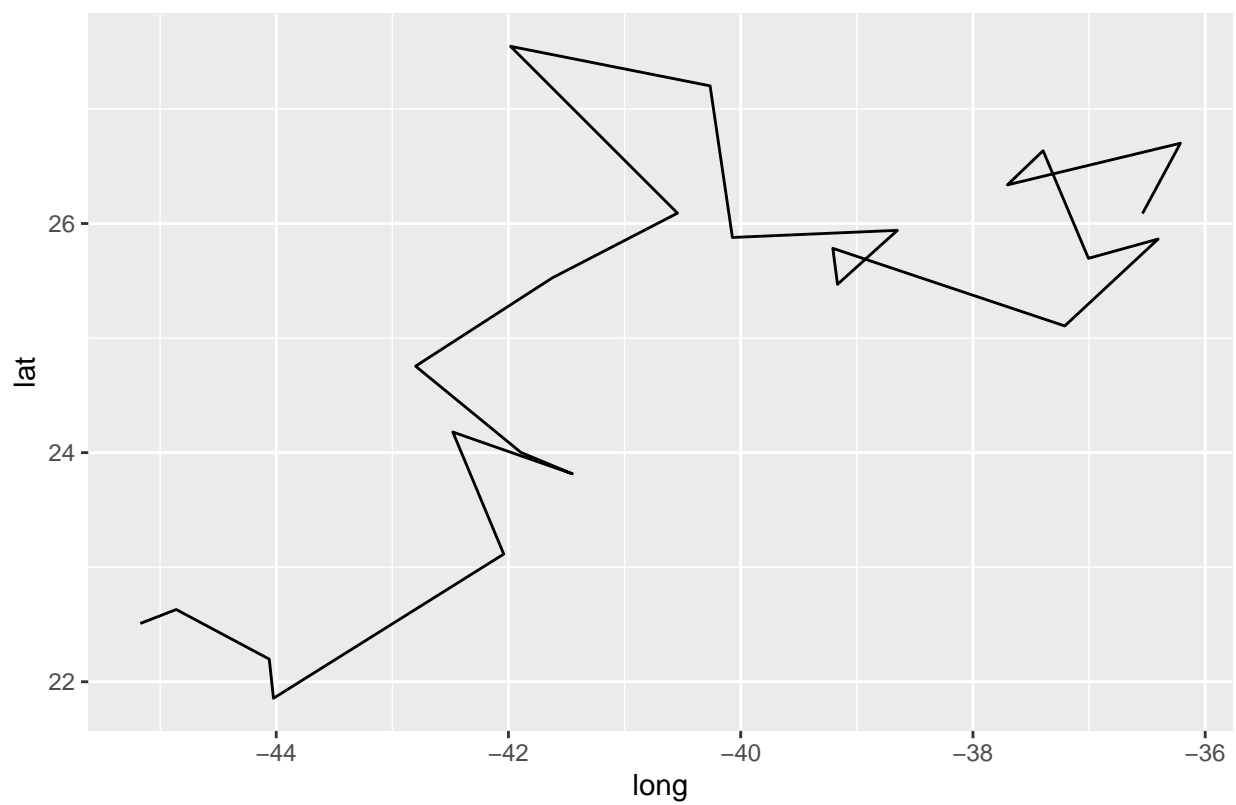
./collar-data-C3-2016-02-26.txt



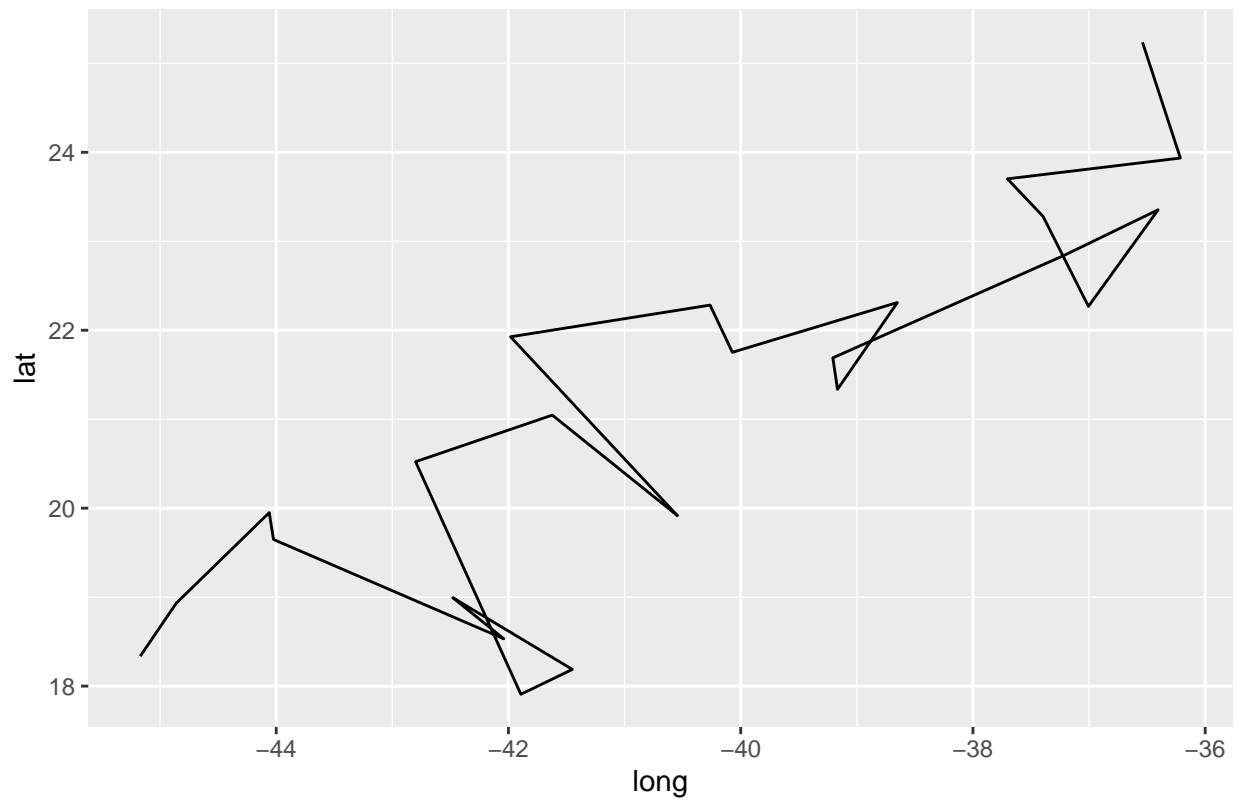
./collar-data-D4-2016-02-26.txt



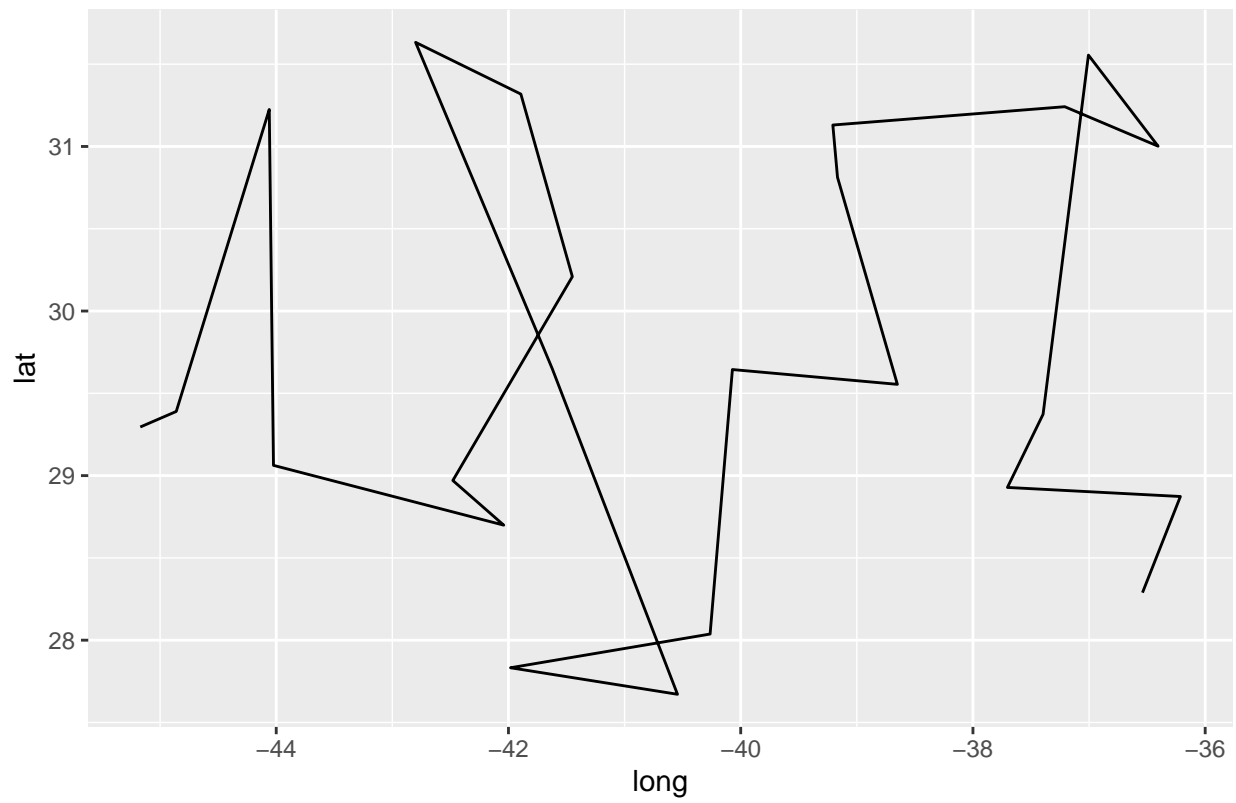
./collar-data-E5-2016-02-26.txt



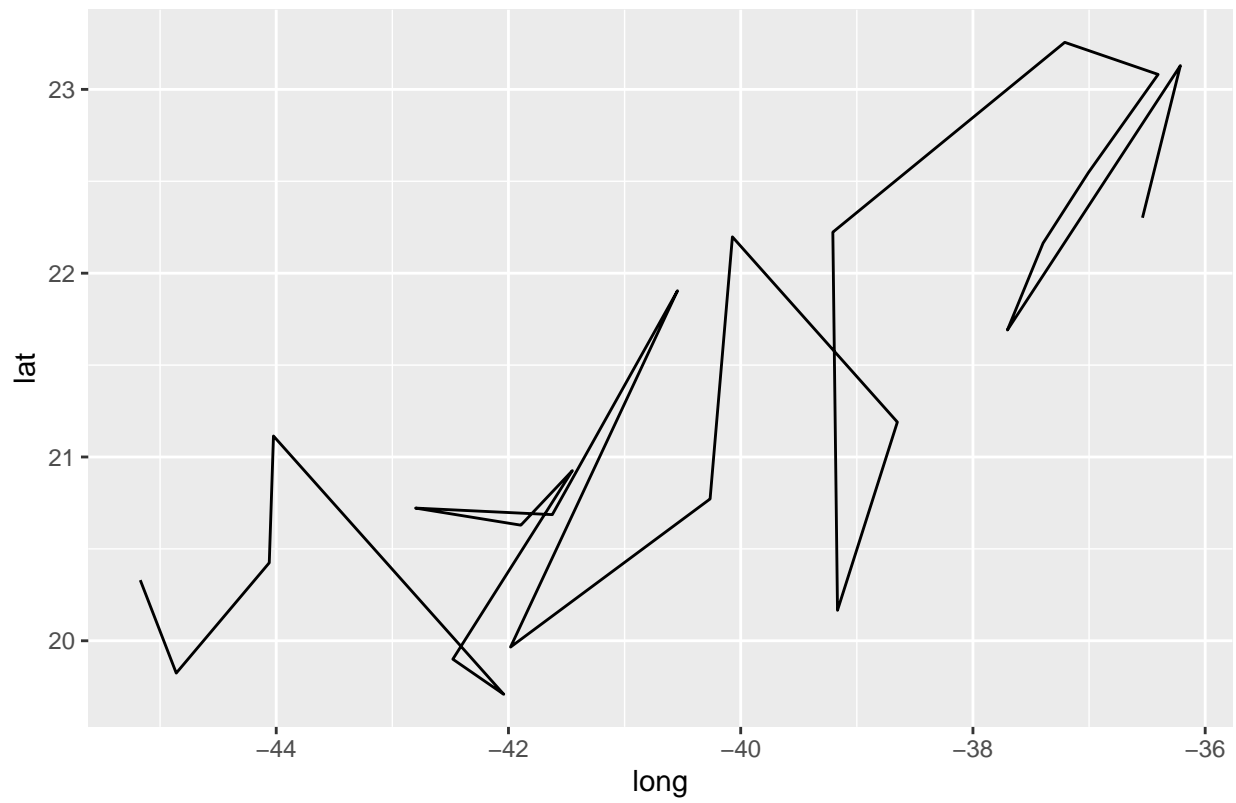
./collar-data-F6-2016-02-26.txt



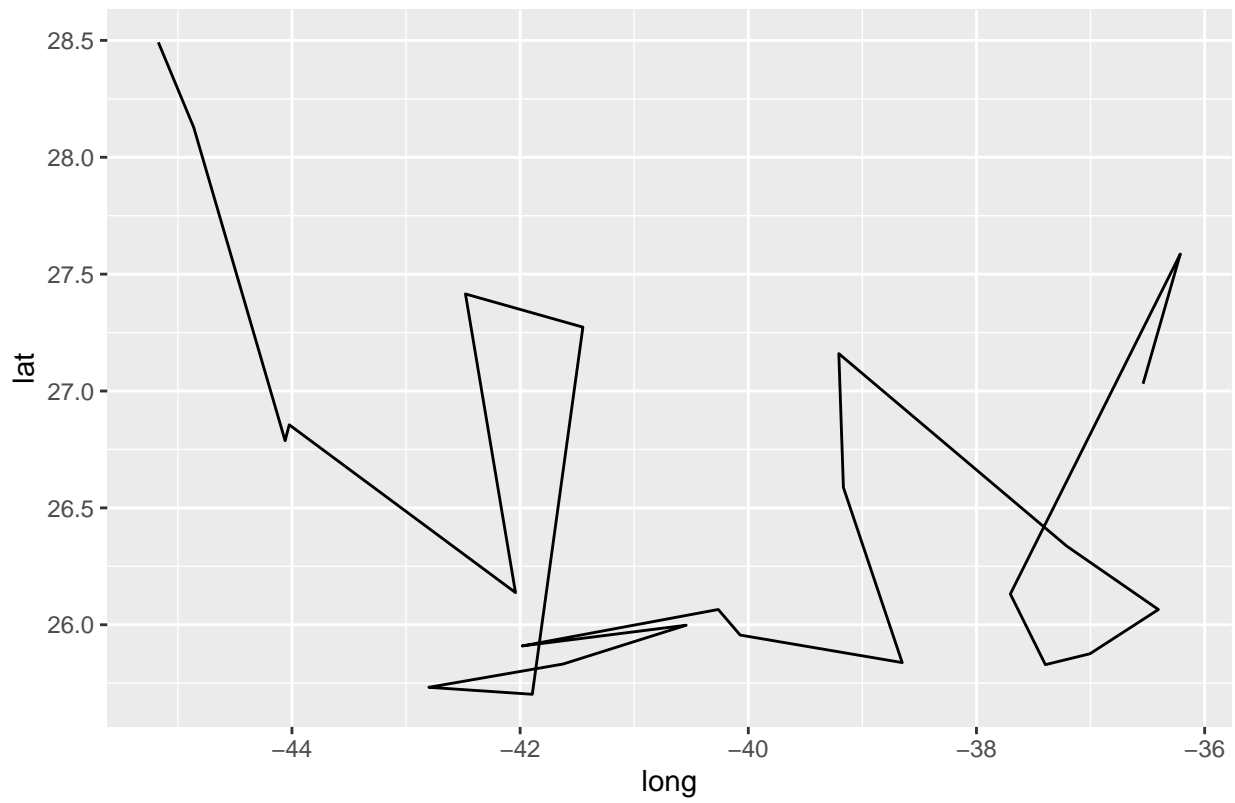
./collar-data-G7-2016-02-26.txt



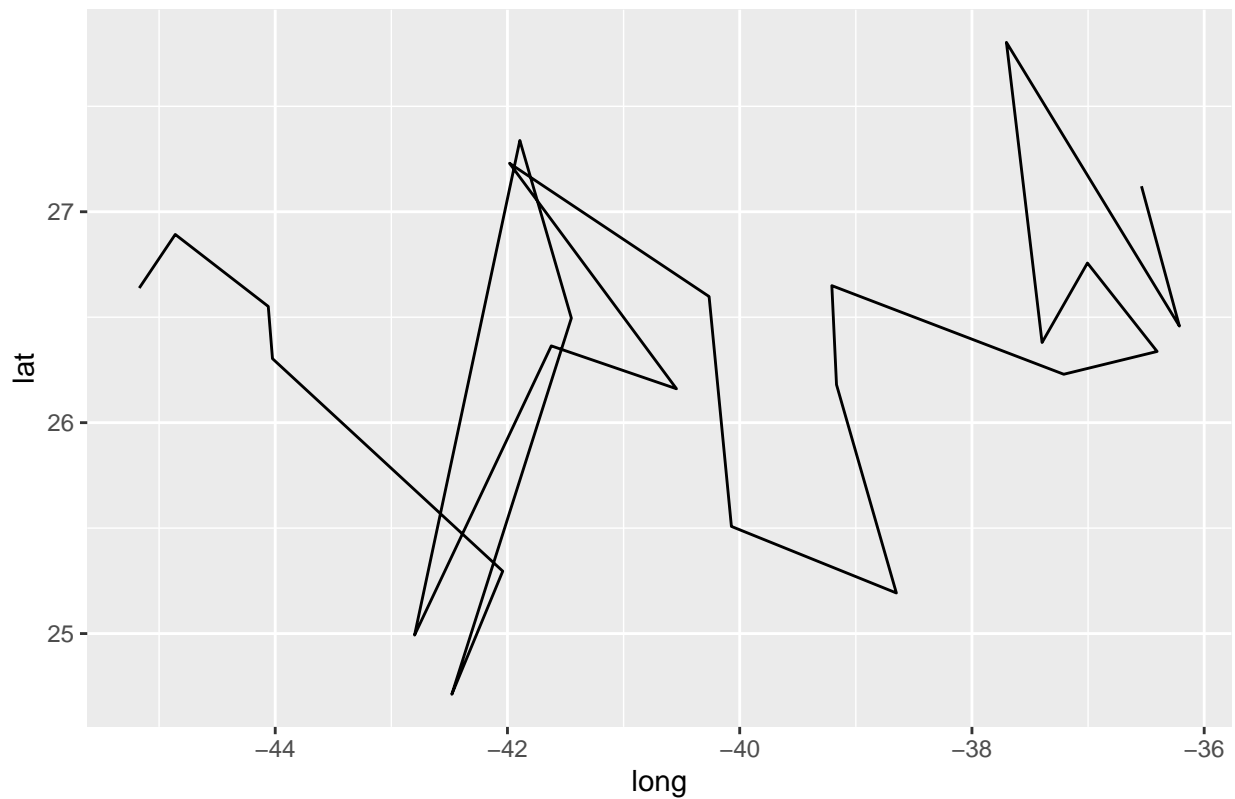
./collar-data-H8-2016-02-26.txt



./collar-data-l9-2016-02-26.txt



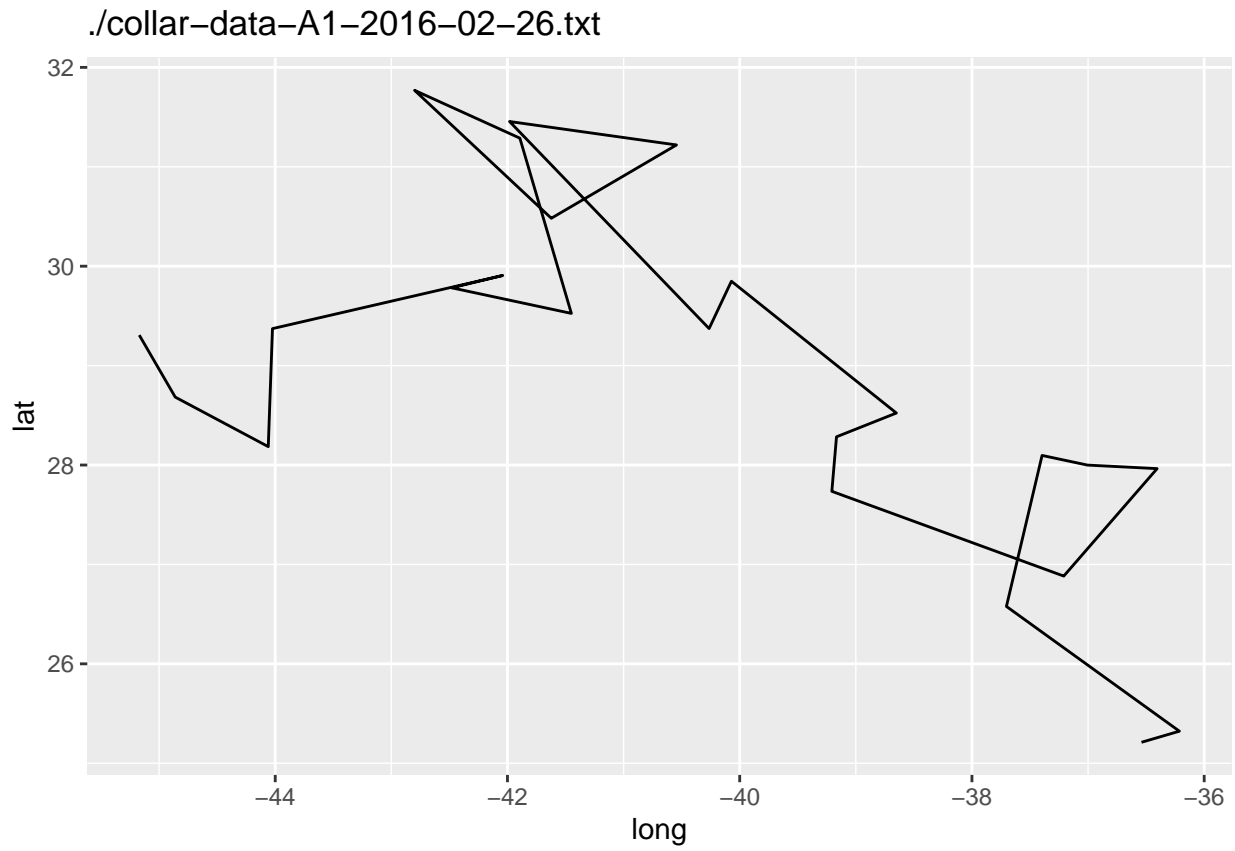
./collar-data-J10-2016-02-26.txt



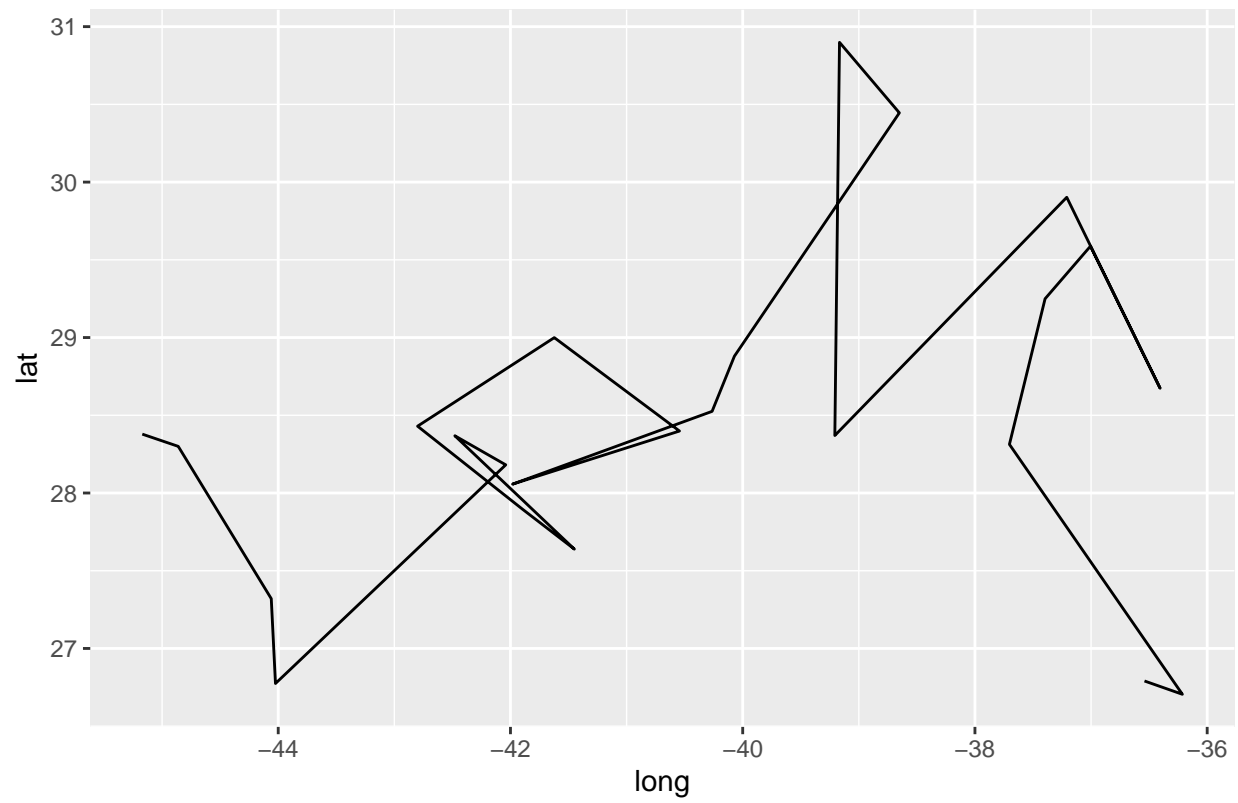
```
print("3b")
```

```
## [1] "3b"
```

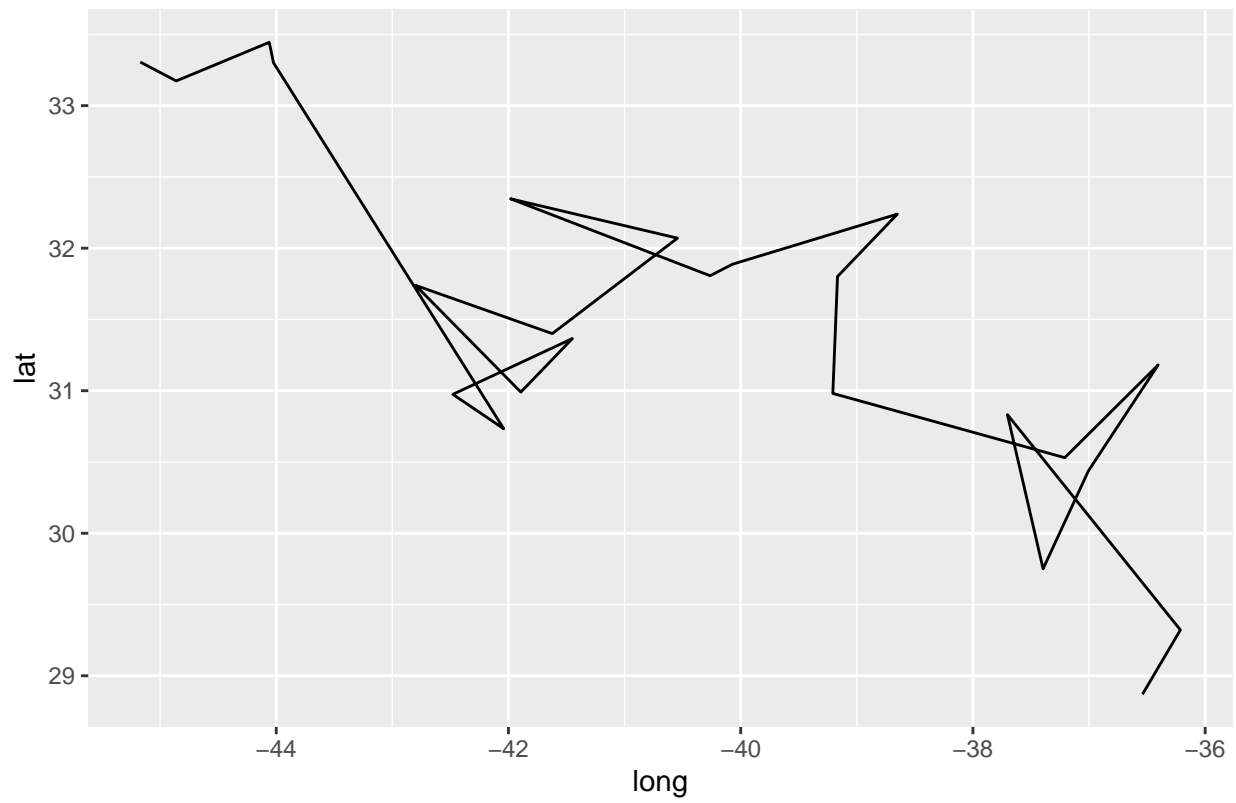
```
for (i in 1:length(collar_data_files)){  
  data = read.csv(collar_data_files[i])  
  p <- ggplot(data, aes(x = long, y = lat)) + geom_path() + labs(title = collar_data_files[i])  
  print(p)  
  output$file_name[i] = collar_data_files[i]  
  output$max_lat[i] = max(data$lat)  
  output$min_lat[i] = min(data$lat)  
  output$observations[i] = nrow(data)  
}
```

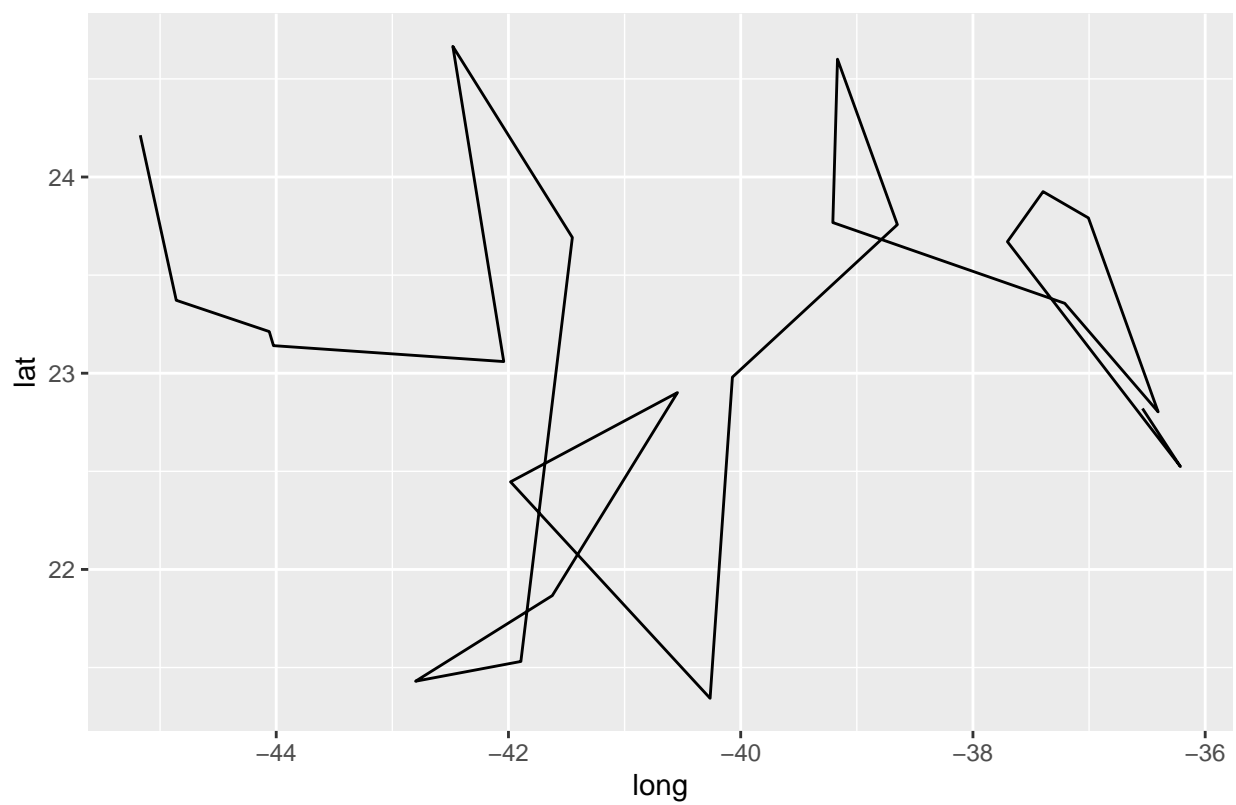
./collar-data-B2-2016-02-26.txt



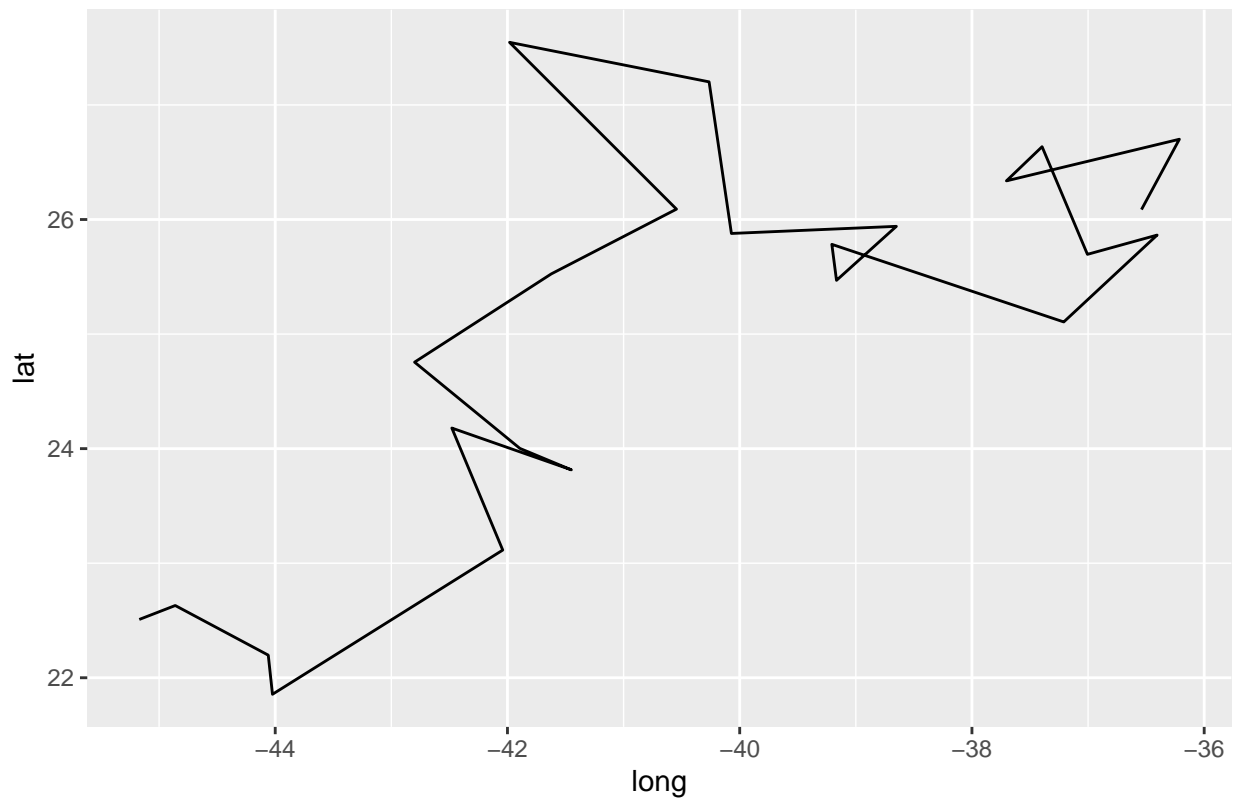
./collar-data-C3-2016-02-26.txt



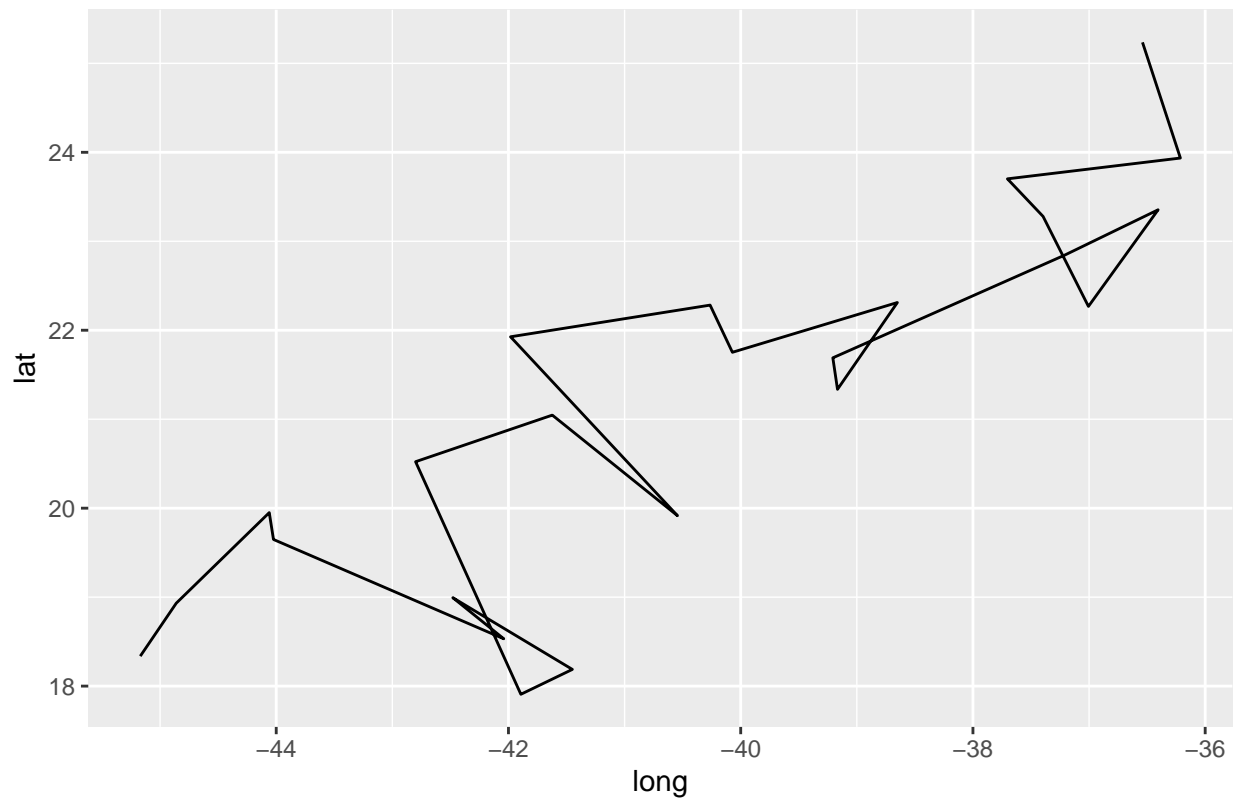
./collar-data-D4-2016-02-26.txt



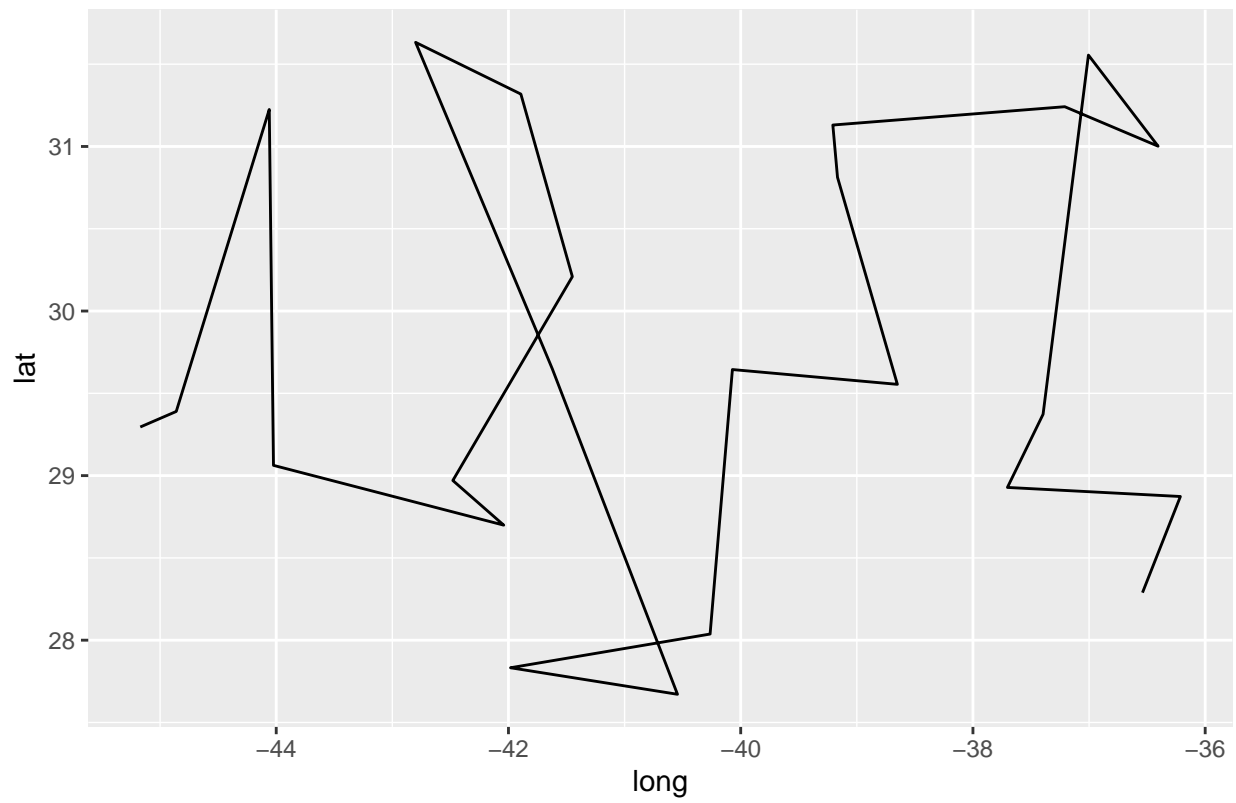
./collar-data-E5-2016-02-26.txt



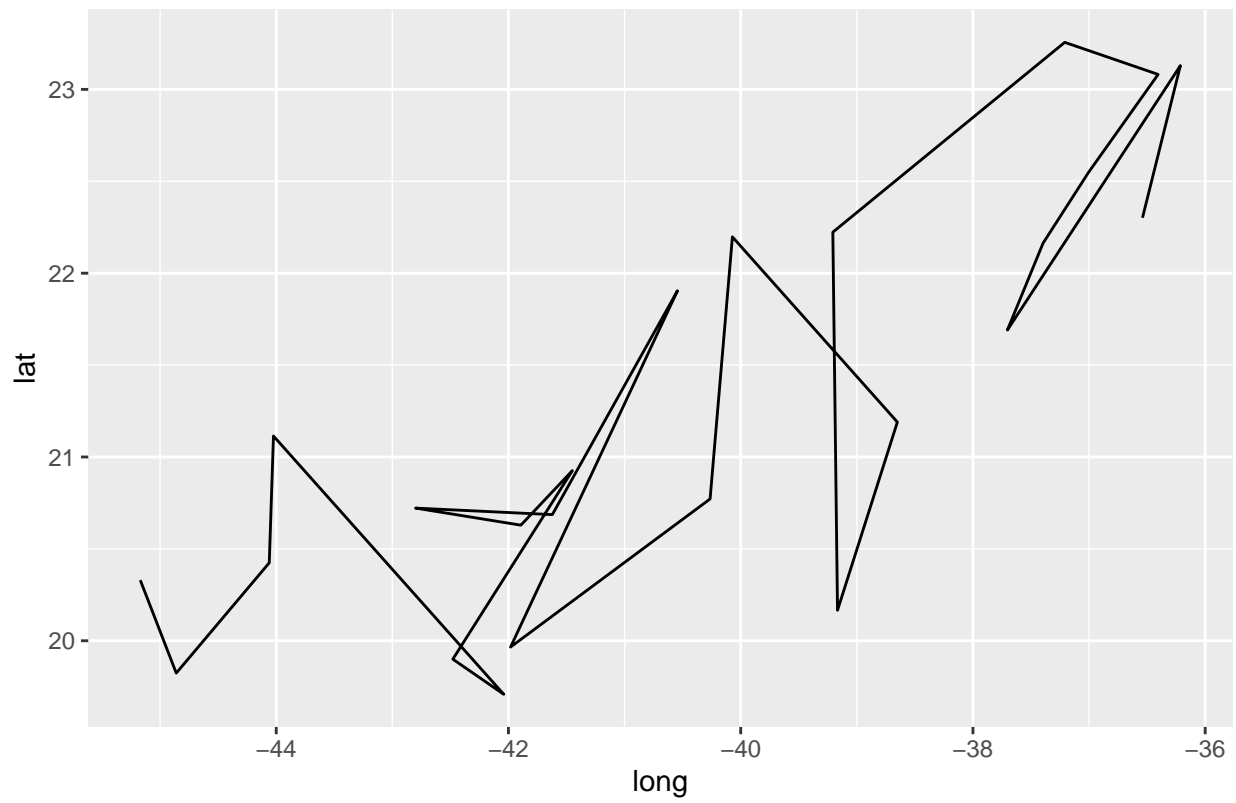
./collar-data-F6-2016-02-26.txt



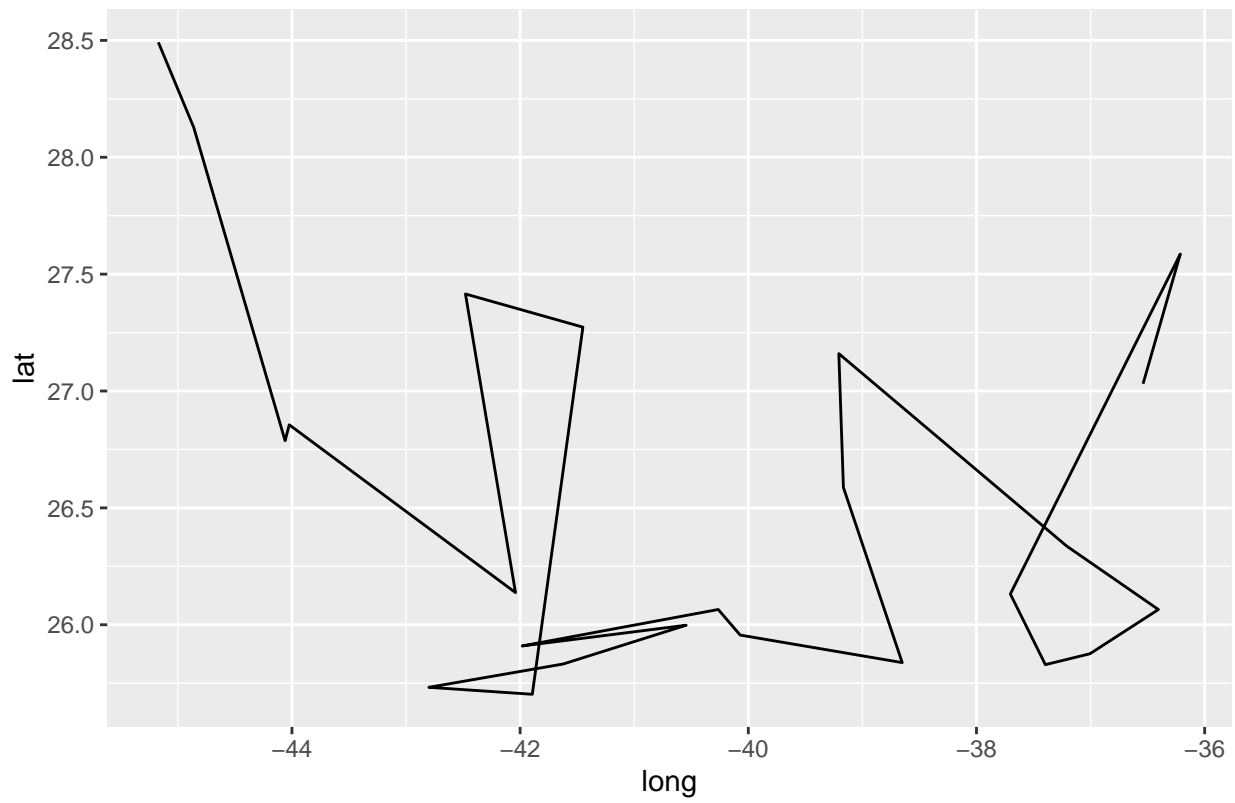
./collar-data-G7-2016-02-26.txt



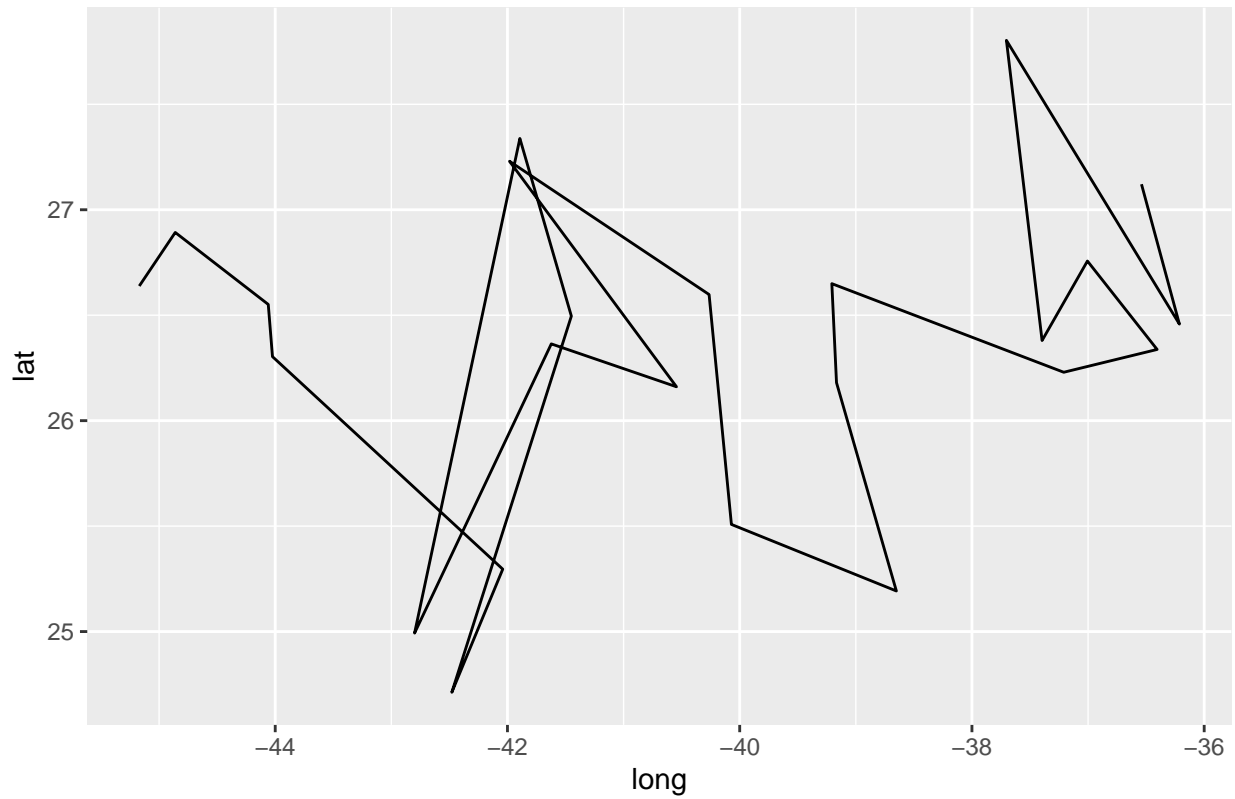
./collar-data-H8-2016-02-26.txt



./collar-data-l9-2016-02-26.txt



./collar-data-J10-2016-02-26.txt



output

##	file_name	max_lat	min_lat	observations
## 1	./collar-data-A1-2016-02-26.txt	31.76912	25.21080	24
## 2	./collar-data-B2-2016-02-26.txt	30.89907	26.70509	24
## 3	./collar-data-C3-2016-02-26.txt	33.44421	28.86998	24
## 4	./collar-data-D4-2016-02-26.txt	24.66598	21.34315	24
## 5	./collar-data-E5-2016-02-26.txt	27.54663	21.85565	24
## 6	./collar-data-F6-2016-02-26.txt	25.23623	17.90788	24
## 7	./collar-data-G7-2016-02-26.txt	31.63272	27.67120	24
## 8	./collar-data-H8-2016-02-26.txt	23.25601	19.70875	24
## 9	./collar-data-I9-2016-02-26.txt	28.49172	25.70252	24
## 10	./collar-data-J10-2016-02-26.txt	27.80325	24.71200	24