PROGRAMMING WITH R, PART 1

Dartmouth College | Research Computing

OVERVIEW

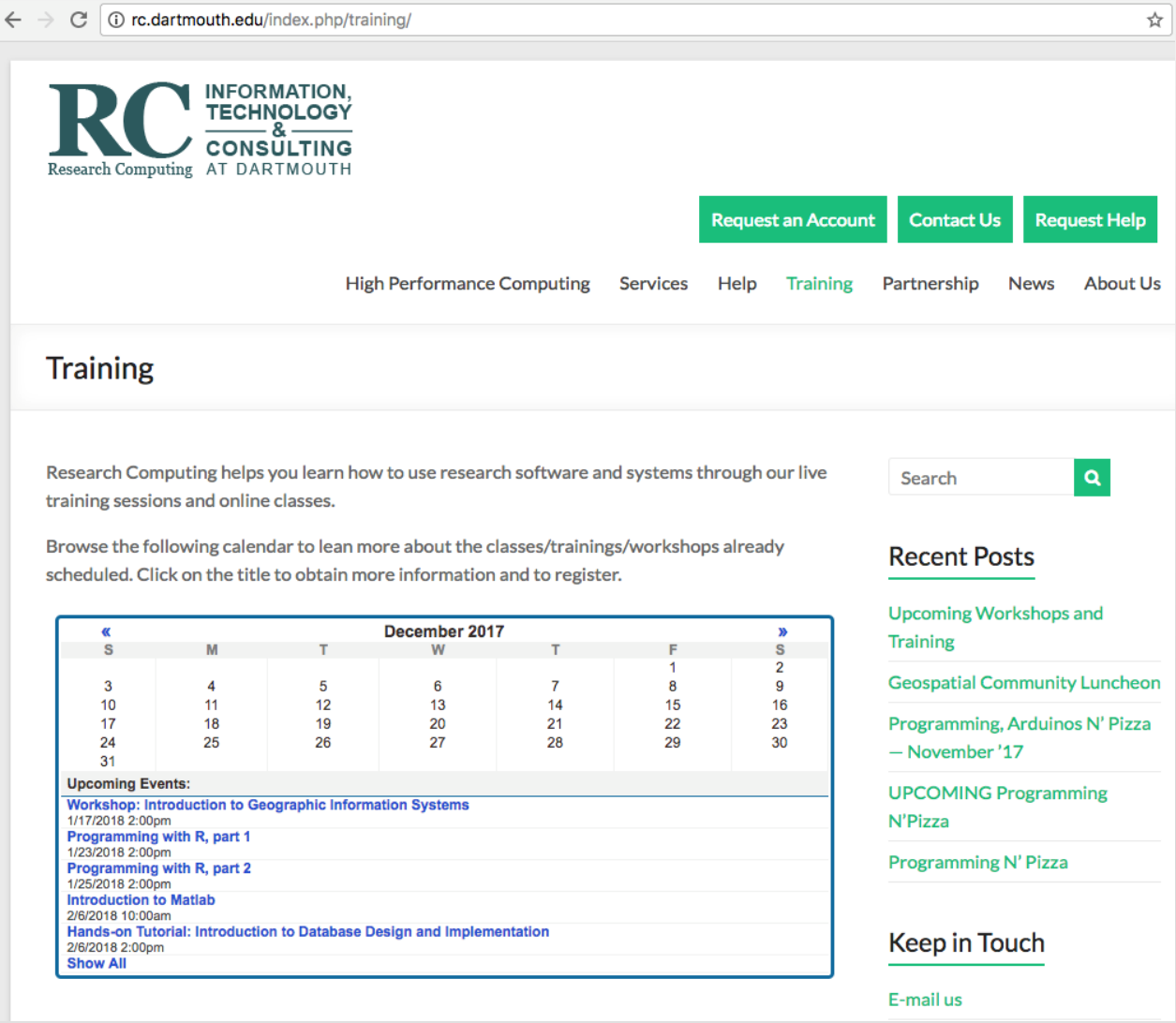
- Info & Workshops
- What is R?
- R as a programming language
- The R Console
- R Studio
- Data Management
- Packages to extend R
- Hands-on programming in R!

INFO & WORKSHOPS (I)

- Data Visualization using R
  - James Adams, Baker-Berry Library, James.L.Adams@dartmouth.edu
- Statistical Consulting (R, Stata, SAS)
  - Jianjun Hua from Ed Tech provides consulting support for statistics-related questions. Jianjun can be contacted at 603-646-6552 or by emailing jianjun.hua@dartmouth.edu
- R for High Performance Computing, parallel computing, GIS
  - Research.computing@Dartmouth.edu and http://rc.dartmouth.edu/
- R Club
  - Katja Koeppen, Microbiology Department organizes an R Club, Katja.Koeppen@Dartmouth.edu
- Programming n' Pizza #4 – Tonight 6:30pm – 8:30pm, Carson 61
  - http://rc.dartmouth.edu/index.php/programming-n-pizza/
- Departmental Courses at Dartmouth, Math, Quantitative Social Sciences, etc
  - Math 10, Math 50 https://math.dartmouth.edu/courses/by-term/ , http://qss.dartmouth.edu/
  - Math 10, Online Stats book "Online Statistics Education: A Multimedia Course of Study" (http://onlinestatbook.com/). David M. Lane, Rice University.

INFO & WORKSHOPS (II)

- RC Workshops, Training and Consulting
  - GIS
  - Python
  - Matlab
  - Database Design for Research
- High-Performance Computing
  - Using the high-performance compute resources at Dartmouth
- http://rc.dartmouth.edu/



INFO & WORKSHOPS (III)

R Club

Discover
- new ways to visualize complex data
- new analysis approaches
- cool R packages

Share
- R programming tricks
- your problems and solutions

Develop
- R fluency
- scientific expertise
- collaborations

Mondays at Noon in Vail 513

Contact: Katja.Koeppen@Dartmouth.edu

WHAT IS R?

- R is a free software environment used for computing, graphics and statistics. It comes with a robust programming environment that includes tools for data analysis, data visualization, statistics, high-performance computing and geographic analysis. Visit https://www.r-project.org/ for more
- R has been around for more than 20 years and it has become popular at universities, research labs and federal and state government offices in the last ten years for many applications
- R consists of base packages but also includes hundreds of add-on packages that greatly extend the capabilities of the programming environment.
- These capabilities include data manipulation, data visualization, statistics, geographic data tools, web publishing, workflow

R AS A PROGRAMMING LANGUAGE (I)

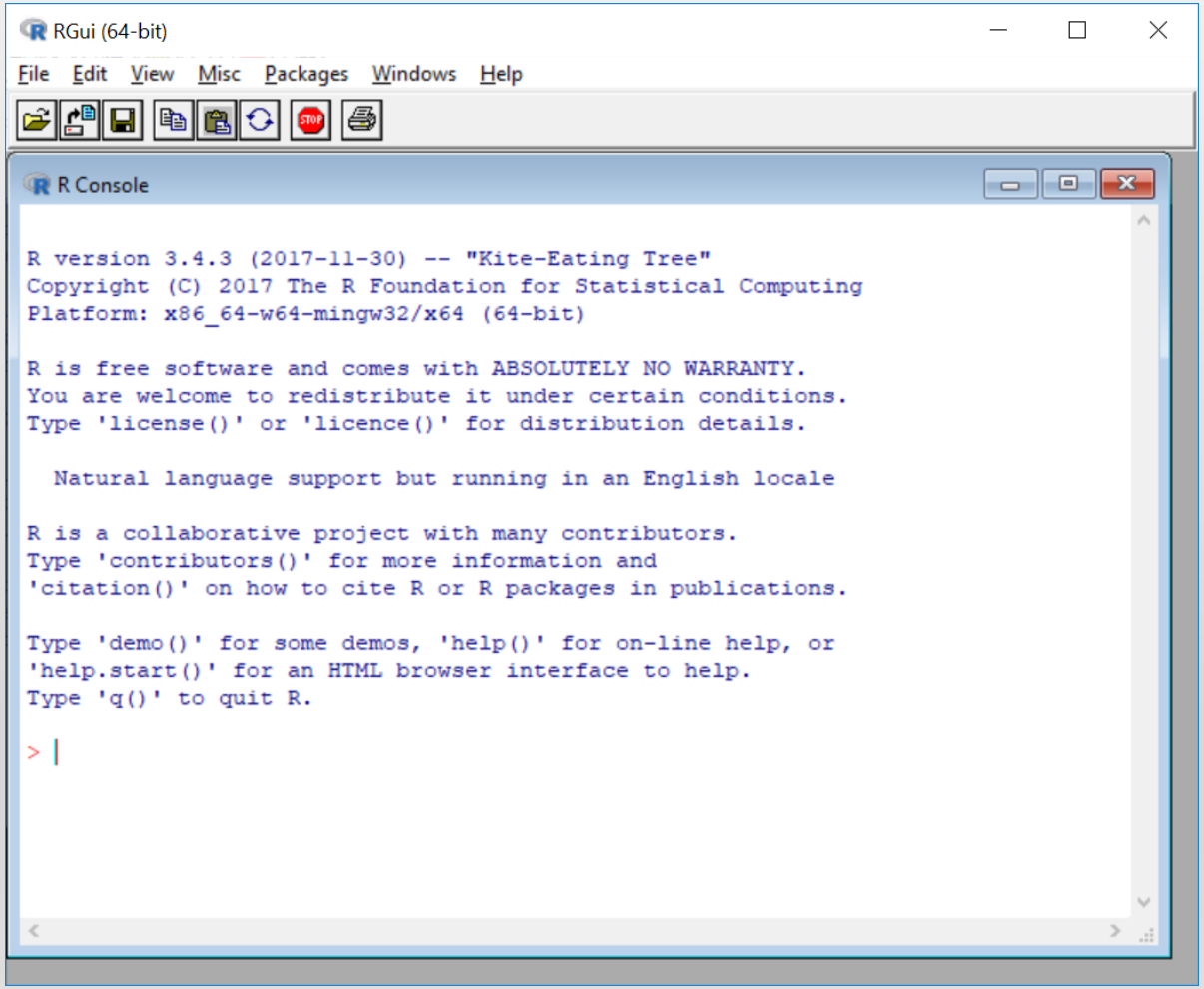
- The R project has a robust programming language that includes basic and advanced programming tools.
- R has the ability to load and manipulate many types of data
- R has programming constructs such as loops and conditionals
- R code can be saved and run at command lines across platforms: PC, Mac and Linux, and it can be optimized to run in High-performance computing environments as well.
- R has packages that allow for connection to databases, advanced visualization, web app design, and more

R AS A PROGRAMMING LANGUAGE (II)

- To work and program effectively with R, we will cover some of the basics:
  - Declaring variables
  - Importing data
  - Running basic functions
  - Creating new functions
  - Programming loops and conditionals
  - Working with Data Frames
  - Extracting data by rows and columns

THE R CONSOLE

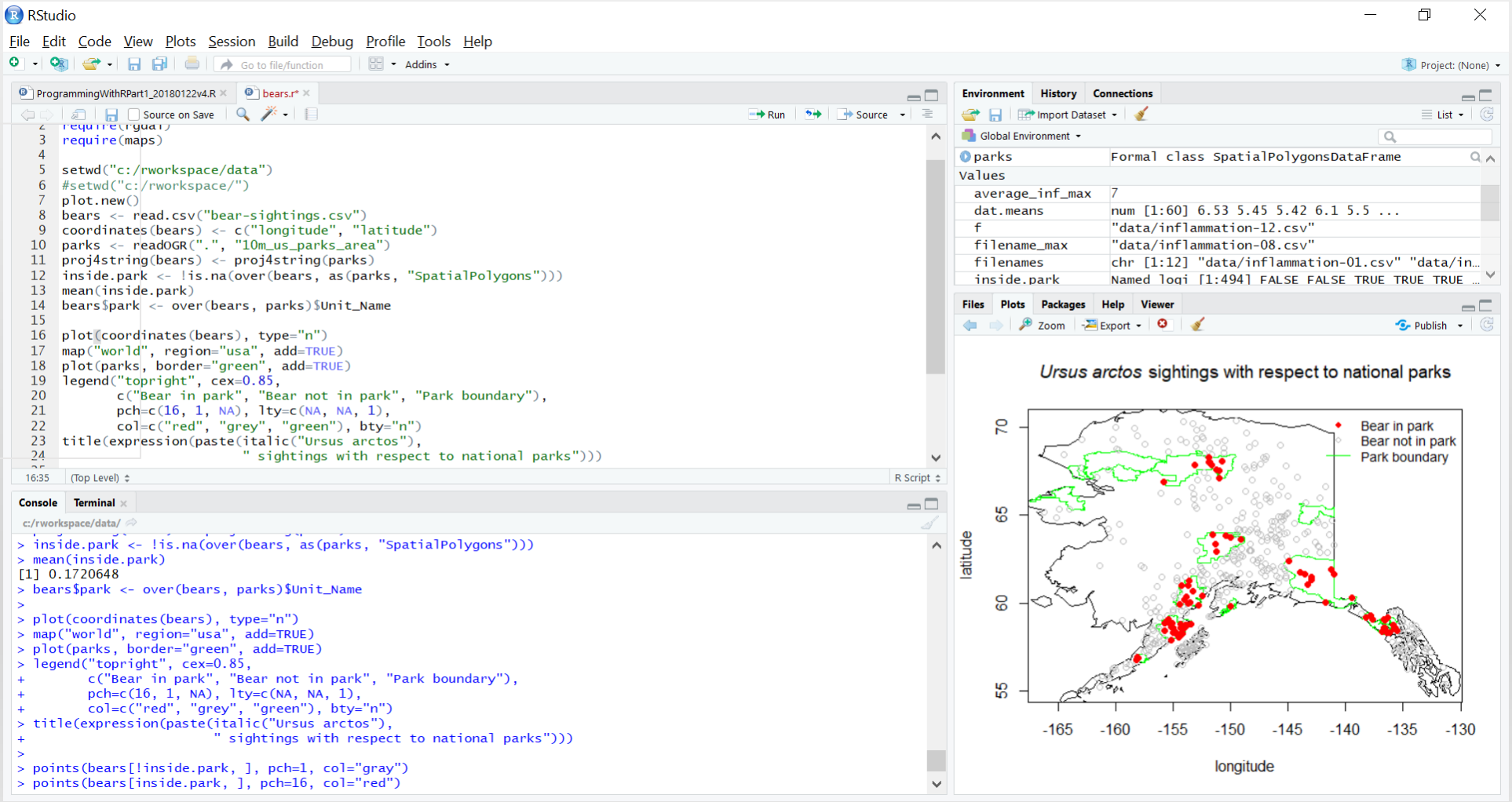
- The R console is a quick, light, multiplatform install



THE R STUDIO IDE

- RStudio is an interactive development environment

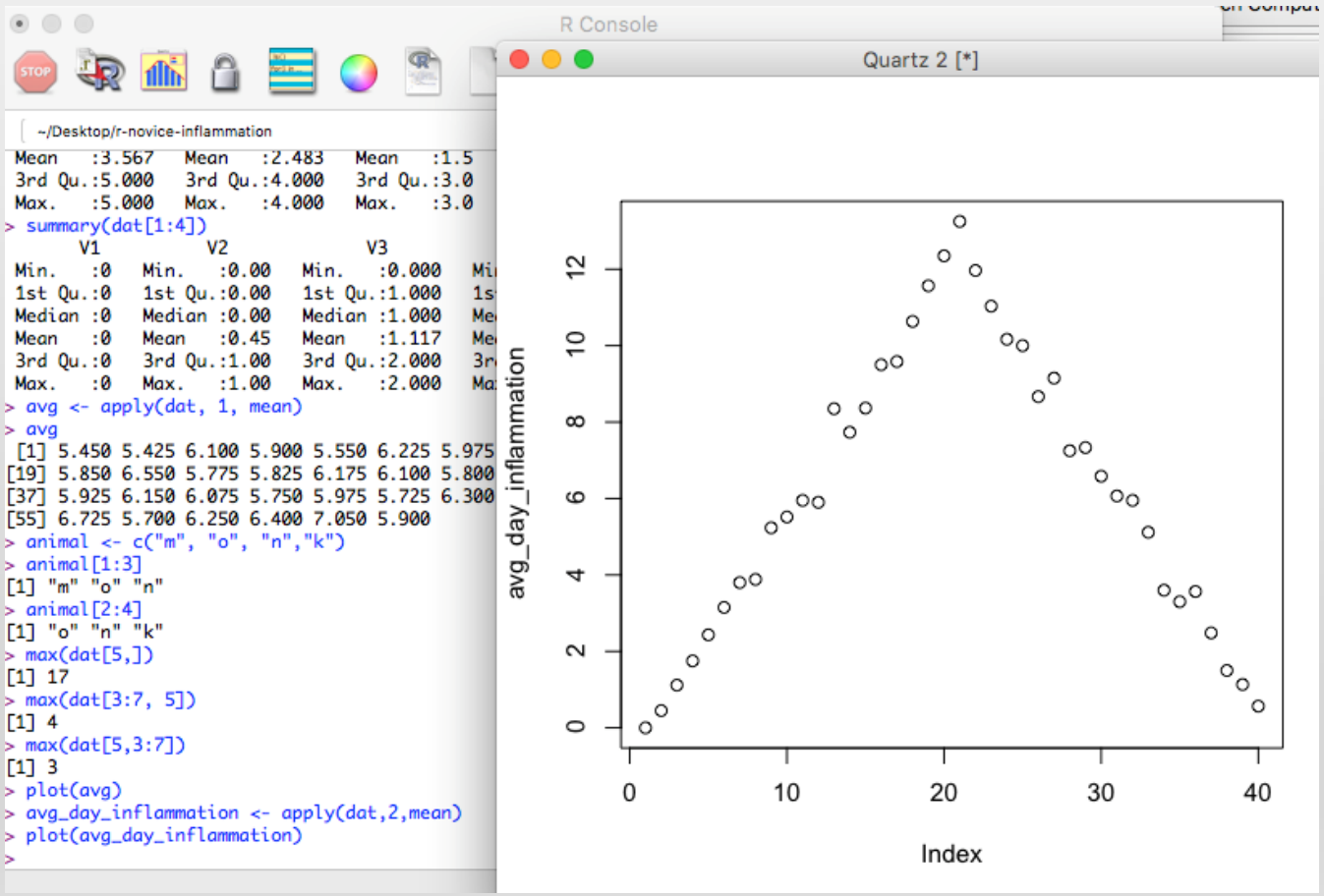
- Console
- Terminal
- Script Editor
- Variables
- Plots, Graphics
- Exports
- Package import

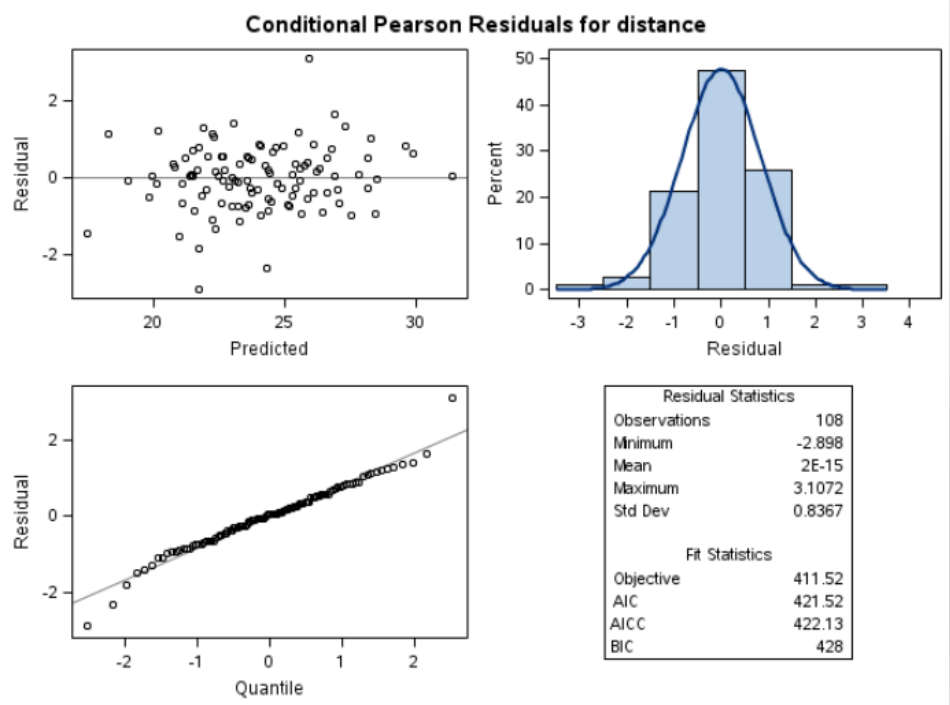


SOME PACKAGES TO EXTEND R

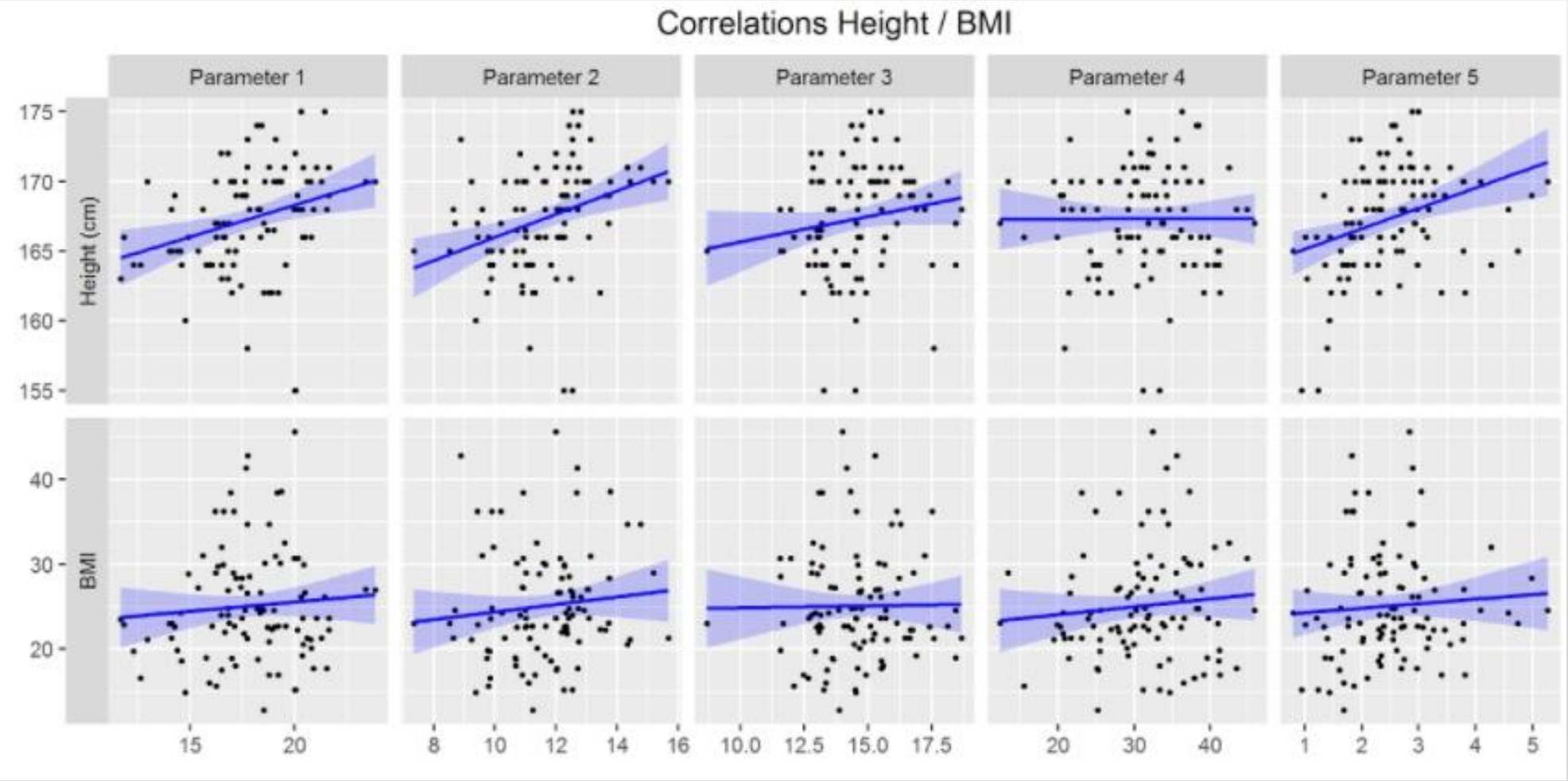
- https://support.rstudio.com/hc/en-us/articles/201057987-Quick-list-of-useful-R-packages
- Tidyr
- Ggplot2
- Dpylr
- xlsx
- Maps
- Sp
- Rgdal
- Parallel

WHAT CAN R DO?
SCATTER PLOTS, HISTOGRAMS, RESIDUALS





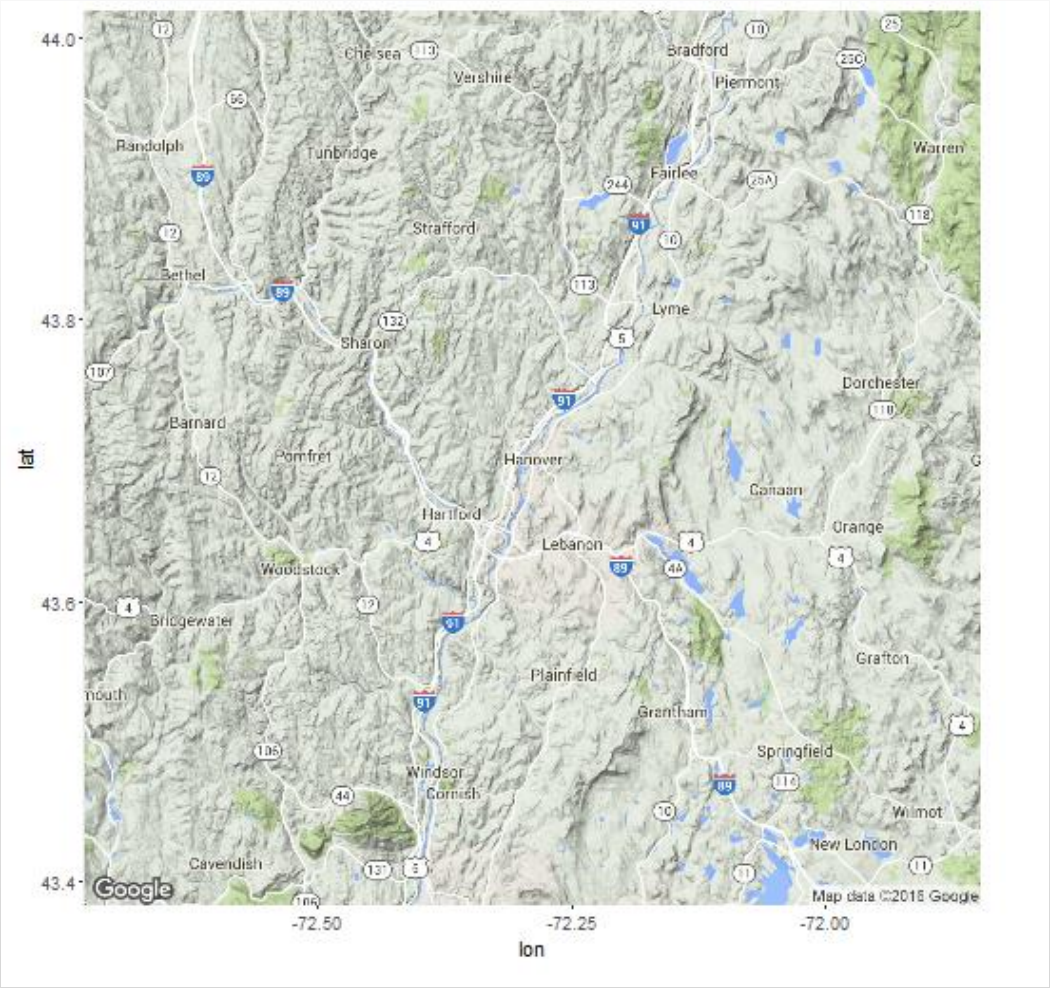
WHAT CAN R DO?
CORRELATION GRAPHS



WHAT CAN R DO? MAP EXAMPLE

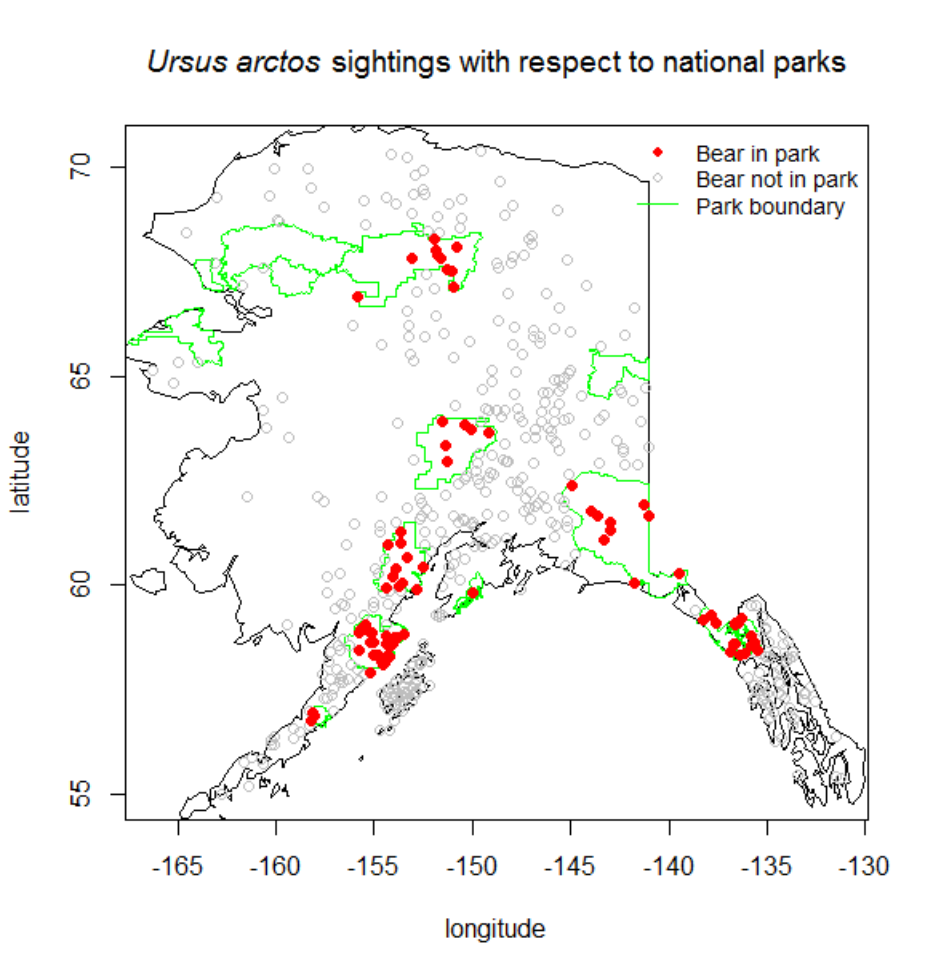
- Put a Google base map right in your plot window, overlay spatial data on to the map plot

```
install.packages("ggplot2")
install.packages("ggmap")
```



WHAT CAN R DO? GEOGRAPHIC INFORMATION ANALYSIS

# Map overlay & spatial statistics
# packages sp, rgdal and maps can turn your R in to a GIS: read, write and analyze spatial data, map overlay
install.packages("sp")
install.packages("rgdal")
install.packages("maps")



READY TO DIVE IN?

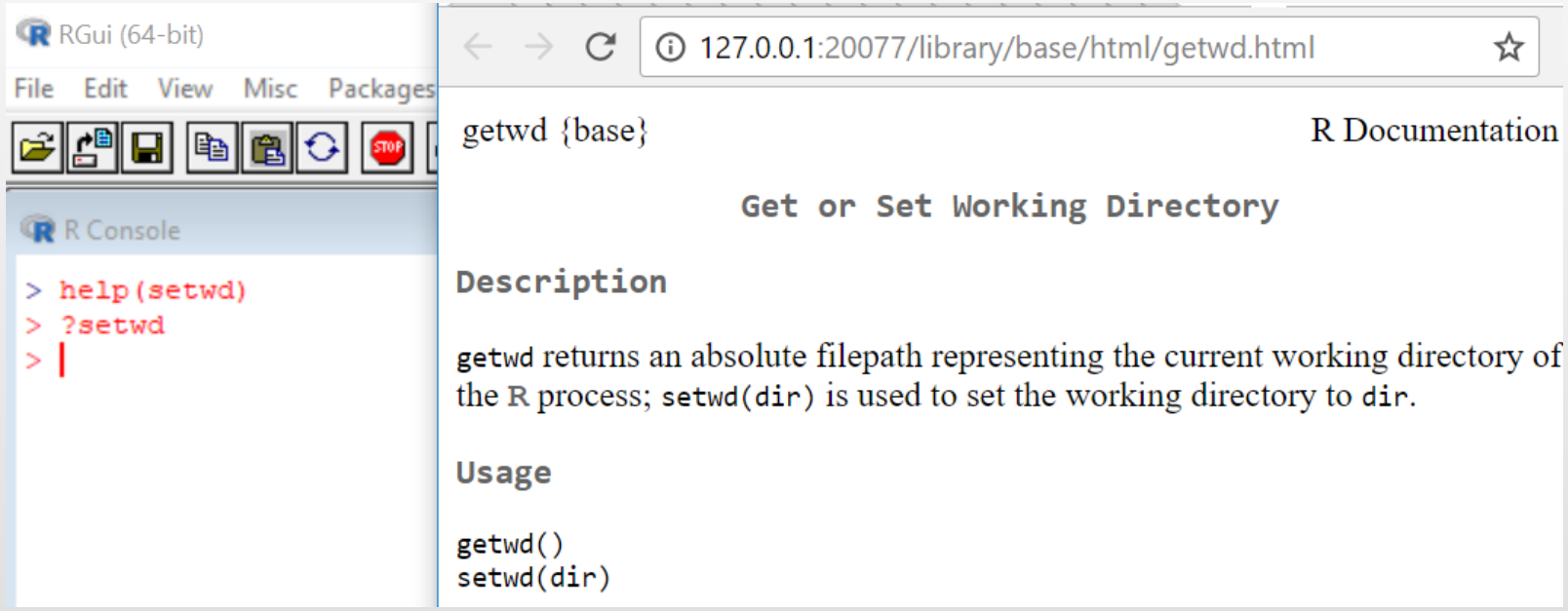
- We'll use the **R CONSOLE** today
- Data for this session can be downloaded at:

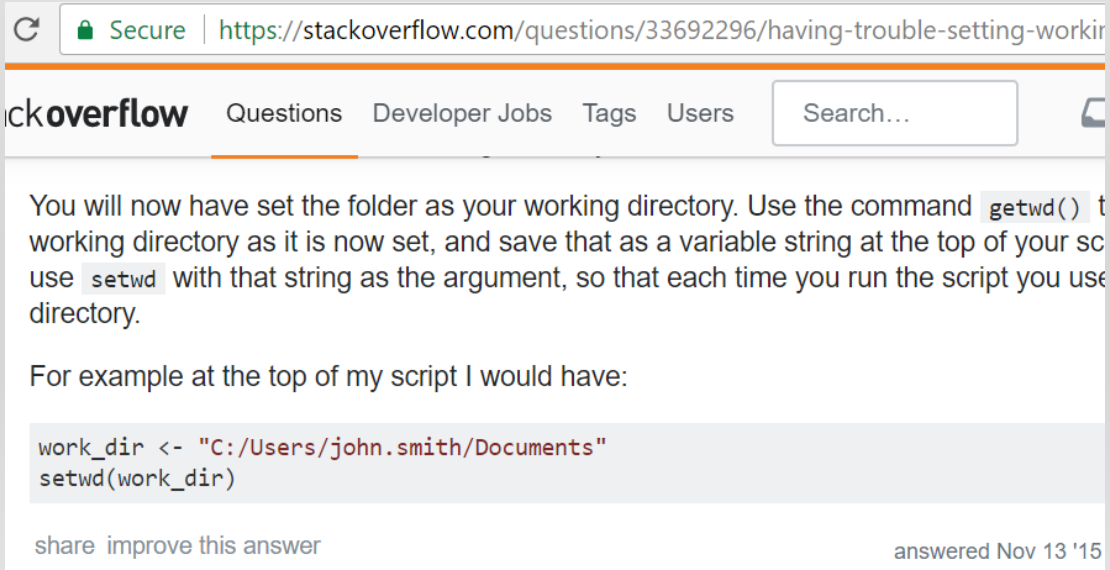
dartgo.org/programwithr1

Materials: John Blischak, Daniel Chen, Harriet Dashnow, and Denis Haine (eds): "Software Carpentry: Programming with R." Version 2016.06, June 2016, https://github.com/swcarpentry/r-novice-inflammation, 10.5281/zenodo.57541.

HELP IN R

- ?setwd
- Help(setwd)
- Web Searches
  - Google ‘r set working directory’
  - Stack Overflow ‘r set working directory stack overflow’





R AND THE R CONSOLE

- Install R and the R Console on your Mac/Win PC.
  - R: https://cran.r-project.org/
- Material:
  - http://swcarpentry.github.io/r-novice-inflammation/
- RStudio
  - RStudio (free): https://www.rstudio.com/products/rstudio/download/#download

DATA MANAGEMENT

- Data Frames
- CSV format (clean csv)
- Tidy Data
- Other formats - Reading out of databases (SQL), Geographic data constructs

QUESTIONS?

EXERCISE #1

Subsetting More Data

Suppose you want to determine the maximum inflammation for patient 5 across days three to seven. To do this you would extract the relevant subset from the data frame and calculate the maximum value. Which of the following lines of R code gives the correct answer?

1. `max(dat[5, ])`
2. `max(dat[3:7, 5])`
3. `max(dat[5, 3:7])`
4. `max(dat[5, 3, 7])`

Source: John Blischak, Daniel Chen, Harriet Dashnow, and Denis Haine (eds): "Software Carpentry: Programming with R." Version 2016.06, June 2016, https://github.com/swcarpentry/r-novice-inflammation, 10.5281/zenodo.57541.

EXERCISE #1 SOLN

- Answer: 3

Explanation: You want to extract the part of the dataframe representing data for patient 5 from days three to seven. In this dataframe, patient data is organised in columns and the days are represented by the rows. Subscripting in R follows the [i,j] principle, where i=columns and j=rows. Thus, answer 3 is correct since the patient is represented by the value for i (5) and the days are represented by the values in j, which is a slice spanning day 3 to 7.

Source: John Blischak, Daniel Chen, Harriet Dashnow, and Denis Haine (eds): "Software Carpentry: Programming with R." Version 2016.06, June 2016, https://github.com/swcarpentry/r-novice-inflammation, 10.5281/zenodo.57541.

EXERCISE #2

Using the Apply Function on Patient Data

Challenge: the apply function can be used to summarize datasets and subsets of data across rows and columns using the MARGIN argument. Suppose you want to calculate the mean inflammation for specific days and patients in the patient dataset (i.e. 60 patients across 40 days).

Please use a combination of the apply function and indexing to:

1. calculate the mean inflammation for patients 1 to 5 over the whole 40 days
2. calculate the mean inflammation for days 1 to 10 (across all patients).
3. calculate the mean inflammation for every second day (across all patients).

Think about the number of rows and columns you would expect as the result before each apply call and check your intuition by applying the mean function.

Source: John Blischak, Daniel Chen, Harriet Dashnow, and Denis Haine (eds): "Software Carpentry: Programming with R." Version 2016.06, June 2016, https://github.com/swcarpentry/r-novice-inflammation, 10.5281/zenodo.57541.

EXERCISE #2 - SOLN

```
# 1.

apply(dat[1:5, ], 1, mean)

# 2.

apply(dat[, 1:10], 2, mean)

# 3.

apply(dat[, seq(1,40, by=2)], 2, mean)
```

Source: John Blischak, Daniel Chen, Harriet Dashnow, and Denis Haine (eds): "Software Carpentry: Programming with R." Version 2016.06, June 2016, https://github.com/swcarpentry/r-novice-inflammation, 10.5281/zenodo.57541.

EXERCISE #3

Create a Function

In the last lesson, we learned to concatenate elements into a vector using the `c` function, e.g. `x <- c("A", "B", "C")` creates a vector `x` with three elements. Furthermore, we can extend that vector again using `c`, e.g. `y <- c(x, "D")` creates a vector `y` with four elements. Write a function called `fence` that takes two vectors as arguments, called `original` and `wrapper`, and returns a new vector that has the wrapper vector at the beginning and end of the original:

```
best_practice <- c("Write", "programs", "for", "people", "not", "computers")
asterisk <- "***"  # R interprets a variable with a single value as a vector
                   # with one element.
fence(best_practice, asterisk)
```

Source: John Blischak, Daniel Chen, Harriet Dashnow, and Denis Haine (eds): "Software Carpentry: Programming with R." Version 2016.06, June 2016, https://github.com/swcarpentry/r-novice-inflammation, 10.5281/zenodo.57541.

EXERCISE #3 - SOLN

```
# write a function to surround or "fence" a variable with another
variable:
# define the vector
best_practice <- c("Write", "programs", "for", "people", "not",
"computers")
# define the fence
fence <- "|||"
# define the buildfence function
buildfence <- function(original, wrapper) {
  answer <- c(wrapper, original, wrapper)
  return(answer)
}
# use the buildfence function to build the fence around the vector
buildfence (best_practice, fence)
```

Source: John Blischak, Daniel Chen, Harriet Dashnow, and Denis Haine (eds): "Software Carpentry: Programming with R." Version 2016.06, June 2016, https://github.com/swcarpentry/r-novice-inflammation, 10.5281/zenodo.57541.

EXERCISE #4

Functions to Create Graphs

Write a function called `analyze` that takes a filename as a argument and displays the three graphs produced in the previous lesson (average, min and max inflammation over time). `analyze("data/inflammation-01.csv")` should produce the graphs already shown, while `analyze("data/inflammation-02.csv")` should produce corresponding graphs for the second data set. Be sure to document your function with comments.

Source: John Blischak, Daniel Chen, Harriet Dashnow, and Denis Haine (eds): "Software Carpentry: Programming with R." Version 2016.06, June 2016, https://github.com/swcarpentry/r-novice-inflammation, 10.5281/zenodo.57541.

EXERCISE #4

```
analyze <- function(filename) {
  # Plots the average, min, and max inflammation over time.
  # Input is character string of a csv file.
  dat <- read.csv(file = filename, header = FALSE)
  avg_day_inflammation <- apply(dat, 2, mean)
  plot(avg_day_inflammation)
  max_day_inflammation <- apply(dat, 2, max)
  plot(max_day_inflammation)
  min_day_inflammation <- apply(dat, 2, min)
  plot(min_day_inflammation)
}
```

Source: John Blischak, Daniel Chen, Harriet Dashnow, and Denis Haine (eds): "Software Carpentry: Programming with R." Version 2016.06, June 2016, https://github.com/swcarpentry/r-novice-inflammation, 10.5281/zenodo.57541.