

The background of the slide is a dark blue field filled with numerous glowing, light blue lines and curves. These lines vary in thickness and brightness, some appearing as sharp, straight streaks while others form soft, flowing arcs. The overall effect is reminiscent of a digital data stream or a complex network visualization, creating a sense of motion and technological sophistication.

# Introduction to R

# Introductions

Instructors:

- Kyle Horton
- Georgia Titcomb
- Brian Gerber

[Download presentation as pdf](#)



**FISH, WILDLIFE, AND  
CONSERVATION BIOLOGY**  
**COLORADO STATE UNIVERSITY**



Colorado Cooperative  
Fish and Wildlife Research Unit

# Why learn to code?

- efficiency
- transparency
- flexibility in application
- shareable
- marketable skill
- needed for publications

# Software



# What is R?

R is a “suite of software facilities for data manipulation, calculation and graphical display.”

R uses **packages** that are collections of functions, data, and compiled code in a “well-defined format”.

**Packages** are downloaded from The Comprehensive R Archive Network (CRAN), R’s central software repository. Also, on GitHub, GitLab, BitBucket or other code sharing platforms.

# Why use R?

- open-source and free
- small total user base / large in ecology and statistics
- find help online, e.g., [stackoverflow](#)
- data management
- statistics
- plotting / graphics

# What is RStudio?

RStudio is an “Integrated Development Environment (IDE)”. It brings tools/languages together. We use R within RStudio.

# Why use RStudio?

- Makes using R easier
- [Projects](#) (file mgmt)
- [R Shiny](#): Interactive online apps
- [R Markdown](#): Interactive documents
- [Quarto](#): interactive articles, websites, blog, ...
- [Posit](#) - Certified B corp



# Online resources to learn R

- [Intro to R for Biologists](#)
- [Introduction to R - tidyverse](#)
- [R for Data Science \(2e\)](#)
- [Advanced R](#)
- [Introduction to the R Language](#)
- [Introduction to R](#)
- [An Introduction to R for Research](#)
- [Introduction to Data Exploration and Analysis with R](#)
- [Working with Data in R](#)

# Today

## Goal

‘Get familiar with fundamentals of R useful for data’

‘To get beyond the initial shock or fear of programming and start using R’

# Today

## Learning Objectives

- Write and execute code in R via RStudio
- R language vocabulary
- Find help
- Read/write data
- Manipulate data efficiently
- Plot data or results

# Today

## Execution

- Presentation / code walk through
- Challenges (independent or in teams of 2-3)

# Today

## Schedule

- 900 - 930: Introductions and setup
- 930 - 1000: RStudio and R (objects and functions)
- 1000 - 1130: Data input and output
- 1130- 1200: Finding help
- 1200 - 1300: Lunch
- 1300 - 1400: Data mgmt
- 1400 - 1500: Plotting
- 1500 - 1600: Final Challenge

# Showcases

Brian - [R Shiny application](#) that allows users to subset data and visualize 14,586 results

Kyle - example here

Georgia - example here

# RStudio

The screenshot displays the RStudio application window. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. Below the menu is a toolbar with icons for file operations and a 'Go to file/function' search bar. The main interface is divided into four panes:

- Console:** Shows the R version 4.4.1 (2024-06-14 ucrt) and copyright information. It also displays the R license text and instructions for using 'demo()', 'help()', and 'q()'.
- Environment:** Currently empty, showing 'Environment is empty'.
- Files:** Shows the file explorer with a search bar and a list of files.
- Packages:** Displays a list of installed and available packages, including 'base', 'base64enc', 'bayesplot', 'BH', 'boot', 'bridgesa...', and 'brms'.

The console output is as follows:

```
R version 4.4.1 (2024-06-14 ucrt) -- "Race for Your Life"
Copyright (c) 2024 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

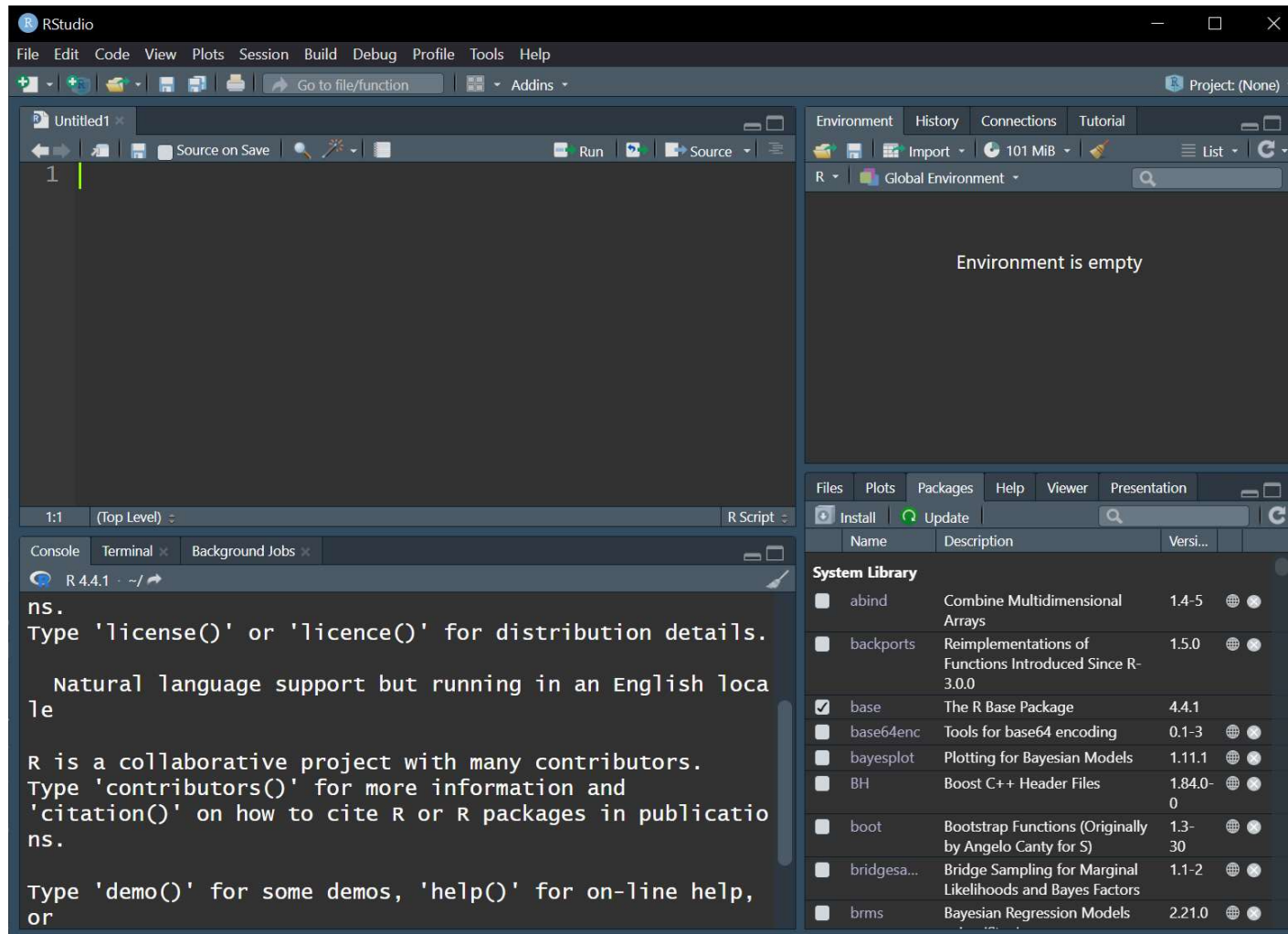
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```

Name	Description	Version
abind	Combine Multidimensional Arrays	1.4-5
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.5.0
base	The R Base Package	4.4.1
base64enc	Tools for base64 encoding	0.1-3
bayesplot	Plotting for Bayesian Models	1.11.1
BH	Boost C++ Header Files	1.84.0-0
boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-30
bridgesa...	Bridge Sampling for Marginal Likelihoods and Bayes Factors	1.1-2
brms	Bayesian Regression Models	2.21.0

# RStudio





# Installing Packages

The screenshot shows the RStudio interface with the 'Install Packages' dialog box open. The dialog box has the following fields and options:

- Install from:** Repository (CRAN) (with a link to 'Configuring Repositories')
- Packages (separate multiple with space or comma):** tidyverse
- Install to Library:** C:/Users/C825033651/AppData/Local/Programs/R/R-4.4.1/lib
- ☒ Install dependencies
- Buttons:** Install, Cancel

The background script editor shows the following code:

```
1 install.packages('tidyverse')
2
```

The right-hand pane shows the 'Environment' tab with a table of values:

values
y
3

The bottom-right pane shows the 'Packages' tab with a table of installed and available packages:

Name	Description	Versi...
<b>System Library</b>		
<input type="checkbox"/> abind	Combine Multidimensional Arrays	1.4-5
<input type="checkbox"/> backports	Reimplementations of Functions Introduced Since R-3.0.0	1.5.0
<input checked="" type="checkbox"/> base	The R Base Package	4.4.1
<input type="checkbox"/> base64enc	Tools for base64 encoding	0.1-3
<input type="checkbox"/> bayesplot	Plotting for Bayesian Models	1.11.1
<input type="checkbox"/> BH	Boost C++ Header Files	1.84.0-0
<input type="checkbox"/> boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-30
<input type="checkbox"/> bridgesa...	Bridge Sampling for Marginal Likelihoods and Bayes Factors	1.1-2
<input type="checkbox"/> brms	Bayesian Regression Models	2.21.0

# The language of R

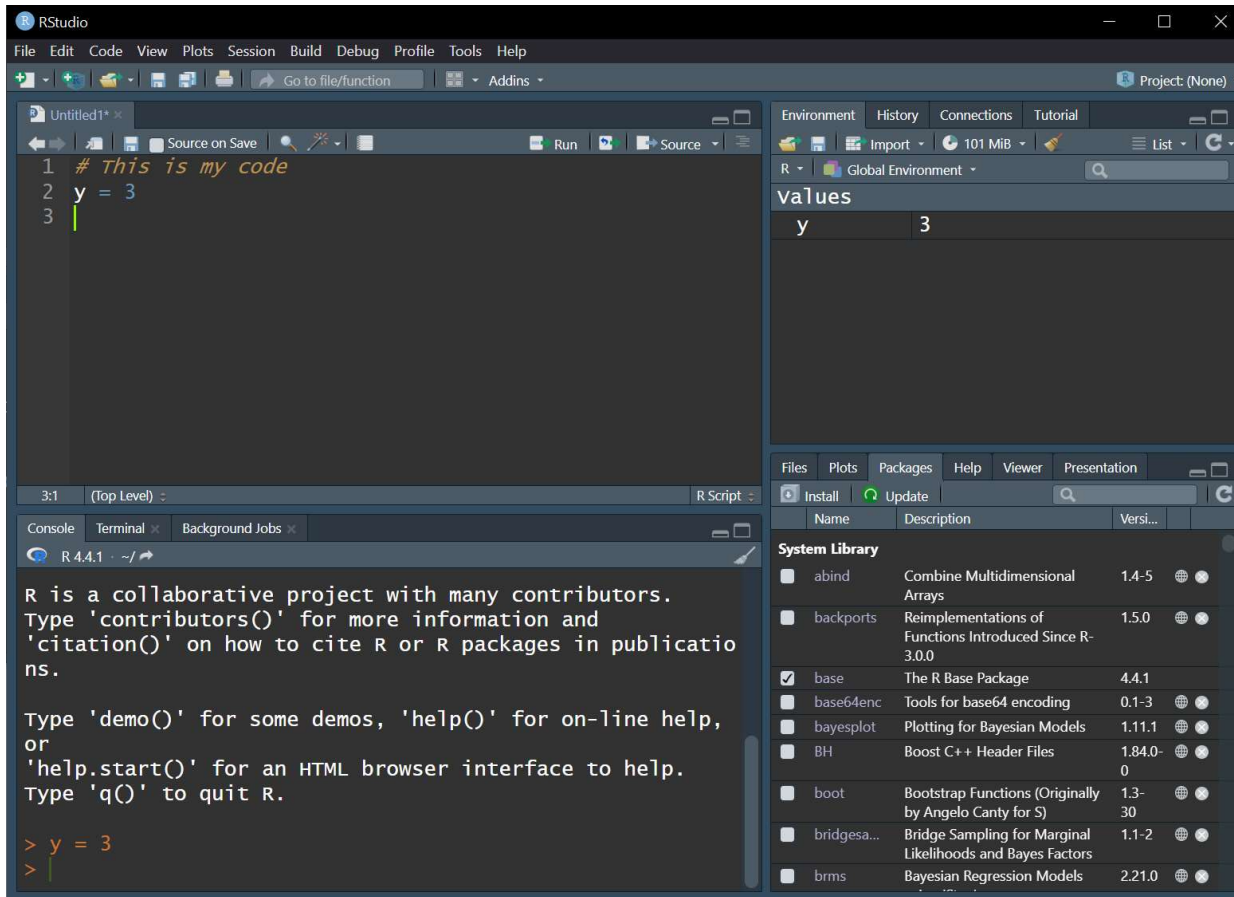
## Objects

A storage place for information; stored in the “Environment”

*‘Attributes’* describes the structure or information of the object

# The language of R

## Objects



The screenshot displays the RStudio environment. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. The toolbar below the menu contains icons for file operations and running code. The main editor window shows a script with the following content:

```
1 # This is my code
2 y = 3
3 |
```

The console at the bottom left shows the R prompt and the following text:

```
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publicatio
ns.

Type 'demo()' for some demos, 'help()' for on-line help,
or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> y = 3
> |
```

The environment pane on the right shows the Global Environment with a single variable 'y' of type 'values' containing the value 3. The bottom right pane shows the System Library with a list of installed packages:

Name	Description	Versi...
abind	Combine Multidimensional Arrays	1.4-5
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.5.0
base	The R Base Package	4.4.1
base64enc	Tools for base64 encoding	0.1-3
bayesplot	Plotting for Bayesian Models	1.11.1
BH	Boost C++ Header Files	1.84.0-0
boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-30
bridgesa...	Bridge Sampling for Marginal Likelihoods and Bayes Factors	1.1-2
brms	Bayesian Regression Models	2.21.0

# Code for Presentation

R Code Script for the remaining code is [HERE](#)

Left-Click 'HERE' -> Ctrl A -> Ctrl C. Go to RStudio, left-click on an empty script. Ctrl V

**OR**

Right-click 'HERE' -> 'Save link as...'. Save file to location. Go to RStudio. File -> Open File... find your file.

# The language of R

## Objects

```
1 # y is an 'object' that is assigned the value 3
2 y = 3
3 y
```

```
[1] 3
```

```
1 # Same operation '=' '<-'
2 y <- 3
```

# The language of R

## Objects

```
1 # We can create new objects from objects
2 y2 = y-2
3 y2
```

```
[1] 1
```

```
1 # We can do math with our objects
2 # Mind your parentheses (order of operation)
3 y*2 / y*4
```

```
[1] 8
```

```
1 y*2 / (y*4)
```

```
[1] 0.5
```

# Challenge 1

Compute the diameter (d) of the Earth (in km) at the equator using this formula for the circumference (c)...

$$d = \frac{c}{\pi}$$

- $c = 24,901.55$  miles
- $1 \text{ km} = 0.621$  miles
- **Hint** type in 'pi' to see what you get

1. Convert the circumference from miles to km.
2. Write the formula in R and plug in the value for circumference

► [Click for Answer](#)

# The language of R

## Functions

‘does stuff’; creates or manipulates objects

*Arguments* are the types of things a function is asking for; the inputs



# The language of R

**object** = function(attribute1 = input1, attribute2 = input2)

**object** = function(input1, input2)

**this** = sign(x = -5)

```
1 sign(-5)
```

```
[1] -1
```

```
1 sign(54)
```

```
[1] 1
```

# The language of R

## Functions

```
1 # function - 'c' - concatenate
2 y = c(1, 2, 3, 4, 5, 6)
```

```
1 is.numeric(y)
```

```
[1] TRUE
```

```
1 # The function 'class' has the argument 'x'
2 is.numeric(x = y)
```

```
[1] TRUE
```

# The language of R

## Functions

- 1 # How to find out the arguments of a function?
- 2 `?is.numeric`

R: Numeric Vectors ▾ Find in Topic

numeric {base} R Documentation

### Numeric Vectors

**Description**

Creates or coerces objects of type "numeric". `is.numeric` is a more general test of an object being interpretable as numbers.

**Usage**

```
numeric(length = 0)
as.numeric(x, ...)
is.numeric(x)
```

**Arguments**

<code>length</code>	A non-negative integer specifying the desired length. Double values will be coerced to integer: supplying an argument of length other than one is an error.
<code>x</code>	object to be coerced or tested.
<code>...</code>	further arguments passed to or from other methods.

**Details**

# The language of R

## Wrapping functions

```
1 # Functions commonly 1) wrapped, 2) have multiple arguments
2 x = matrix(
3     data = c(1, 2, 3, 4, 5, 6),
4     nrow = 2,
5     ncol = 3
6 )
```

```
1 x
```

	[, 1]	[, 2]	[, 3]
[1, ]	1	3	5
[2, ]	2	4	6

# The language of R

## Values

- numeric
- integer
- character
- factor

## Objects

- vector
- matrix
- array
- list
- dataframe

# Types of Values

## Numeric

```
1 y = 3.3  
2 class(y)
```

```
[1] "numeric"
```

## Integer

```
1 y = integer(3)  
2 class(y)
```

```
[1] "integer"
```

## Character

```
1 y = "habitat"  
2 class(y)
```

```
[1] "character"
```

# Factor

```
1 y = factor("habitat")  
2 class(y)
```

```
[1] "factor"
```

# Types of Objects

## Vector

```
1 # An ordered collection indexed 1,2,...n
2 # Using the function 'c' to concatenate
3 z1 = c(4,5,6)
4 z1
```

```
[1] 4 5 6
```

4 is in element/index/position 1 of the vector

6 is in element/index/position 3 of the vector

```
1 # the dimension of a vector
2 length(z1)
```

```
[1] 3
```

```
1 # A vector of characters
2 z2 = c("dog","cat","horse")
3 z2
```



```
[1] "dog"    "cat"    "horse"
```

```
1 z3 = c("dog", "1", "horse")
```

```
2 z3
```

```
[1] "dog"    "1"      "horse"
```

```
1 z3 = c("dog", 1, "horse")
```

```
2 z3
```

```
[1] "dog"    "1"      "horse"
```

# Types of Objects

## Subsetting a vector

```
1 z3 = c("dog", "1", "horse", "chicken")  
2 z3[1]
```

```
[1] "dog"
```

```
1 z3[2]
```

```
[1] "1"
```

```
1 z3[2:4]
```

```
[1] "1"          "horse"      "chicken"
```

```
1 z3[c(2, 3)]
```

```
[1] "1"          "horse"
```

```
1 z3[-1]
```

```
[1] "1"      "horse"  "chicken"
```

# Types of Objects

## Vector of factors, a special kind of character string

```
1 z4 = factor(  
2       c("dog", "dog", "cat", "horse")  
3       )
```

```
1 z4
```

```
[1] dog    dog    cat    horse  
Levels: cat dog horse
```

```
1 levels(z4)
```

```
[1] "cat"    "dog"    "horse"
```

```
1 summary(z4)
```

```
cat  dog  horse  
  1    2    1
```

# Types of Objects

## Matrix

```
1 x = matrix(  
2           c(1,2,3,4,5,6),  
3           nrow = 2,  
4           ncol = 3  
5           )
```

```
1 x
```

```
[1,] 1 3 5  
[2,] 2 4 6
```

```
1 dim(x)
```

```
[1] 2 3
```

# Types of Objects

## Subsetting a matrix

```
1 # get element of row 1 and column 2  
2 x[1,2]
```

```
[1] 3
```

```
1 # get element of row 2 and column 6  
2 x[2,3]
```

```
[1] 6
```

```
1 # get element all elements of row 2  
2 x[2,]
```

```
[1] 2 4 6
```

```
1 # same as  
2 x[2,1:3]
```

```
[1] 2 4 6
```

# Types of Objects

## Array

```
1 # ARRAY - more than just two dimensions
2 z5 = array(
3         c("a","b","c","d","e","f"),
4         dim= c(2,2,2)
5     )
```

```
1 dim(z5)
```

```
[1] 2 2 2
```

```
1 z5
```

```
, , 1
```

```
 [1,] [,1] [,2]
 [2,] "a"  "c"
      "b"  "d"
```

```
, , 2
```

```
 [,1] [,2]
```

$\begin{bmatrix} 1, \\ 2, \end{bmatrix}$     "e"    "a"  
              "f"    "b"



# Types of Objects

## List

```
1 # LIST - a bucket - will take anything
2 my.list = list(z1,z2,z3,z4,z5)
```

```
1 #Subset a list
2 my.list[[1]]
```

```
[1] 4 5 6
```

```
1 my.list[[4]]
```

```
[1] dog    dog    cat    horse
Levels: cat dog horse
```

# Types of Objects

## Data frame

A row for each observation and a column for each variable (can be different types).

```
1 x = data.frame(outcome = c(1, 0, 1, 1),  
2                 exposure = c("yes", "yes", "no",  
3                             age = c(24, 55, 39, 18)  
4                 )  
5 x
```

	outcome	exposure	age
1	1	yes	24
2	0	yes	55
3	1	no	39
4	1	no	18

# Types of Objects

## Subset data.frame

```
1 x$exposure
```

```
[1] "yes" "yes" "no"  "no"
```

```
1 x['exposure']
```

```
exposure
1      yes
2      yes
3      no
4      no
```

```
1 x[,2]
```

```
[1] "yes" "yes" "no"  "no"
```

## Challenge 2

1. Create a vector of numbers that has length 6; call this object 'vec1'.
2. Use the function 'mean' to find the mean of the values of vec1.
3. Subset vec1 to only elements 4 through 6. Call this new object 'vec1a'.
4. Create a new vector (length 3) of characters called "hab1", "hab2", and "hab3". Call this object 'vec2'.
5. Put vec1a and vec2 together into a data frame and call this object 'dat'

► [Click for Answer](#)