

Intro to R and Bioconductor

HMS Research Computing

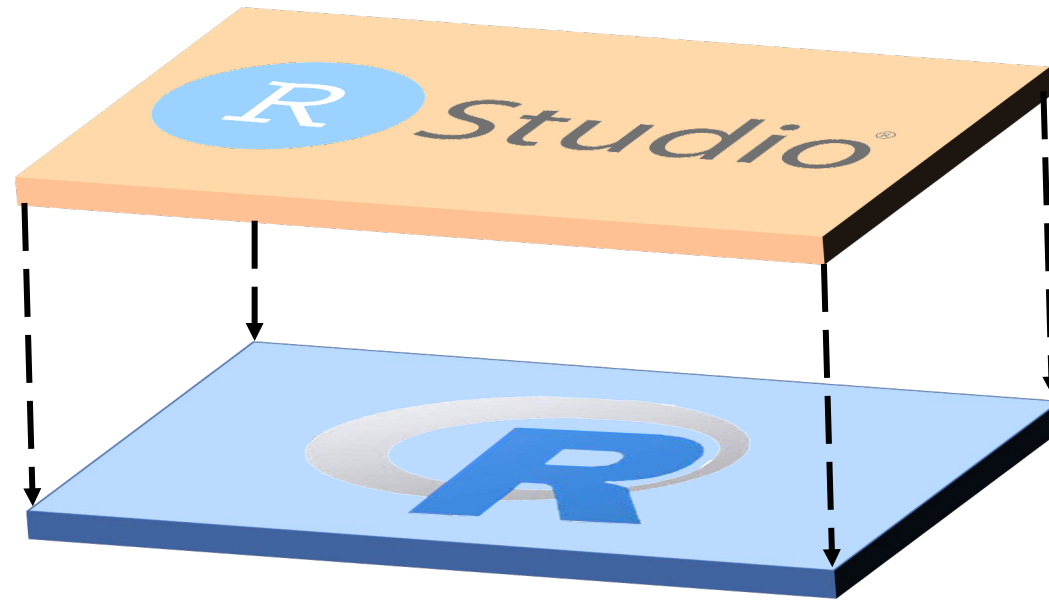
Please fill out the survey

- Accessible through the Harvard Training Portal
- <https://trainingportal.harvard.edu>
- Click on “Me” then “Intro to R and Bioconductor”
- Scroll to “Evaluations” and click on the survey
- We appreciate any feedback or comments!

Course Objectives

- Learn to run RStudio on O2
- Gain familiarity with R and Bioconductor
- Learn to import and export data
- Class Exercise

R vs RStudio



R on O2

Interactive Job

```
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.
```

```
> |
```

Sbatch Job

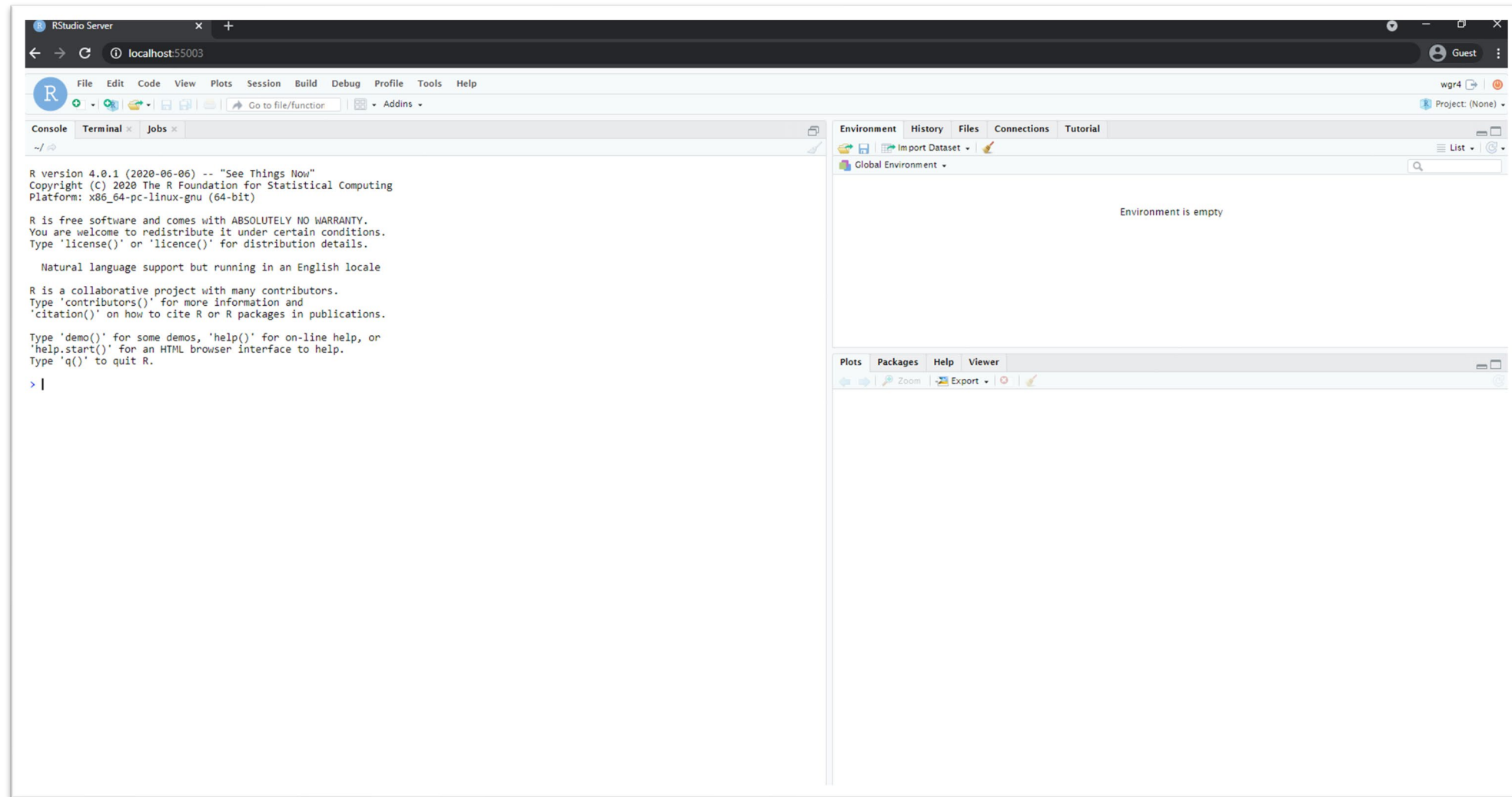
```
#!/bin/bash
```

```
#SBATCH -p short # Partition to submit to  
#SBATCH -t 0-00:01 # Time in minutes DD-HH:MM; DD-HH; MM:SS  
#SBATCH -c 1 # Number of cores requested  
#SBATCH -N 1 # Ensure that all cores are on one machine  
#SBATCH --mem=2G # Memory total in GB  
#SBATCH -o hostname.%j.out # Standard out goes to this file  
#SBATCH -e hostname.%j.err # Standard err goes to this file
```

```
# Commands below  
module load gcc/6.2.0  
module load R/3.6.1
```

```
# To run a R script called my_r_script.R  
Rscript my_r_script.R
```

RStudio on O2



Launch RStudio on O2

1

```
wgr4@login05:~ wgr4@login06:~
Welcome to O2 (Orchestra 2)!

  O2

You've landed on login06 which is a
8 core system with 15.49 GiB memory
running kernel 3.10.0 born on 2020-07-07

=== O2 =====

News (Mar 25 2021)

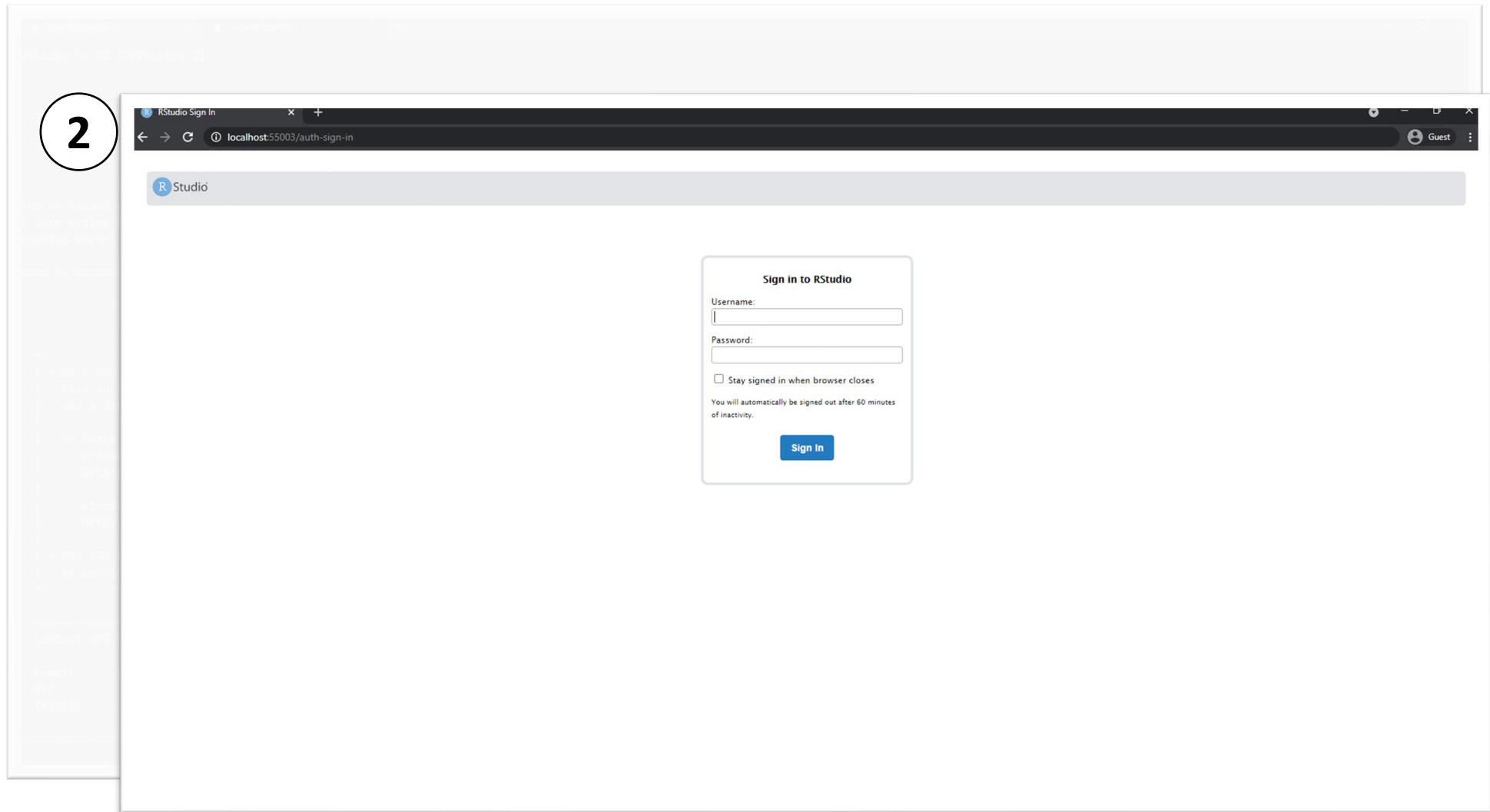
+-----+
| * O2's GPU capacity has increased with funding from the Blavatnik Institute! |
| This includes: 71 new NVIDIA GPUs: 27 RTX8000 and 44 Tesla V100S cards,   |
| and a dedicated GPU scratch storage filesystem.                             |
|                                                                              |
| ** These GPU resources are currently available only to labs whose PI has a  |
| primary or secondary appointment in an HMS pre-clinical department.         |
| Details at:                                                                  |
|                                                                              |
|   https://wiki.rc.hms.harvard.edu/display/O2/Using+O2+GPU+resources         |
|   https://wiki.rc.hms.harvard.edu/display/O2/Scratch_gpu+Storage           |
|                                                                              |
| * HMS VPN is not required for using O2. Only use VPN when absolutely needed. |
| ** Copying large data over VPN can greatly impact its performance!          |
+-----+

-----
Contact HMS Research Computing:

E-mail   rchelp@hms.harvard.edu
Web      it.hms.harvard.edu/rc
Twitter  @hms_rc

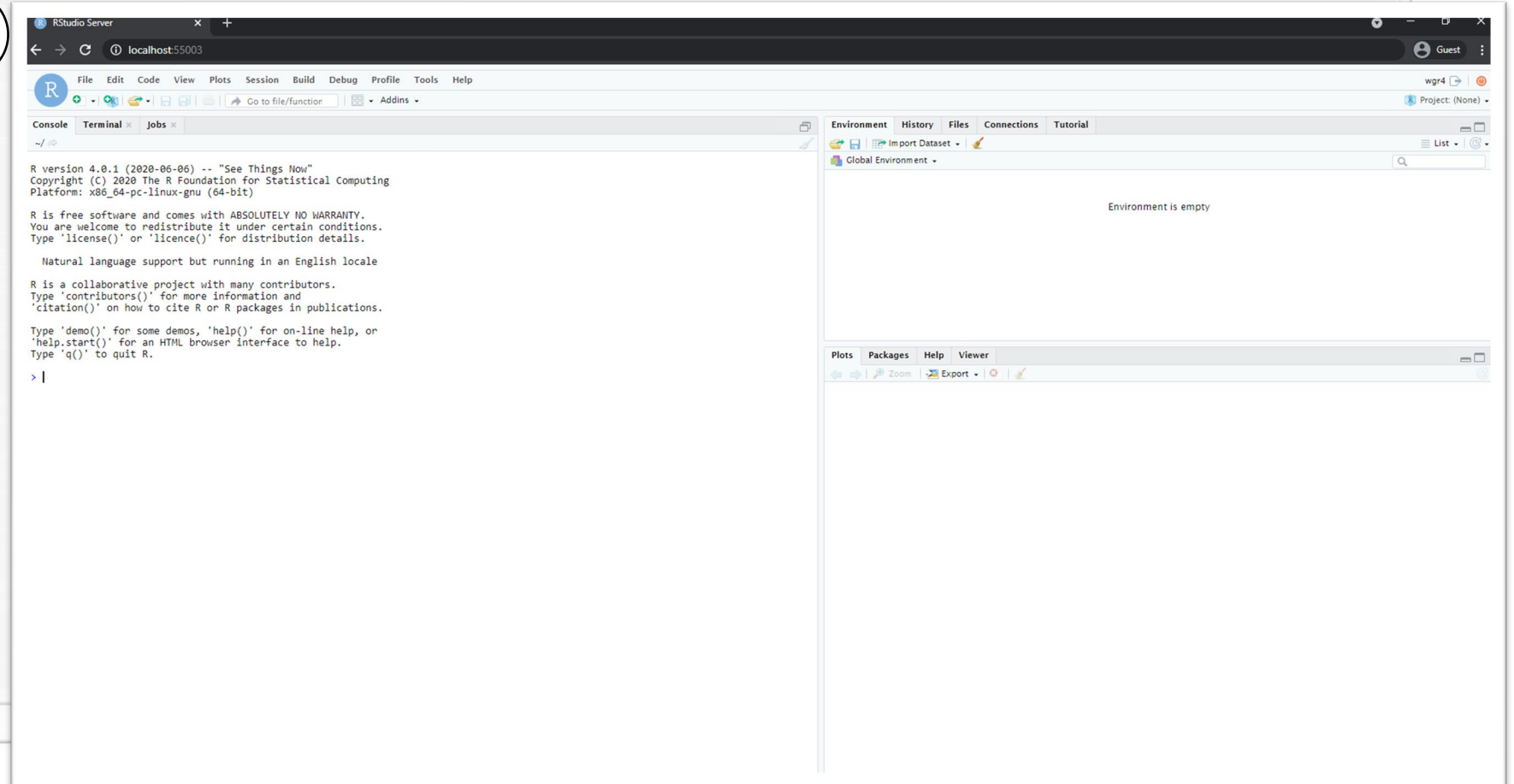
=====
```

Launch RStudio on O2



Launch RStudio on O2

3



Blue content: try it out!

Copy to clipboard

Class Material

- Intro to R and Bioconductor - https://github.com/hmsrc/user-training/blob/master/Intro_to_R_and_Bioconductor.pdf

Managing your R packages on O2

Create an R Personal Library

```
1 # Create a library on your HOME ("~") directory
2 mkdir ~/R-4.0.1
3 # Create an .Renviron file
4 echo 'R_LIBS_USER=~/.R-4.0.1' > $HOME/.Renviron'
```

Copy



Running RStudio on O2

Logging Into O2



While running XQuartz on the background, open a terminal (type “terminal” on the search bar)



Open a terminal (type “terminal” on the search bar)



Open MobaXterm

Logging Into O2



While running XQuartz on the background, open a terminal (type “terminal” on the search bar)



Open a terminal (type “terminal” on the search bar)



Open MobaXterm

```
ssh -Y -L PORT:127.0.0.1:PORT <your_HMS_ID>@o2.hms.harvard.edu
```

R on O2

- Available versions

\$ module spider R

- Load R module

\$ module load gcc/6.2.0 R/version

- How to unload an R module

\$ module unload R/version

- Important: start R from an interactive session (not login!)

\$ R

Managing your R packages on O2

- An R Personal Library is required on O2
- You must create an R Personal Library per version.
- It can be done in two steps

Managing your R packages on O2

- An R Personal Library is required on O2
- You must create an R Personal Library per version.
- It can be done in two steps

1) Create an R Personal Library directory

```
$ mkdir ~/R-4.1.1
```

2) Create an .Renviron file

```
$ echo 'R_LIBS_USER=~ /R-4.1.1'> $HOME/.Renviron'
```


RStudio on O2

1. Connect to O2

```
me@my_computer:~$ ssh -Y -L PORT:127.0.0.1:PORT <your_HMS_ID>@o2.hms.harvard.edu
```

2. Load Modules

```
ecommons@login01:~$ module load rstudio_launcher/1.0
```

```
ecommons@login01:~$ module load gcc/6.2.0
```

```
ecommons@login01:~$ module load R/4.1.1
```

3. Launch RStudio

```
ecommons@login01:~$ srun -t 0-2:00 --pty -p interactive -c 1 --mem=2G --x11 --tunnel PORT:PORT  
RStudio_launcher.sh PORT
```

Launch RStudio command on O2

\$ **srun -t 0-2:00 --pty -p interactive -c 1 --mem=2G --x11 --tunnel PORT:PORT RStudio_launcher.sh PORT**

**SLURM command to obtain
a job allocation**

Launch RStudio command on O2

```
$ srun -t 0-2:00 --pty -p interactive -c 1 --mem=2G --x11 --tunnel PORT:PORT RStudio_launcher.sh PORT
```

Walltime
(DD-HH:MM)

Launch RStudio command on O2

```
$ srun -t 0-2:00 --pty -p interactive -c 1 --mem=2G --x11 --tunnel PORT:PORT RStudio_launcher.sh PORT
```



Pseudo terminal mode

Launch RStudio command on O2

```
$ srun -t 0-2:00 --pty -p interactive -c 1 --mem=2G --x11 --tunnel PORT:PORT RStudio_launcher.sh PORT
```



Partition name

Launch RStudio command on O2

```
$ srun -t 0-2:00 --pty -p interactive -c 1 --mem=2G --x11 --tunnel PORT:PORT RStudio_launcher.sh PORT
```



Number of CPUs

Launch RStudio command on O2

```
$ srun -t 0-2:00 --pty -p interactive -c 1 --mem=2G --x11 --tunnel PORT:PORT RStudio_launcher.sh PORT
```



Memory
(units: K|M|G|T)

A red line connects the bottom of the red box containing `--mem=2G` to the top of the **Memory** box.

Launch RStudio command on O2

```
$ srun -t 0-2:00 --pty -p interactive -c 1 --mem=2G --x11 --tunnel PORT:PORT RStudio_launcher.sh PORT
```



**Enables X11
forwarding**

Launch RStudio command on O2

\$ `srun -t 0-2:00 --pty -p interactive -c 1 --mem=2G --x11 --tunnel PORT:PORT RStudio_launcher.sh PORT`

Required to execute RStudio
launcher

Launch RStudio command on O2

```
$ srun -t 0-2:00 --pty -p interactive -c 1 --mem=2G --x11 --tunnel PORT:PORT RStudio_launcher.sh PORT
```

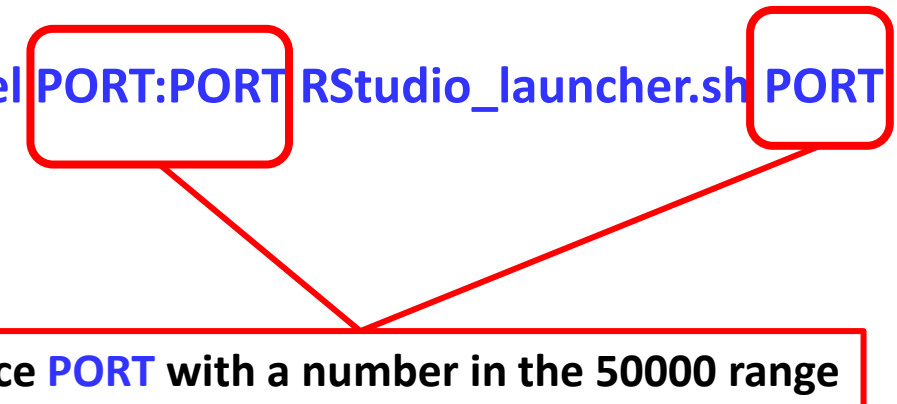


```
graph TD; A[PORT:PORT] --- B[Replace PORT with a number in the 50000 range]; C[PORT] --- B;
```

Replace **PORT** with a number in the 50000 range

Launch RStudio command on O2

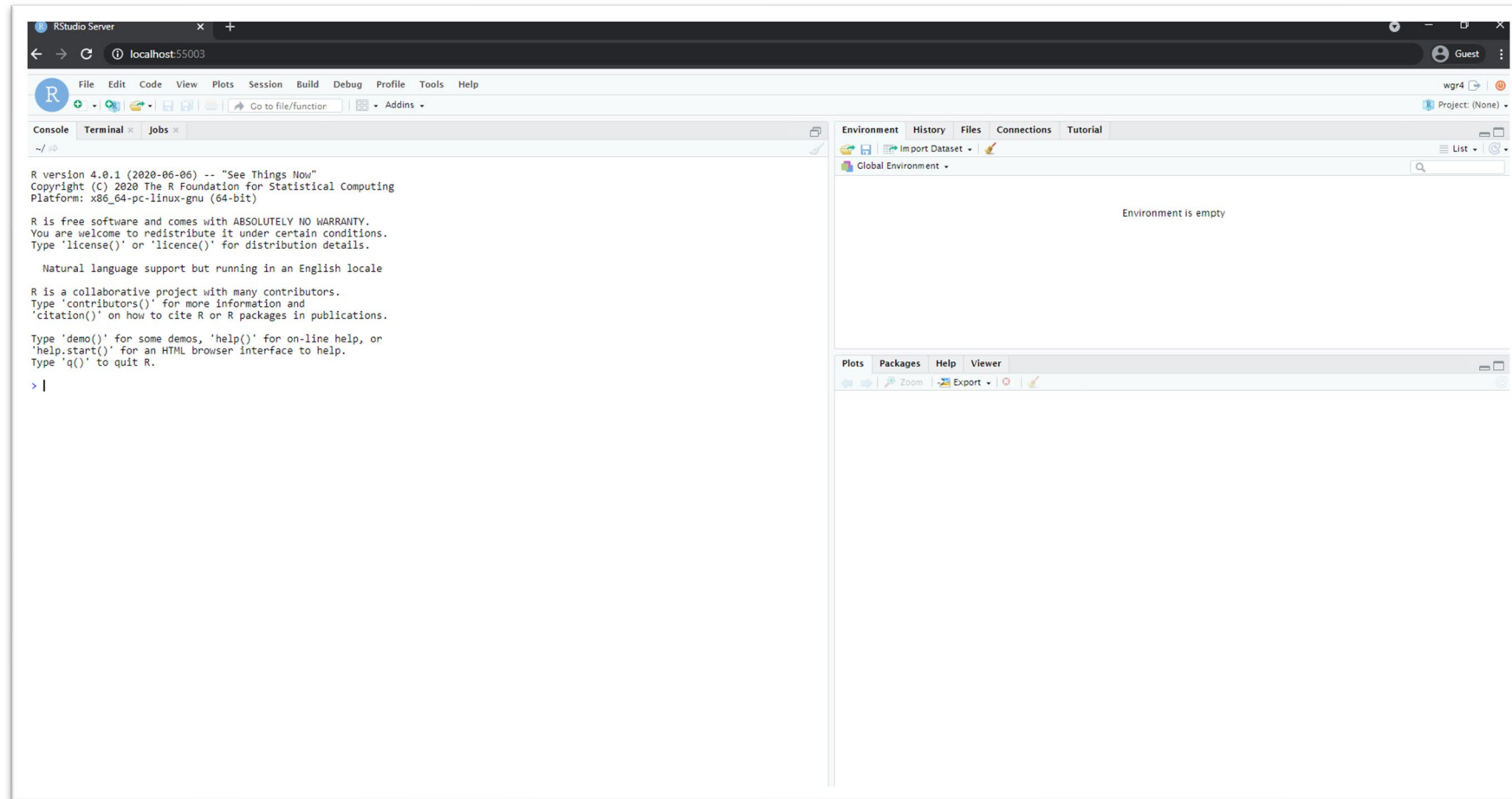
```
$ srun -t 0-2:00 --pty -p interactive -c 1 --mem=2G --x11 --tunnel PORT:PORT RStudio_launcher.sh PORT
```



Replace **PORT** with a number in the 50000 range

```
[wgr4@login06 ~]$ srun -t 0-1:00 --pty -p interactive -c 1 --mem=2G --x11 --tunnel 55003:55003 RStudio_launcher.sh 55003
srun: job 31216484 queued and waiting for resources
srun: job 31216484 has been allocated resources
You can now access RStudio on your local web browser at http://localhost:55003
Login username = wgr4
Password = gqYRIyfRie0y4v9QgaeQ
```

Launch RStudio on O2



Create an R script

- Copy the “[Coding in R](#)” block to your clipboard
- Click on File -> New File -> R Script
- Paste the “Coding in R” from the clipboard to the new empty file
- Save it as “IntroToR” on your home directory on O2
 - If prompted to choose an encoding, select “ASCII (System default)”

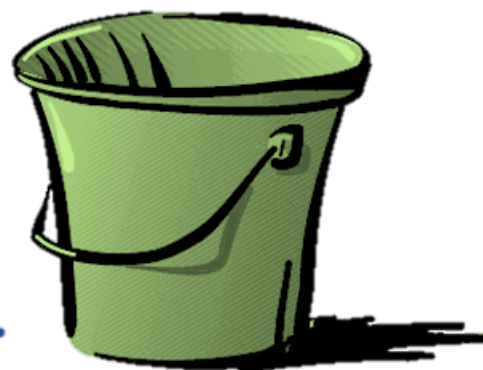
Installing R packages - four general “bins”



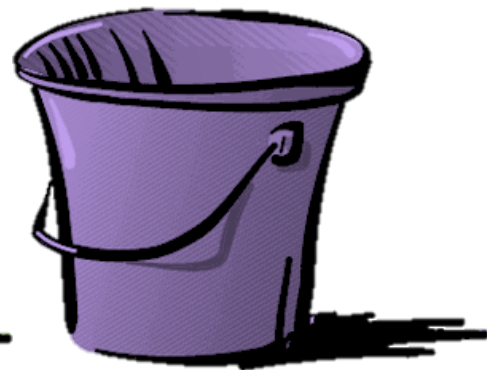
CRAN



Source



Bioconductor



GitHub

Installing R packages - four general “bins”

```
install.packages("<package_name>")
```



Installing R packages - four general “bins”

```
install.packages("<package_name>")
```



```
install.packages("path/to/package/file", repos=NULL)
```



Installing R packages - four general “bins”

```
install.packages("<package_name>")
```



CRAN

```
install.packages("path/to/package/file", repos=NULL)
```



Source

```
BiocManager::install("<package_name>")
```

Bioconductor



Installing R packages - four general “bins”

```
install.packages("<package_name>")
```



CRAN

```
install.packages("path/to/package/file", repos=NULL)
```



Source

```
BiocManager::install("<package_name>")
```

Bioconductor



```
install_github("<GH_repo>/<package_name>@vX.X.X")
```



GitHub

Exercise: Install and load the edgeR package from Bioconductor

- Install package from Bioconductor
 - > **BiocManager::install("edgeR")**
- Load package
 - > **library("edgeR")**

*For support of any Bioconductor package - <https://support.bioconductor.org/>

R documentation

- General R help on a function

?name_of_function

- For example:

> **?t.test**

Setting your R “working directory”

- Returns filepath to current **w**orking **d**irectory:

> **getwd()**

- Setting your working directory

> **setwd(“a/file/path/somewhere”)**

-OR-

ctrl + shift + H

Setting your R “working directory”

- Returns filepath to current **w**orking **d**irectory:

> **getwd()**

- Setting your working directory

> **setwd(“a/file/path/somewhere”)**

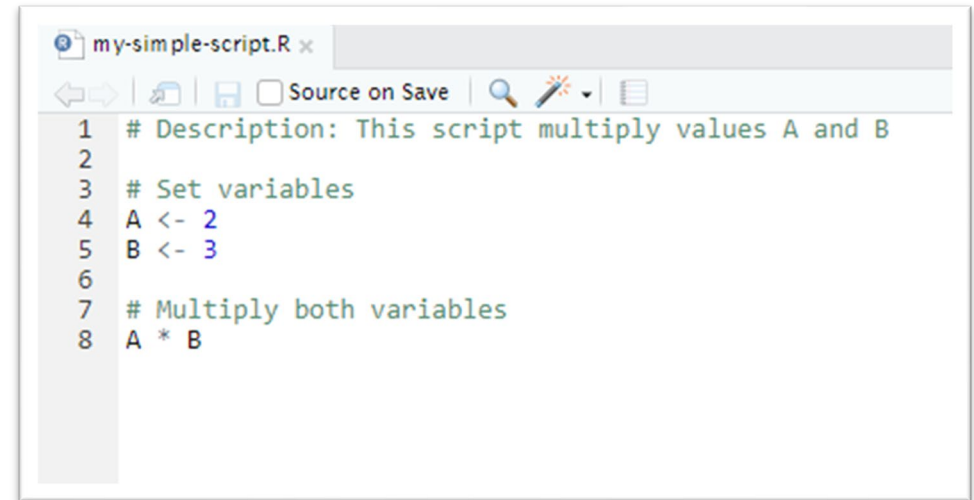
-OR-

ctrl + shift + H

Coding in R

- Create an R script is easy!
- Start a line with “#” to create a comment
- Comments are important
- Assign variables with a “<-”
- Example:

> **myX <- 5**



```
1 # Description: This script multiply values A and B
2
3 # Set variables
4 A <- 2
5 B <- 3
6
7 # Multiply both variables
8 A * B
```



Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

Vectors

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- Basic way to store data
- “c” is used to create a vector
- Vectors types: numeric, character, & logical
- Example:
 > **myvector <- c(3,5,7)**

Lists

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- Similar to vectors, but with mixed data types
- Example:

```
> myvector <- c(3,"Tp53",7)
```

Factors

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- Makes a vector nominal
- Mostly use in statistical modeling
- Numeric and character vectors can be made into factors
- Example:
 - > `gender <- c("male", "male", "female", "female", "female")`
 - > `gender <- factor(gender)`

Matrices

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- Data must be all the same type (i.e., numeric, character, or logical)
- Columns must have the same length
- Example:
 > **mymatrix <- matrix(c(1:6), nrow=3, ncol=2)**

Data frame

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- Subset of matrices allowing mixed types
 - > `mydataframe <- as.data.frame(mymatrix)`
- Columns can be named
 - > `names(mydataframe) <- c("col1", "col2")`
- Rows as well
 - > `row.names(mydataframe) <- mydataframe[,1]`

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

Indexing

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- Accessing elements

- Example:

```
> mymatrix <- matrix(c(1:6), nrow=3, ncol=2)
```

```
> mymatrix[1,2] #returns element in row 1 and column 2
```

```
> mymatrix[1,] #return all elements in row 1
```

```
> mymatrix[,1] #return all elements in column 1
```

Joining rows or columns

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- “**rbind**” to add row(s) to a pre-existing data frame or matrix
> mymatrix <- rbind(mymatrix, newrow)
- “**cbind**” to add column(s) to a pre-existing data frame or matrix
> mymatrix <- cbind(mymatrix, newcol)

Missing values

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- **NA**: Not Available
- **NaN**: Not a Number
- Example:
 - > **is.na(x)** #logical test for NA or NaN
 - > **is.nan(x)** #logical test for only NaN
 - > **x[!is.na(x)]** #subsets and excludes NAs

Change data type

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- Functions start with “as.” followed by the type

- Example:

```
> myvector <- c(3,5,7)
```

```
> myvector <- as.character(myvector)
```

Apply function

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- *Returns a vector, array, or list of values obtained by applying a function to margins of an array or matrix*
- Format: **apply** (to_what, *how*, *function*)
- Where *how* accepts a “1” to apply the *function* over rows or “2” to apply over columns
- For example:
 - > **apply**(mymatrix, 1, sum) #row sums
 - > **apply**(mymatrix, 2, sum) #column sums

Useful functions

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- > **class(object)** #gives object class
- > **mode(object)** #gives object type
- > **length(vector)** #gives length
- > **head(object)** #gives first 6 rows
- > **tail(object)** #gives last 6 rows
- > **summary()** #quick statistics
- > **nrow(object)** #gives number of rows
- > **ncol(object)** #gives number of columns
- > **str(object)** #gives object structure
- > **dim(object)** #gives matrix/df dimensions

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

Arithmetic

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

Operator	Description
+	Addition
-	Substraction
*	Multiplication
/	Division
^	Exponent
%%	Modulo (remainder)

Logical

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

Operator	Description
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
==	Exactly equal to
!=	Not equal to
	OR
&	AND

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

Text file



Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

Format: **`mydata <- read.table(file=" filename.csv", header=TRUE, sep=",")`**

- “sep=” field separator character
- “header=” logical value to specify whether the file contain column names
- “row.names=” a vector of row names (must be unique identifiers!)



MS Excel

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- R package is required (e.g., “xlsx”)
 `> install.packages(“xlsx”)`
 `> library (“xlsx”)`
- Read in the first worksheet from the workbook myexcel.xlsx
 `> mydata <- read.xlsx(“myexcel.xlsx”, sheetIndex=1)`
- Read in the worksheet named mysheet
 `> mydata <- read.xlsx(“myexcel.xlsx”, sheetName = "mysheet")`

Export

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

Format: **`write.table(x="ObjectName", file="FileName.txt", sep="\t")`**

- Optional arguments:

`row.names=FALSE` #turn off row names

`col.names=FALSE` #turn off column names

`quote=FALSE` #turn off character string quoting

Saving and Loading your workspace

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- Save and pick up where you leave off – saves variables

Format: **save.image(file="FileName.RData")**

-OR-

Format: **save(object list, file="FileName.RData")**

- Load workspace

Format: **load(file="FileName.RData")**

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

Import Data

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- Import Rcoursetestdata1.csv using a comma separator, header set to true, and row names to first column

```
> mdata <- "/n/groups/rc-training/introR/Rcoursetestdata1.csv"
```

```
> mydf <- read.table(mdata, header=TRUE, row.names=1, sep=",")
```

```
> head(mydf)
```

```
> head(mydf)
          TNBC1 TNBC2 TNBC3 Normal1 Normal2 Normal3
ENSG00000008988 15258 15077 144720   12095   43544   46883
ENSG00000009307 14660 20767   8678   13774   23030   18917
ENSG00000019582 50866 55775  15089    6696   13754   86319
ENSG00000026025 21174 47966  26682    6068   21126   12728
ENSG00000034510 25645 31574  56403   29590   25216   37199
ENSG00000044574 23910 27200  13757   13364   10852   12378
```

Basic Statistics

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- Return basic statistics

> **summary(mydf)**

```
> summary(mydf)
      TNBC1      TNBC2      TNBC3      Normal1
Min.   :    0  Min.   :   65  Min.   :   31  Min.   :   22
1st Qu.: 7888  1st Qu.: 9538  1st Qu.: 9324  1st Qu.: 5074
Median :13034  Median :16568  Median :19108 Median :10869
Mean   :18596  Mean   :26036  Mean   :25646  Mean   :14746
3rd Qu.:23850  3rd Qu.:28194  3rd Qu.:30389  3rd Qu.:18866
Max.   :103007 Max.   :351603 Max.   :272582 Max.   :89837
      Normal2      Normal3
Min.   :   208  Min.   :   15
1st Qu.: 7124  1st Qu.: 8944
Median :14005  Median :17710
Mean   :19425  Mean   :25481
3rd Qu.:21576  3rd Qu.:32191
Max.   :212582 Max.   :244692
```

Transposing Data

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- Need your data to read the other way? Turn it into a matrix, and transpose!
- For example:
 - > **mymatrix** <- **as.matrix(mydf)**
 - > **myTmatrix**<- **t(mymatrix)** #t = transpose
 - > **myTdf** <- **as.data.frame(myTmatrix)** #as data frame again

Plotting with ggplot2

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- To explore later
- Three general components
 - Data set
 - Coordinate system
 - Geoms

Plotting with ggplot2

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- To explore later
- Three general components
 - Data set
 - Coordinate system
 - Geoms

```
> library("ggplot2")
```

```
> data("midwest", package = "ggplot2")
```

```
> gg <- ggplot(midwest, aes(x=area, y=poptotal)) +  
  geom_point(aes(col=state, size=popdensity)) +  
  labs(y="Population",  
        x="Area",  
        title="Scatterplot",  
        caption = "Source: midwest")  
> gg
```

Plotting with ggplot2

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- To explore later
- Three general components
 - Data set
 - Coordinate system
 - Geoms

```
> library("ggplot2")
```

```
> data("midwest", package = "ggplot2")
```

```
> gg <- ggplot(midwest, aes(x=area, y=poptotal)) +  
  geom_point(aes(col=state, size=popdensity)) +  
  labs(y="Population",  
        x="Area",  
        title="Scatterplot",  
        caption = "Source: midwest")  
> gg
```

Plotting with ggplot2

Data Types

Data Wrangling

Operators

Data Import

Workspace

Exercise

- To explore later
- Three general components
 - Data set
 - Coordinate system
 - Geoms

```
> library("ggplot2")
```

```
> data("midwest", package = "ggplot2")
```

```
> gg <- ggplot(midwest, aes(x=area, y=poptotal)) +  
  geom_point(aes(col=state, size=popdensity)) +  
  labs(y="Population",  
        x="Area",  
        title="Scatterplot",  
        caption = "Source: midwest")  
> gg
```

Plotting with ggplot2

Data Types

Data Wrangling

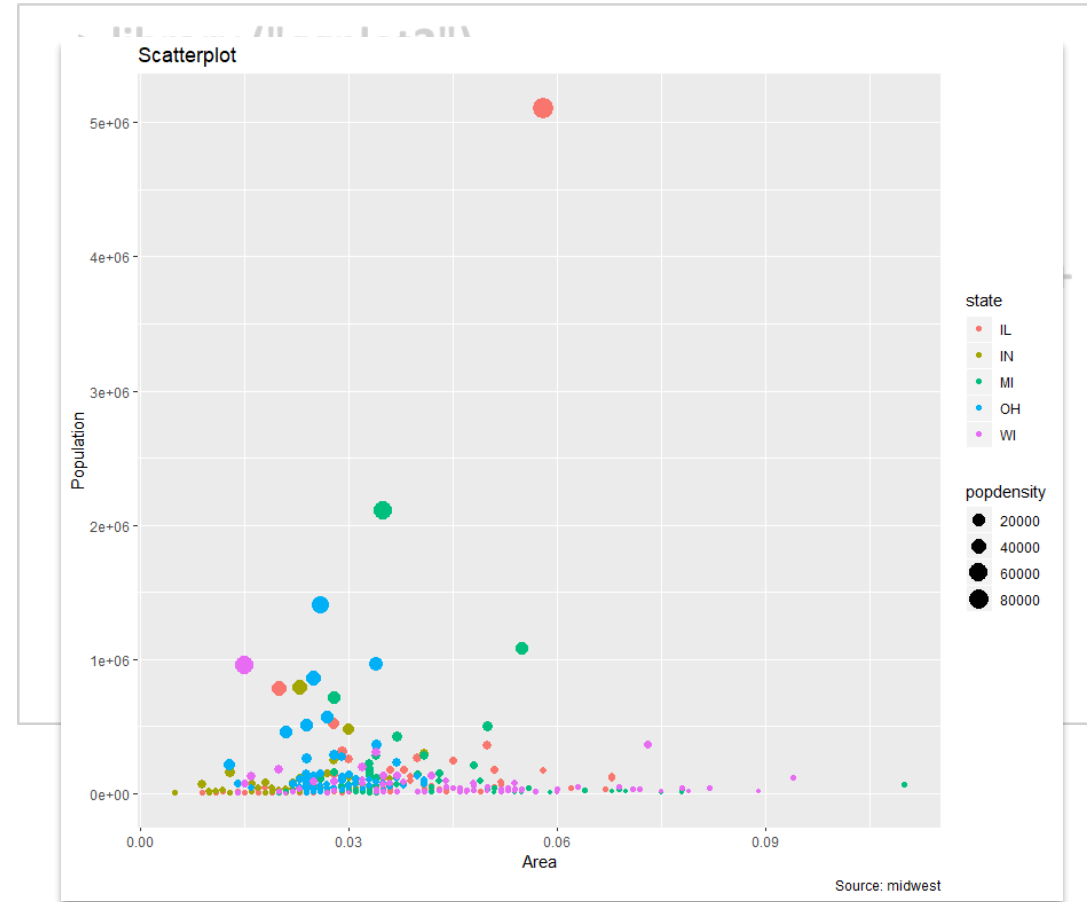
Operators

Data Import

Workspace

Exercise

- To explore later
- Three general components
 - Data set
 - Coordinate system
 - Geoms
- For more info:
Chan Bioinformatics Core –
[GitHub training class](#)



Contact



Email: rchelp@hms.harvard.edu



Website: <https://it.hms.harvard.edu/our-services/research-computing>

Wiki: <https://wiki.rc.hms.harvard.edu/display/O2>



Phone: 617-432-2000 (HMS IT Service Desk, 8a-5p)



Twitter: @hms_rc



Location: Gordon Hall 500, 5th Floor, 25 Shattuck Street

- <https://rc.hms.harvard.edu/office-hours/> for Zoom web conferencing during remote work



Office hours: Wednesdays 1-3p for pressing needs, but appointments encouraged.