The Elizabeth H. and James S. McDonnell III
McDONNELL GENOME INSTITUTE
at Washington University

GenViz Module 2: Using R for genomic data visualization and interpretation

Malachi Griffith, Obi Griffith, Zachary Skidmore

Genomic Data Visualization and Interpretation
September 11-15, 2017
Berlin

Attribution-ShareAlike 4.0 International (CC BY-SA 4.0)

This is a human-readable summary of (and not a substitute for) the license. Disclaimer.

You are free to:

Share — copy and redistribute the material in any medium or format

Adapt — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Learning objectives of the course

- Module 1: Introduction to genomic data visualization and interpretation
- **Module 2: Using R for genomic data visualization and interpretation**
- Module 3: Introduction to GenVisR
- Module 4: Expression profiling, visualization, and interpretation
- Module 5: Variant annotation and interpretation
- Module 6: Q & A, discussion, integrated assignments, and working with your own data

- Tutorials
  - Provide working examples of data visualization and interpretation
  - Self contained, self explanatory, portable

Learning objectives of module 2

- Review basic R usage
- Learn to use R for basic data manipulation
- Learn to create publication quality graphs to display data
- Learn to create interactive graphics

A brief history of R

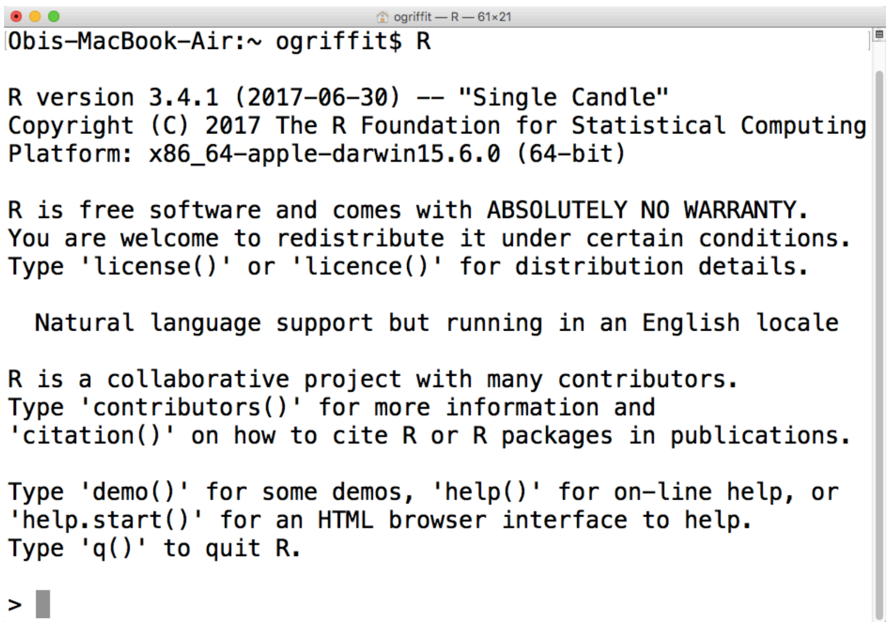
- R is an implementation of the S programming language combined with lexical scoping semantics inspired by Scheme.
- S was created by John Chambers while at Bell Labs
- There are some important differences, but much of the code written for S runs unaltered.
- R was created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand
- Currently developed by the R Development Core Team, of which Chambers is a member.
- The R project was conceived in 1992, with an initial version released in 1995 and a stable beta version in 2000

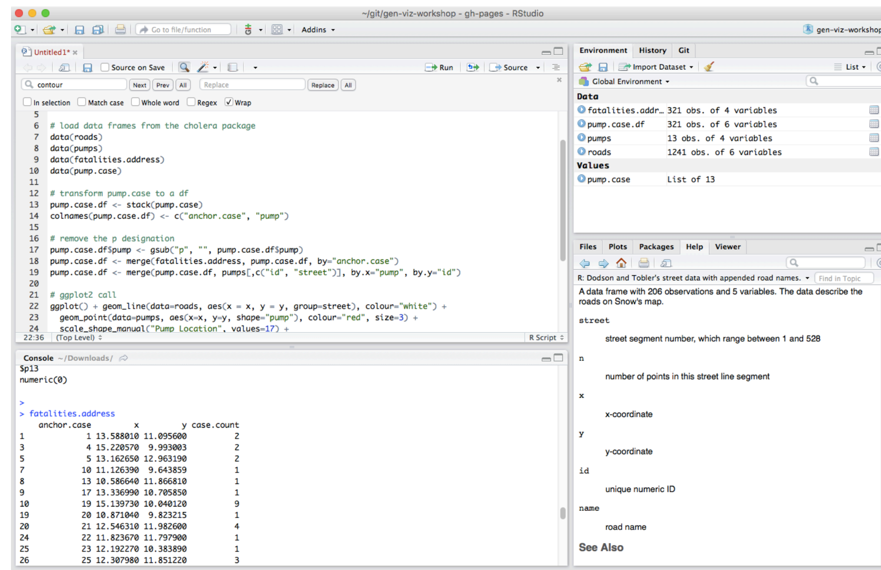
https://en.wikipedia.org/wiki/R_(programming_language)

R is available via command-line or a number of integrated development environments (IDE)

R Project

Rstudio





https://cran.r-project.org/

https://www.rstudio.com/

Open-source, non-profit

Open-source, free + commercial

Installation and versions

- Shiny installation is very simple
- R installation generally only a little more complicated
- Pre-compiled binaries exist for most operating systems

- Be aware of R versions
  - Occasionally some packages may be version dependent or interdependent
  - Less of an issue these days

CRAN and BioConductor

Getting help: ?, vignettes(), and data()

Variables and Data types

- As with any programming language, you need to use various variables to store various information. When you create a variable you reserve some space in memory and keep a record of its location for later retrieval and use.
- The information you wish to store might be characters (e.g., text), integers, boolean (e.g., True/False) etc.
- In contrast to other many programming languages (e.g., C, java, etc) in R, variables are not declared as a specific data type.
    - The variables are assigned with R-Objects and the data type of the R-object becomes the data type of the variable.
    - Frequently used R-Objects include: Vectors, Lists, Matrices, Arrays, Factors, Data Frames
- The simplest R-object is the atomic vector
    - There are six data types for atomic vectors, also termed as six classes of vectors: logical, numerical, integer, complex, character, and raw
- The other R-Objects are built upon atomic vectors
- Lists are also vectors but are not atomic vectors, meaning that they can include multiple data types and can be recursive (contain lists of lists)

<- vs =

Data structures (R objects)

Understanding data and object types with class(), typeof() and is.*()

x	class(x)	typeof(x)	is.*(x)
x <- 1.0	numeric	double	is.numeric(x)=TRUE is.double(x)=TRUE
x <- 1L	integer	integer	is.integer(x)=TRUE
x <- “a”	character	character	is.character(x)=TRUE
x <- TRUE	logical	logical	is.logical(x)=TRUE
x <- charToRaw(“a”)	raw	raw	is.raw(x)=TRUE
x <- 4 + 4i	complex	complex	is.complex(x)=TRUE
x <- matrix(1:4, nrow=2)	matrix	integer	is.matrix(x)=TRUE Is.integer(x)=TRUE
x <- data.frame(x=1:2, y=c(“a”, “b”))	data.frame	list	is.data.frame(x)=TRUE is.list(x)=TRUE

Vectors

Factors

Lists

Attributes

Importing and Exporting Data

Dataframes

Apply functions

Custom functions

Introducing ggplot2

Wide vs long format

melt(data=chrData, id.vars=c("region"))

Long Format

region	variable	value
chr1	gc.content	0.43
chr4	gc.content	0.38
chr8	gc.content	40.00
chr1	mappability	63.00
chr4	mappability	47.00
chr8	mappability	40.00

Legend

- Value
- Variable Name
- ID Variable

region	gc.content	mappability
chr1	0.43	63
chr4	0.38	47
chr8	40.00	40

Wide Format

dcast(data=chrData, region ~ variable, value.var="value")

Graphics options in R

- At least 3 primary graphics options in R
  - base R graphics
    - plot(), par(), etc
  - lattice
  - ggplot2

Why use ggplot2?

base R

ggplot2

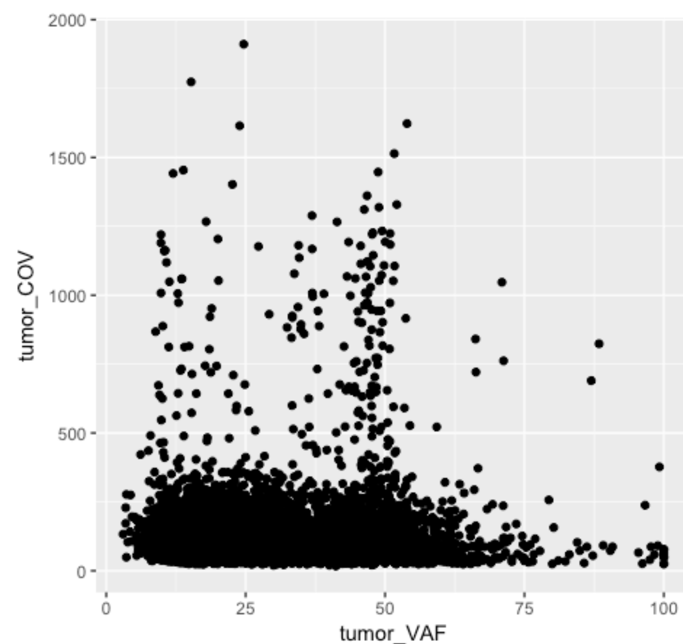
ggplot2 syntax

ggplot(data=variantData, aes(x=tumor_VAF, y=tumor_COV)) + geom_point()

dataframe with data to be plotted

Aesthetic mappings describe how variables in the data are mapped to visual properties (aesthetics) of geometric objects (geoms)

geometric objects specify how data should be plotted



Faceting

Themes

RMarkdown?

Interactive graphics with R shiny

- Optimizing a graphic often requires multiple iterative alterations
- Analysis and interpretation often benefits from active filtering, variable selection, and parameterization
- Interactive graphics allow end-users, especially non-experts, to more effectively explore data
- The R shiny package allows you to quickly and easily create sophisticated web-accessible interactive graphics

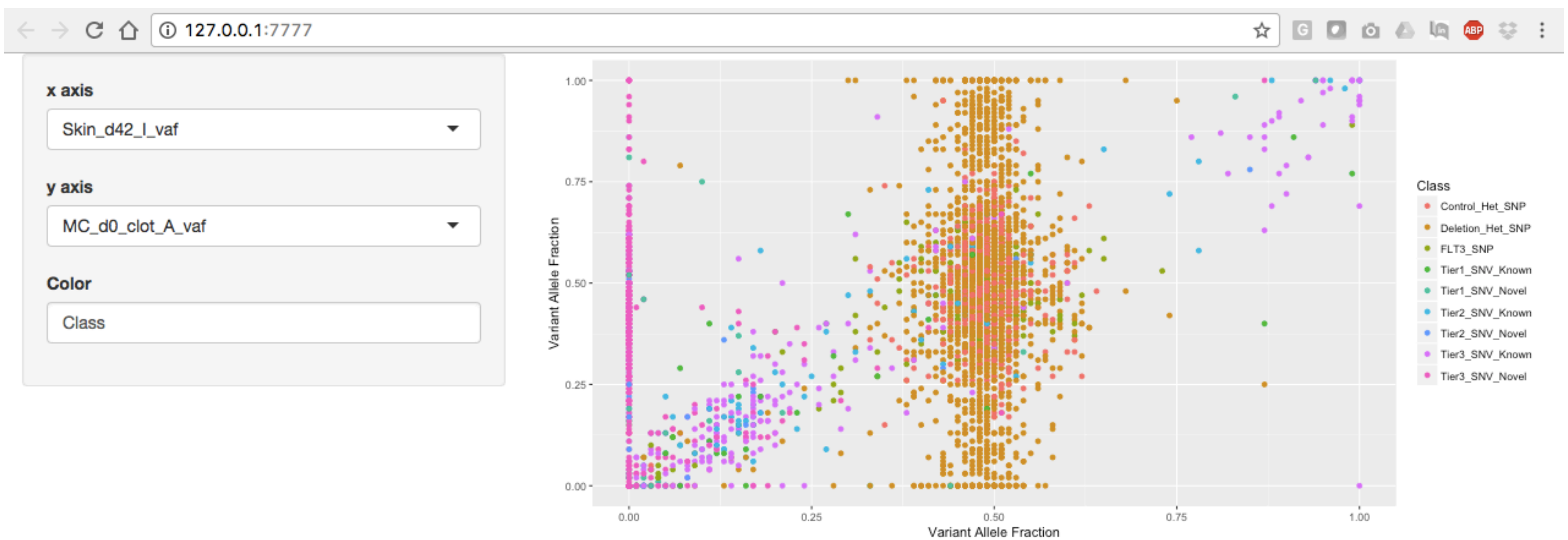
Basic organization of a shiny application

shinyServer
server.R

User
input

R
output

shinyUI
ui.R



Interactive User Interface (UI) = website

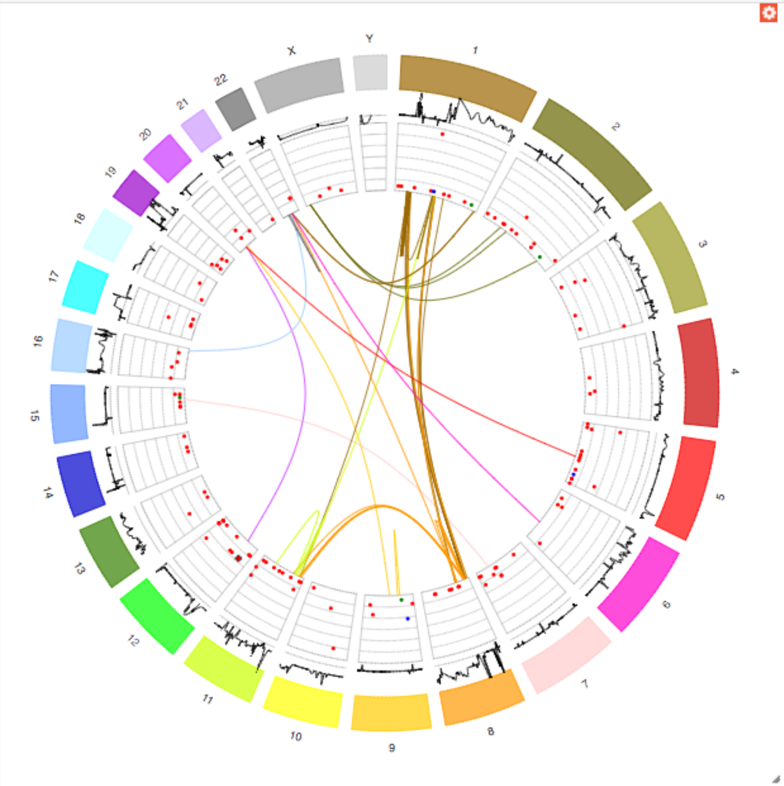
Demo of shiny gallery genomics example

Shiny from RStudio

Back to Gallery

ICGC PANCREATIC CANCER (DUCTAL ADENOCARCINOMA) - GENOME VIEWER

More Information



Cohort Top ClinVar Gene Summary:

HGNC	Chr	Start	From	To	Consequence	Count
SMARCA4	19	11144847	C	T	missense_variant	18
TP53	17	7578437	G	A	stop_gained	18
KRAS	12	25398284	C	T	missense_variant	12
TP53	17	7578437	G	A	exon_variant	10
SMARCA4	19	11144847	C	T	exon_variant	8
TP53	17	7577121	G	A	downstream_gene_variant	6
TP53	17	7577121	G	A	missense_variant	6
KRAS	12	25398285	C	G	missense_variant	4
SMARCA4	19	11144847	C	T	downstream_gene_variant	4
TP53	17	7578437	G	A	downstream_gene_variant	4

Please select a donor ID:

DO49184

SNP Consequences

Resize Factor:

1 5 15

1 3 5 7 9 11 13 15

https://shiny.rstudio.com/gallery/

Questions?