2013 - BMMB 597D: Analyzing Next Generation Sequencing Data

Week 15, Lecture 30

István Albert
Bioinformatics Consulting Center
Penn State

R for high throughput data analysis

- **Strengths**
  - extensive statistical libraries
  - data frames ➔ outstanding data structure
  - allows for “loop-less” transformations and vector indexing
  - it *just works* on every platform with (minimal installation woes)

- **Weaknesses**
  - the environment seems to have some design ambiguities
  - often it is difficult to identify what an object actually is/does
  - silent errors and default behaviors
  - over the decades it has accumulated lots of odd behaviors

Personal observations on bioinformatics R packages

- Often built by novice programmers

- Usually **overspecialized** ➔ work for only one type of problem and with many tacit assumptions

- Are riddled with extremely subtle errors and unexpected behaviors (bug or feature?)

- Many were designed to be a black box

- Exceedingly difficult to trace the chain of events and/or verify that a result is correct!

R and BioConductor
R hero: Hadley Wickham

Amazing R packages that change the way we operate on data
- stringr
- reshape
- ggplot2

What if I told you that you have to pick a side?
Computational tools have an effect on us.
When we pick a tool it will change how we think about problems

Usually people pick one and stick with it

- Python, awk, command line tools, shell – is like being an ant building a solution from small pieces
- You can also use R for all of the above but it will be like re-engineering an existing building into a different one

What you probably cannot do easily is switch between the approaches.

Understand early on: R is idiosyncratic

- There are features that make little sense
- There are operations that are greatly counterintuitive
- There are fundamental flaws in the language
- No one has authority to fix the above

Many (very smart) people deeply dislike this state and therefore avoid R as plague.

Coping mechanism: “That's the way the cookie crumbles”
Particular strengths of R

- processing/filtering/visualizing datasets stored in a TABULAR (row x column) format
- Processing data that easily fits into memory
- Plotting and visualizing data
- Rstudio is an environment around R

Discovery: if you know what an object is you can find out what it does

• ?something → help

Every object in R has a:

- `mode()` → object storage mode
- `class()` → object class
- `typeof()` → internal storage mode

Data representation classes: vector, matrix, list, table data.frame

Data Frames: workhorse of data analysis

```r
# the data frame is a data representation format
annot <- read_delim("annot.csv", comment.char="#", header=FALSE)

# show the names associated with the columns
names(annot)

# you can look at these columns
head(annot)

# you may also assign your own names to these columns
names(annot) <- c("chrom", "source", "type", "start", "end", "value", "strand", "phase", "attr")

# now you can access columns by name
annot$start
```

Other people are so envious of the data frame that they implemented them in other languages Python → Panda (Python Data Analysis Library)

Data Frames are the workhorses of biological data analysis

```r
# now you can access columns by name
annot$start

# set the maximal print to 10 rows
options(max.print=100)

# slice rows, columns
annot[1:3]
annot[1:3, 1:2]

# look at this object annont
annot$chrom

# investigate the objects
class(annot$chrom)
mode(annot$chrom)
typeof(annot$chrom)

# look at head or tail
head(annot$chrom)
```
• All elements of a vector will be of the same mode.
• R will not tell you what an object is, you have to ask is.vector, is.list.
• Indices may be named and the names may be changed later.

Summary: output depends on data type

Factors
Factors represent categorical data; strings in a file are usually loaded directly as factors.

Indexing vectors, sub-setting data frames
Sub-select for genes

```r
> # select rows that contain gene type
genes = subset(gff, type == "gene")
> dim(genes)
[1] 6607   9
> # write the output to a file
> write.table(genes, "genes.gff", col.names=FALSE)
```

We want to match to keep verified strings but the default R string manipulation is lacking.

Install stringr

```r
> install.packages('stringr')
```

...Few if any other programming or data analysis environments have installation processes that are so simple!

Sub-selecting verified genes from the annotation file

```r
> # load up the string matching library
library(stringr)
> # create a new column on gff to be verified
> gffverified = str_detect(gff, attrib, 'Verified')
> # filter for genes
> genes = subset(gff, type == "gene")
> # delete the verified column
> genesverified = NULL
> # write the file
> write.table(genes, "genes.gff", col.names=FALSE)
```

...There are probably many-many other ways to get the same result – shorter, longer, more or less explicit, subtle and not so subtle.

Grouping by Factors

```r
> # read the annotation
gff = read.delim("annot.gff", comment.char="#", header=FALSE)
> # add the column names
names(gff) = c('chr', 'source', 'type', 'start', 'end', 'value',
' strand', 'phase', 'attrib')
> # split the data by the factor stored in column
> names(res) = "binding_site"
> centromere_DNA_Element_1T = "Centromere_DNA_Element_1T"
> centromere_DNA_Element_2T = "Centromere_DNA_Element_2T"
> external_transcribed_exon = "external_transcribed_exon"
> insertion = "insertion"
> intron = "intron"
> # subset data
> res[SARS] = SARS[1:8]
```

Visualizing high dimensionality data

by Hadley Wickham: http://had.co.nz/

**ggplot2**

ggplot2 is a plotting system for R, based on the grammar of graphics, which tries to take the good parts of base and lattice graphics and none of the bad parts. It takes care of many of the fiddly details that make plotting a hassle (like drawing legends) as well as providing a powerful model of graphics that makes it easy to produce complex multi-layered graphics.

A copy of this site in local use is available here, as a 6 meg zip file. To use it, unzip and open the index.html page.

Search the site: [Google](#)

There is nothing like it in any programming environment!

Parts of this presentation follow the tutorial of ggplot2
Getting started with ggplot2

Chapter 2
Getting started with qplot

2.1 Introduction
In this chapter, you will learn to make a wide variety of plots with just few ggplot2 commands. qplot() allows the ggplot() aplha to be
called as a function in a way that is similar to base R. This makes it
easier to use, and it is especially useful for those who are familiar
with base R. However, we will also provide a brief introduction to
the basic features of ggplot2 as these are similar to those in base R.

We will start out with example plots from this manual
Then at the end we generate a peak distribution plot
around gene starts sites.


Install ggplot2

# install the package first
install.packages('ggplot2')

# this needs to be done once per session only
library(ggplot2)

# some libraries come with test data
ggplot has a dataset called diamonds
# you can see the content as a file by executing
table(diamonds, 'diamonds.txt', sep="\t", quote=FALSE)

# you can get some information on the data with
diamonds

diamonds.txt (data comes with ggplot2)

| 1 karat cut color clarity depth table price x y  |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 1   | 0.23| Ideal| E    | SI2 | 61.5| 55  | 326 3|
| 2   | 0.23| Ideal| E    | SI1 | 59.8| 61  | 326 3|
| 3   | 0.20| Premium| E   | VS1 | 56.9| 65  | 334 4|
| 4   | 0.20| Premium| E   | VS2 | 62.7| 58  | 335 4|
| 5   | 0.31| Good  | J    | SI2 | 63.3| 58  | 335 4|
| 6   | 0.24| Very Good| J | VS2 | 62.8| 57  | 337 4|
| 7   | 0.24| Very Good| I   | VS1 | 62.3| 57  | 337 4|
| 8   | 0.26| Very Good| H   | SI1 | 61.9| 55  | 337 4|
| 9   | 0.22| Fair  | E    | VS2 | 65.1| 61  | 338 4|
| 10  | 0.23| Very Good| H | VS1 | 59.4| 61  | 338 4|
| 11  | 0.23| Ideal  | J    | SI2 | 64  | 55  | 339 4|
| 12  | 0.31| Good   | J    | SI1 | 62.2| 54  | 344 4|
| 13  | 0.22| Premium| F   | SI1 | 60.4| 61  | 342 4|
| 14  | 0.31| Ideal  | J    | SI2 | 62.2| 54  | 344 4|
| 15  | 0.23| Good   | J    | SI2 | 64.2| 58  | 345 4|
| 16  | 0.32| Premium| F   | SI2 | 60.4| 62  | 345 4|
| 17  | 0.32| Premium| F   | SI1 | 60.9| 58  | 345 4|
NOTE
For the next few slides I will be changing only line 10
(sometimes we use all data or just the small data)
ggplot2 concepts

• geometry → what plot looks like
• faceting → how many plots/panels
• statistics → transformation on the data
• positioning → fine tunes locations in the plot
• scales → maps data to an x,y coordinate
Faceting and shapes and colors

```r
# two panel plot price vs carat colored by column color
11: res = qplot(carat, price, data=diamonds,
12:   facets=cut-clarity, color=color, alpha=1/10)
```

---

scripts are in supporting data located in the 26.tar.gz file on the website