Introduction to Bioconductor
March 15, 2017
Jenny Drnevich and Jessica Holmes

WELCOME! Please sign in and pick up:

1. iClicker
2. R Cheat Sheet
3. Name tag for back of computer
4. Red and green sticky notes

BOOKMARK the link to workshop wiki:
https://wiki.illinois.edu/wiki/x/LgHJJ
Learning Objectives, Intro to R:

1. Be able to describe **what R is** and how the programming environment works.

2. Be able to **install R and add-on packages** on your own computer.

3. Be able to **read, understand** and write simple R code

4. Know how to get **help**.

5. Be able to describe the **differences** between R, RStudio and Bioconductor
How well do you understand the basics of R after the 1st day of the workshop?

A. Not very well at all - I'm still mostly confused by it
B. I'm starting to be able to follow along
C. I can understand the demo codes pretty well, but remembering what to do on my own is still hard.
D. I was able to answer the practice questions very easily
E. I was pretty comfortable with R already, but I learned some good new things
Did you download R and attempt the homework?

A. No - I was too busy with other work

B. I just downloaded R

C. I started on the practice questions but found it hard to answer much on my own.

D. I was able to answer most of the practice questions and even did another swirl lesson or two

E. I had time to answer all of the questions and do many swirl lessons!
1-minute Brainstorm

Think of at least 5 symbols/characters that are very important in the R language...
Let's start by reviewing the "Understanding R code Cheatsheet"

Link to cheatsheet
How to make sense of this!?!?

First, look for `<-`, typically near the beginning

(or sometimes `=` if not inside parentheses)

If no `<-` or `=`, then not assigning results to anything (e.g., plots, writing to external files)
results.RNA[results.RNA$batch > 1 , 2:3 ] <- log2(matrix(1:10,nrow = 5, ncol = 2)) * test.data$value[i.type == "cDNA"]

<- is the assignment operator, and it means:

___put_results_here___  <- ___compute_this___
Second, look for left parentheses ( the words just to the left are functions:

```
results.RNA[results.RNA$batch > 1 , 2:3 ] <- log2(matrix(1:10,nrow = 5, ncol = 2)) * test.data$value[i.type == "cDNA"]
```

Can look up the help pages for functions:

```
?log2
?matrix
```
results.RNA[results.RNA$batch > 1 , 2:3 ] <- log2(matrix(1:10,nrow = 5, ncol = 2)) * test.data$value[i.type == "cDNA"]

Identify the matching right parenthesis `)` for each left parenthesis; everything in between are the arguments to that function

\[ \text{log2( matrix(1:10, nrow = 5, ncol = 2 ) )} \]

\[ \text{matrix( 1:10, nrow = 5, ncol = 2 )} \]

How many arguments to each function?

A. 0
B. 1
C. 2
D. 3
E. 4

If more than 1 argument, will be separated by commas
results.RNA[results.RNA$batch > 1 , 2:3 ] <- log2(matrix(1:10,nrow = 5, ncol = 2)) * 
    test.data$value[i.type == "cDNA""]

Next look for subsetting, indicated by $, [ or [[]

$ - pulls out a named column from a data frame or named item from a list

[ - starts subsetting, if [ ] either a vector or list, if [ , ] a matrix or data frame, if [ [ ] ] usually a list

results.RNA[results.RNA$batch > 1 , 2:3 ]

test.data$value[ i.type == "cDNA" ]
results.RNA[results.RNA$batch > 1, 2:3] <- log2(matrix(1:10, nrow = 5, ncol = 2)) * test.data$value[i.type == "cDNA"]

Math symbols +, -, *, /, ^ have normal functions:

log2(matrix(1:10, nrow = 5, ncol = 2)) * test.data$value[i.type == "cDNA"]

==, !=, <, > ask for comparisons and yield logical (T/F) results

results.RNA$batch > 1
i.type == "cDNA"

: is a shorthand way to get all integers in a sequence

2:3
1:10

= see detailed explanation at end
Not in this example code, but used often:

%\textbf{x}\%  Special binary operators, \textit{x} can be replaced by any valid name

\texttt{sum( !( rownames( raw.data ) %in% gene.lengths$gene ) )}

\%\textbf{in}\% asks whether the values on the left side are found in the values on the right side, and returns TRUE or FALSE for each value on the left side in a vector.

To look up help for these types of functions, need to enclose in back-ticks (upper left on keyboard, not the single quote):

?`%\textbf{in}\%`

Examples:  1:5 %\textbf{in}\% 4:20

!(1:5 %\textbf{in}\% 4:20)
results.RNA[results.RNA$batch > 1 , 2:3 ] <- log2(matrix(1:10,nrow = 5, ncol = 2)) * test.data$value[i.type == "cDNA"]

Words in quotes are character data only, not objects or functions.

i.type == "cDNA"

What’s left? Any words not followed by a single = is (usually) an object in the workspace:

results.RNA[results.RNA$batch > 1 , 2:3 ]

test.data$value[i.type == "cDNA"]

Words followed by a single = are special cases:

matrix(1:10, nrow = 5, ncol = 2) see next slide…
= is also an assignment operator

(use == for “equal to”)

4 == 2 + 2

When outside (), = is equivalent to <-

a <- log2( 8 )
a = log2( 8 )

When inside (), = either names an argument:

RG <- read.maimages(targets$FileName, source = "genepix", wt.fun = f)

or creates a named column or list item:

Tally <- data.frame(number = 1:5, color = c("black", "red", "green", "blue", "yellow"))
results.RNA[results.RNA$batch > 1 , 2:3 ] <- log2(matrix(1:10,nrow = 5, ncol = 2)) * test.data$value[i.type == "cDNA"]

---

Break it down into separate lines of code:

```r
u <- matrix(1:10,nrow = 5, ncol = 2)
v <- log2( u )
w <- test.data$value
x <- i.type == "cDNA"
y <- w[ x ]
z <- results.RNA$batch > 1

results.RNA[ z , 2:3 ] <- v * y
```
hc.raw <- flashClust(dist(t(log2(pm(raw.demo)))))

Actual line of code from one of my old demos! Take 1-2 min and break it down into separate lines…
magrittr packages’s %>% function

- Similar to pipe function | in Linux
- Explanation taken from Data Wrangling cheat sheet:

%>% passes object on left hand side as first argument (or argument) of function on right hand side.

\[ x \ %>% \ f(y) \quad \text{is the same as} \quad f(x, y) \]

\[ y \ %>% \ f(x, ., z) \quad \text{is the same as} \quad f(x, y, z) \]

Note: if no other arguments given to function, don’t have to put ()!

\[ x \ %>% \ f \quad \text{is the same as} \quad f(x) \]

\[ x \ %>% \ f \ %>% \ g \ %>% \ h \quad \text{is the same as} \quad h(g(f(x))) \]
Learning Objectives for today

1. Be able to describe **what Bioconductor is** and what it does, and how it is different from R/CRAN

2. Be able to **navigate through BioC's package repository** and find the descriptions/examples for each package

3. Understand **what a package vignette is**, where to get them, and how to extract the code from them

4. Be able to describe how **S4 objects** are different than regular (S3) objects and be able to begin manipulating them

5. Be able to **load in a gene model (gtf/gff) file** and explore the information inside
What is Bioconductor?

www.bioconductor.org

• “… open source, open development software project to provide tools for the analysis and comprehension of high-throughput genomic data”

• Primarily based on R language (functions can be in other languages), and run in R environment

• Current release consists of 1294 software packages (sets of functions) for specific tasks

• Also maintains 939 annotation packages for many commercial arrays and model organisms plus 308 experiment data packages
More about Bioconductor

http://bioconductor.org/about/

• Overseen by a core team, mostly located at the Roswell Park Cancer Institute in Buffalo, NY*
  - Provide infrastructure and access to packages
  - Include metadata, annotation and data sets
  - Develop/extend a common software platform to provide interoperability between packages
  - Provide documentation and training

• But majority of software packages contributed by users
  - Any package that is related to genomic data and passes BioC's checks is accepted
  - BioC enforces more rigorous standards than CRAN; git is wild west!

*Had been at Fred Hutchinson Cancer Research Center in Seattle until September 2015
Orchestrating high-throughput genomic analysis with Bioconductor


Affiliations | Corresponding author

Received 30 July 2014 | Accepted 09 December 2014 | Published online 29 January 2015

http://www.nature.com/nmeth/journal/v12/n2/abs/nmeth.3252.html
Navigating Bioconductor's 1294 software packages

http://bioconductor.org/packages/release/BiocViews.html

• BiocViews – allows partitioning of packages by categories

Take a few minutes to investigate the different software packages. Can you find the names of 2 or more packages that might be useful for your research?

How many packages are linked to Gene Set Enrichment?

A. 13
B. 48
C. 74
D. 184
Navigating Bioconductor's 939 Annotation Data packages

http://bioconductor.org/packages/release/BiocViews.html

- Annotation packages also partitioned by categories
- Under PackageType:
  - BSgenome - genome sequences
  - OrgDb - gene annotation packages for species

Does your research organism have any packages in BioC?

A. Yes

B. No
What to do if your organism does not have a pre-built annotation package

See software packages:

1. AnnotationForge - build your own org.*.db package

2. AnnotationHub - access to resources for > 1000 "less-model organisms" from the following databases:


Highly recommend doing AnnotationHub's How To vignette
A package's "landing page"

Click on the link to the **limma** package

- Summary and comparison information
  - platform availability, *popularity*, *support site activity*, age, build OK, *updates*, test coverage
- Description/authors/maintainer/citation
- Documentation/Vignettes - more on this next slide
- Details: biocViews categories, version, related packages
- Package Archives: *Don't download this way until have no other choice; use biocLite() instead!*
Open RStudio and use the examples in the link above to:

1. Open the main PDF explaining how to use these packages and scan the first page or two:
   1. DESeq2
   2. edgeR
2. Extract all of the R codes from the vignettes for the GenomicRanges package
3. Open the R code for GenomicRangesIntroduction.R in RStudio
Remember these R tips for newbies

1. You don't have to understand everything that a code does in order to modify it - just be able to **recognize the part that does need modification**

2. **Copy and paste** working codes, then modify instead of typing all out from scratch

3. When a line of code contains multiple computations, **run each computation separately** to get a better understanding of what it does or why it may be throwing an error
Typical files/tasks for sequencing data and packages to work with them

**Sequencing**
- FASTQ
  - ShortRead, Biostrings, qrqc

**Alignment**
- BAM
  - Rsamtools, GenomicAlignments

**Reduction**
- Counts (.csv)
  - Base R, GenomicAlignments, Rsubread
  - Peaks (.bed, .wig)
    - rtracklayer
  - Variants (.vcf)
    - VariantAnnotation, VariantTools, h5vc

**Analysis**
- Differential expression (genes, transcripts)
  - edgeR, DESeq2, DEXSeq, SGSeq, ...
- Annotation; Differential binding
  - ChIPpeakAnno, DiffBind, ChIPseeker, csaw, ...
- Effect prediction; GWAS
  - ensemblVEP, VariantFiltering, snpStats, ...

**Integration & Visualization**
- IRanges, GenomicRanges, GenomicAlignments
- AnnotationDbi, GenomicFeatures, org.*, TxDb*, biomaRt, PSICQUIC, KEGGREST, ...
- Gvis, ggbio, epivisr, rtracklayer (UCSC), SRAdb (IGV)

Additional Help

• BioC provides some example workflows for analyzing different types of genomic data.
  - Pick a workflow that is of the most interest to you. Write down some of the packages they suggest using.

• BioC runs various training courses and also make the training materials available on the web.
  - Search for CSAMA 2016; I've pulled lots of material from their Day 1 lectures and labs
  - BioC2016 materials are also very useful

• Community-supplied resources and tutorials
Bioconductor Coursera course

https://www.coursera.org/learn/bioconductor

- Part of Johns Hopkins' 7-course Genomic Data Science Specialization
- Can audit each course for free if you don't care about earning a certificate.
- Courses start again Mar 13th and Apr 10th
- Course materials also available without enrolling: http://kasperdanielhansen.github.io/genbioconductor/
Additional Help II

• New Bioconductor support site
  - https://support.bioconductor.org/

  Be sure to read the posting guide before posting!
  http://www.bioconductor.org/help/support/posting-guide/

• R help mailing list
  - https://stat.ethz.ch/mailman/listinfo/r-help

• Google!
BioC 2017: Where Software and Biology Connect

- Bioconductor holds an annual conference for developers and users
- Both lectures and workshops - very useful!
- July 27-28 (Developer Day: July 26), Dana Farber Cancer Institute, Boston, MA
- Registration opening soon…
Create folder to save in

- Open RStudio
- Then type `getwd()` and note the location
- Use File Explorer/Finder to navigate to this directory and create a folder named “introR_sp17”

Download [introBioC_15mar17.zip](https://wiki.illinois.edu/wiki/x/LgHJJ) file from the wiki, move them all to introR_sp17:
S4 objects demo

• In RStudio, open introR_sp17/S4_objects_15mar17.R

• We will work on these together using a gff file downloaded from NCBI
NCBI Genome Resources

- Genomes in .fasta format (sequence of each chromosome)
- Gene model sets in .gff format (start and end locations)
- NCBI actually has multiple copies in multiple locations. Easiest way:

1. Search only genome database
2. Put in common or scientific name
3. Click to search
Document genome AND gene annotation versions

- Gene models updated more frequently than genome, but their file name only has genome version in it!!
How to download

- Don’t just click on the download links!
- Instead, right click on any one and pick “Copy link address”
How to download, continued

Paste address in new tab, but don’t hit enter yet!

Delete file name and hit enter to go to directory instead

Genome .fasta (.fna.gz)
Genome .fasta (.fna.gz) only need this one today. Right-click, select “Save link as” and put in introR_fa17/

Want these also for documentation
GFF/GTF contain genomic locations of genes

- **GFF3** – **Generic feature** format – 9 columns

<table>
<thead>
<tr>
<th>Chromosome ID</th>
<th>Source</th>
<th>Gene feature</th>
<th>Start location</th>
<th>End location</th>
<th>Strand</th>
<th>Score (user defined)</th>
<th>Attributes (hierarchy)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chr1</td>
<td>amel_OGSv3.1</td>
<td>gene</td>
<td>204921</td>
<td>223005</td>
<td>+</td>
<td>.</td>
<td>ID=GB42165</td>
</tr>
<tr>
<td></td>
<td>amel_OGSv3.1</td>
<td>mRNA</td>
<td>204921</td>
<td>223005</td>
<td>+</td>
<td>.</td>
<td>ID=GB42165-RA; Parent=GB42165</td>
</tr>
<tr>
<td></td>
<td>amel_OGSv3.1</td>
<td>3'UTR</td>
<td>222859</td>
<td>223005</td>
<td>+</td>
<td>.</td>
<td>Parent=GB42165-RA</td>
</tr>
<tr>
<td></td>
<td>amel_OGSv3.1</td>
<td>exon</td>
<td>204921</td>
<td>205070</td>
<td>+</td>
<td>.</td>
<td>Parent=GB42165-RA</td>
</tr>
<tr>
<td></td>
<td>amel_OGSv3.1</td>
<td>exon</td>
<td>222772</td>
<td>223005</td>
<td>+</td>
<td>.</td>
<td>Parent=GB42165-RA</td>
</tr>
</tbody>
</table>

- **GTF** – **Gene transfer** format – 9 columns

<table>
<thead>
<tr>
<th>Chromosome ID</th>
<th>Source</th>
<th>Gene feature</th>
<th>Start location</th>
<th>End location</th>
<th>Strand</th>
<th>Score (user defined)</th>
<th>Attributes (non-hierarchy)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB000381</td>
<td>Twinscan</td>
<td>CDS</td>
<td>380</td>
<td>401</td>
<td>+</td>
<td>0</td>
<td>gene_id &quot;001&quot;; transcript_id &quot;001.1&quot;;</td>
</tr>
<tr>
<td>AB000381</td>
<td>Twinscan</td>
<td>CDS</td>
<td>501</td>
<td>650</td>
<td>+</td>
<td>2</td>
<td>gene_id &quot;001&quot;; transcript_id &quot;001.1&quot;;</td>
</tr>
<tr>
<td>AB000381</td>
<td>Twinscan</td>
<td>CDS</td>
<td>700</td>
<td>707</td>
<td>+</td>
<td>2</td>
<td>gene_id &quot;001&quot;; transcript_id &quot;001.1&quot;;</td>
</tr>
<tr>
<td>AB000381</td>
<td>Twinscan</td>
<td>start_codon</td>
<td>380</td>
<td>382</td>
<td>+</td>
<td>0</td>
<td>gene_id &quot;001&quot;; transcript_id &quot;001.1&quot;;</td>
</tr>
<tr>
<td>AB000381</td>
<td>Twinscan</td>
<td>stop_codon</td>
<td>708</td>
<td>710</td>
<td>+</td>
<td>0</td>
<td>gene_id &quot;001&quot;; transcript_id &quot;001.1&quot;;</td>
</tr>
</tbody>
</table>
Need to know what type and attribute to use for alignment and counting

- **GFF3** – Generic feature format – 9 columns

<table>
<thead>
<tr>
<th>Chr</th>
<th>assembly</th>
<th>type</th>
<th>start</th>
<th>stop</th>
<th>strand</th>
<th>ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chr1</td>
<td>amel_OGSv3.1</td>
<td>gene</td>
<td>204921</td>
<td>223005</td>
<td>+</td>
<td>ID=GB42165</td>
<td></td>
</tr>
<tr>
<td>Chr1</td>
<td>amel_OGSv3.1</td>
<td>mRNA</td>
<td>204921</td>
<td>223005</td>
<td>+</td>
<td>ID=GB42165-RA;Parent=GB42165</td>
<td></td>
</tr>
<tr>
<td>Chr1</td>
<td>amel_OGSv3.1</td>
<td>3'UTR</td>
<td>222859</td>
<td>223005</td>
<td>+</td>
<td>Parent=GB42165-RA</td>
<td></td>
</tr>
<tr>
<td>Chr1</td>
<td>amel_OGSv3.1</td>
<td>exon</td>
<td>204921</td>
<td>205070</td>
<td>+</td>
<td>Parent=GB42165-RA</td>
<td></td>
</tr>
<tr>
<td>Chr1</td>
<td>amel_OGSv3.1</td>
<td>exon</td>
<td>222772</td>
<td>223005</td>
<td>+</td>
<td>Parent=GB42165-RA</td>
<td></td>
</tr>
</tbody>
</table>

- **GTF** – Gene transfer format – 9 columns

<table>
<thead>
<tr>
<th>Accession</th>
<th>Species</th>
<th>type</th>
<th>start</th>
<th>stop</th>
<th>strand</th>
<th>ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB000381</td>
<td>Twinscan</td>
<td>CDS</td>
<td>380</td>
<td>401</td>
<td>+</td>
<td>gene_id &quot;001&quot;; transcript_id &quot;001.1&quot;;</td>
<td></td>
</tr>
<tr>
<td>AB000381</td>
<td>Twinscan</td>
<td>CDS</td>
<td>501</td>
<td>650</td>
<td>+</td>
<td>gene_id &quot;001&quot;; transcript_id &quot;001.1&quot;;</td>
<td></td>
</tr>
<tr>
<td>AB000381</td>
<td>Twinscan</td>
<td>CDS</td>
<td>700</td>
<td>707</td>
<td>+</td>
<td>gene_id &quot;001&quot;; transcript_id &quot;001.1&quot;;</td>
<td></td>
</tr>
<tr>
<td>AB000381</td>
<td>Twinscan</td>
<td>start.codon</td>
<td>380</td>
<td>382</td>
<td>+</td>
<td>gene_id &quot;001&quot;; transcript_id &quot;001.1&quot;;</td>
<td></td>
</tr>
<tr>
<td>AB000381</td>
<td>Twinscan</td>
<td>stop.codon</td>
<td>708</td>
<td>710</td>
<td>+</td>
<td>gene_id &quot;001&quot;; transcript_id &quot;001.1&quot;;</td>
<td></td>
</tr>
</tbody>
</table>
How to tell what type and attribute to use for your gtf/gff file?

- Look inside!
  - But using head/less on the command line or notepad can be unreadable
  - Could open in Excel *(no, no, no!!!)*
  - or could use Bioconductor’s rtracklayer package…
Bioconductor S4 objects

- Historically, R wasn't suited to Big Data due to memory and speed limitations.
- Package authors create their own object types (usually S3), which reduces interoperability between packages.
- BioC decided to create well-defined classes and methods to operate on the classes specially designed for sequence data and experimental design.
- Classes can be combined and extended as needed within a particular package, yet still retain inter-operability between packages.

For a more detailed explanation, see:
Navigating S4 objects

- Pre-defined "slots" instead of named items in list
- Instead of $, could use @, but don’t use this method
- Use accessor functions/methods to access the data. Knowing what methods exit for a particular S4 class takes a bit of sleuthing…
- Class inheritance: new classes can be made by extending current classes, and the methods for the current classes get inherited by the new class
- Where possible, BioC has defined methods so that their S4 objects can be manipulated like similar S3 objects
  - DataFrame ≈ data.frame
  - GRangesList ≈ list
One problem with GFF files...

- STAR has been buggy in our hands when using a .gff file instead of .gtf file
- Counting programs may also have trouble counting at the gene with .gff files if there is not a gene-level attribute in the exon/CDS rows
- The example code shows how check the type and attribute, pull out gene-level IDs and write out a modified gene set in gtf format
Questions? Comments?

If you would like a certificates of attendance, please e-mail Jessica jholmes5@Illinois.edu

Please fill out the evaluation form below; we will also email the link to you: http://bit.ly/2mNS3jX

Thank you!!