Introduction: Advanced R / Bioconductor Workshop on High-Throughput Genetic Analysis

27-28 February 2012

1 Logistics

Scope and structure This advanced *R / Bioconductor* workshop is a unique combination of expert practitioners and advanced students working together to explore statistical and bioinformatic issues in analysis of high throughput genomic data. We envision 'high-throughput' to include leading edge sequencing technologies, but also the integrative analysis of diverse data types.

A tentative schedule is in Table˜1. The workshop begins with a review of *Bioconductor* packages, data structures, and resources relevant to high throughput analysis. This portion of the workshop is lead by the Bioconductor core team. The workshop continues with presentations and short exercises on specific statistical and bioinformatic challenges in data analysis and comprehension This portion of the workshop is lead by invited experts from the Bioconductor community. Participants engage in lectures, hands-on exercises, and discussion; be prepared with a modern laptop, R-2.15 (R-devel), and additional software packages to be identified prior to the start of the course.

The cloud For this workshop, we are using *R* and *Bioconductor* hosted in 'the cloud' as Amazon machine instances (AMI). We'll use the cloud throughout the course; you'll be able to continue using your instance after the end of the course, or to install the software and other resources on your laptop or other computer.

Exercise 1
As a first exercise and sanity check, log on to the AMI instance assigned to you. Load the package SeattleAdvancedWorkshop2012, and confirm, using the R function `sessionInfo`*, that your instance is running the 'devel' version of R.*

2 *R / Bioconductor*

Of course, *R* is a statistical programming language. The first part of that moniker, *statistical*, is important because it defines the domain in which *R* will

Table 1: Schedule

Monday
- Morning, 9:00-12:30
  - Data Structures and Packages for Reproducible High-throughput Analysis (Bioconductor Core Team).
- Afternoon, 1:30-5:00
  - Applied ChIP-seq Analysis (Thomas Girke).
  - Isoform-specific RNA-seq (Michael Lawrence).
  - Discussion (All)

Tuesday
- Morning, 9:00-12:30
  - Genetics of Expression and the Implementation of Very High Throughput Statistical Test Procedures (Vincent Carey).
  - Exome sequence analysis (Sean Davis).
- Afternoon, 1:30-5:00
  - Machine Learning for Next-Generation Sequence Analysis (Steve Lianoglou).
  - Biclustering and Analysis of Large Gene Sets (Aedin Culhane).
  - Discussion (All).

be most useful. Obviously, high-throughput genomic data provides ample opportunity for statistical analysis, whether during quality assessment, exploratory analysis, analysis of designed experiments, and visualization and reproducible communication of results. As a *programming language* *R* also provides the facility for scripting (and more!) for automation, reproducibility, and sharing, and for development of novel analytic routines. *R* is an obvious tool for the practicing statistician, data analyst, and bioinformatics professional.

Bioconductor is a collection of *R* packages for the analysis and comprehension of high-throughput genomic data. *Bioconductor* started more than 10 years ago. It gained credibility for its statistically rigorous approach to microarray pre-preprocessing and designed experiments, and integrative and reproducible approaches to bioinformatic tasks. There are now more than 500 *Bioconductor* packages for expression and other microarrays, sequence analysis, flow cytometry, imaging, and other domains. The *Bioconductor* web site provides installation, package repository, help, and other documentation.

So we are on the same page... An *R* user references *R* data objects, and functions that operate on those objects. *R* data objects are generally *vectors* or more elaborate basic types (e.g., `data.frame`, `matrix`) and types and concepts (e.g., `factor` and handling of missing values) that are particularly suited to statistical analysis. More complicated objects can be represented in one of several *class* systems ('S3' and 'S4' are common, S4 is very common in *Bioconductor*). An object is generally manipulated through *methods*, and can be subject to *introspection* to discover appropriate methods and internal structure.

An *R* user typically writes a *script*, using their favorite editor and saved as a plain text file at some location on their disk. The script can be evaluated from the command line, sourced in to *R*, or used in a more interactive manner by sending portions of the script to the *R* interpreter. Scripts can be mature into *vignettes* of *R* code embedded in LaTeX documents (such as these notes!) and formalized into a *package* that can be easily shared with other *R* users.

Getting help Start the *R* help browser with `help.start()`. Search for help on a function with `?read.table`, and for a package with `help(package='IRanges')`. Run the example code on a help page with `example(points)`. Discover vignettes with `vignette(package='SeattleAdvancedWorkshop2012')`; use `vignette('Introduction', package='SeattleAdvancedWorkshop2012')` to view. Use `methods(predict)` to discover which S3 classes have a `predict` method defined, `methods(class='lm')` to find methods defined on S3 class *lm*. For S4, the analogous commands for methods and classes defined in the *GenomicRanges* package are

```
> showMethods('countOverlaps', where=getNamespace('GenomicRanges'))
> showMethods(class='GRanges', where=getNamespace("GenomicRanges"))
```

Tab completion is your friend.

The *Bioconductor* web site The *Bioconductor* web site is at `bioconductor.org`. Features include:

- Brief introductory work flows.
- A manifest of all *Bioconductor* packages arranged alphabetically or as BiocViews.
- Annotation (data bases of relevant genomic information, e.g., Entrez gene ids in model organisms, KEGG pathways) and experiment data (containing relatively comprehensive data sets and their analysis) packages.
- Access to the mailing lists, including searchable archives, as the primary source of help.
- Course and conference information, including extensive reference material.
- General information about the project.
- Information for package developers, including guidelines for creating and submitting new packages.

Key Resources Dalgaard [3] provides an introduction to statistical analysis with *R*. Matloff [5] introduces *R* programming concepts. Chambers [2] provides more advanced insights into *R*. Gentleman [4] emphasizes use of *R* for bioinformatic programming tasks. The *R* web site enumerates additional publications from the user community.

Exercise 2
Scavenger hunt. Spend five minutes tracking down the following information.

Table 2: Core packages for sequence analysis.

Software	
Biostrings	DNA (and other) sequence representation and manipulation, e.g., pattern matching.
GenomicRanges	Representation of genomic coordinates (of annotations, or of aligned sequence reads). *GenomicFeatures* adds specialized functions for working with gene, transcript, exon, coding sequence annotations.
rtracklayer	Common file input / output, e.g., bed, wig, gff.
VariantAnnotation	Input / output and interpretation of VCF (variant call format) files.
Rsamtools	Input / output of BAM (binary alignment) file formats.
ShortRead	Input / output and manipulation of sequence plus quality (fasta, fastq) data.
Annotation	
BSgenome	Whole-genome representations. Annotation packages for common genomes and builds, e.g., *BSgenome.Hsapiens.UCSC.hg19*.
*TxDb.**	Stable data base representation of known gene models, e.g., *TxDb.Hsapiens.UCSC.hg19.knownGene*.
*SNPlocs.**	dbSNP data, e.g., *SNPlocs.Hsapiens.dbSNP.20110815*.
*org.**	Organism-centric gene-level annotations, e.g., *Org.Hs.eg.db*.
GO.db & friends	Pathway, homology, and other gene-level data annotations.
biomaRt	With *rtracklayer*, providing access to ENSEMBL and UCSC resources.

a. *The package containing the* `library` *function.*

b. *The author of the* `alphabetFrequency` *function, defined in the Biostrings package.*

c. *A description of the GappedAlignments class.*

d. *The number of vignettes in the GenomicRanges package.*

e. *From the Bioconductor web site, instructions for installing or updating Bioconductor packages.*

f. *A list of all packages in the current release of Bioconductor.*

g. *The URL of the Bioconductor mailing list subscription page.*

3 Core packages

Table˜2 outlines core packages for representing sequence data, including packages that provide access to static and on-line annotation.

Sequences and qualities Nucleotides and base qualities are important in early stages of high-throughput sequence analysis work flows, for instance during quality assessment or remediation, as well as in some later steps such as variant calling. A common format for this data is as *FASTQ* text files, consisting of multi-line records of identifier, sequence, and per-nucleotide quality scores. The *ShortRead* package can read and representing this data, as developed in the following exercise.

Exercise 3
Load the ShortRead package.

```
> library(ShortRead)
```

Create a character vector containing the path to a fastq file containing 1 million short (37 cycle) read sequences from the 'pasilla' experiment in Drosophila *[1].*

```
> fl <- system.file(package="SeattleAdvancedWorkshop2012Data",
+                   "extdata", "SRR031724_1_subset.fastq",
+                   mustWork=TRUE)
```

Read the data in to R using the `readFastq` *function; display it by enclosing the entire expression in parentheses.*

```
> (fq <- readFastq(fl))
```

Since we're statistician and fastq files can be very large, we might want to draw a random sample instead of taking the whole file (in reality, a million reads is easy to deal with).

```
> sampler <- FastqSampler(fl, n=100000)
> (fqsample <- yield(sampler))
```

The `fq` *and* `fqsample` *objects are instances of S4 classes; they hold a lot of data but don't spew it to the screen. Discover the class of these objects, and consult the corresponding help page for details on manipulation.*

```
> class(fq)
> ?"ShortReadQ-class"
```

For instance, retrieve the sequences from this object and discover operations that can be performed.

```
> reads <- sread(fq)
> class(reads)
> ?"DNAStringSet-class"
```

As a fun example, let's calculate how nucleotide use changes by cycle, using the `alphabetByCycle` *function; introspect the result to determine the type of object that we are dealing with.*

```
> abc <- alphabetByCycle(sread(fq))
> class(abc)
> dim(abc)
> abc[1:5, 1:4]
```

Finally, create a plot of alphabet-by-cycle. Any surprises?

```
> matplot(t(abc[1:4,]), type="l")
```

Transcript annotations Many operations on sequence data involve alignment to a reference genome, and the interpretation of the alignment in terms of known annotations. *Bioconductor* 'TxDb' packages provide one source of genome annotation, as the following exercise explores.

Exercise 4
Load the 'TxDb' package for D.˜melanogaster. *The package contains a single object (named after the package), and we create a convenient alias for it.*

```
> library(TxDb.Dmelanogaster.UCSC.dm3.ensGene)
> txdb <- TxDb.Dmelanogaster.UCSC.dm3.ensGene
```

We can introspect the returned object, and discover interesting methods that we might employ.

```
> class(txdb)
> showMethods(class=class(txdb), where=getNamespace("GenomicFeatures"))
```

Extract exons organized by gene, and interpret the results – a list of 'ranges' describing the limits of each exon, in a strand and chromosome-aware fashion.

```
> (exByGn <- exonsBy(txdb, "gene"))
```

Use `elementLengths` *and* `table` *to summarize the number of elements (exons) in each gene; subset* `exByGn` *to identify the gene with the most exons.*

```
> table(elementLengths(exByGn))
> exByGn[which.max(elementLengths(exByGn))]
```

Select exons from genes with particular FlyBase gene ids. Use `range` *and* `unlist` *to determine the genomic coordinate boundaries of each gene.*

```
> ids <- c("FBgn0002183", "FBgn0003360", "FBgn0025111", "FBgn0036449")
> ex <- unlist(range(exByGn[ids]))
```

Alignments A final ingredient in sequence analysis projects are the reads as aligned to genomes. A very common format for representing aligned reads is as binary alignment (BAM) files. The *Rsamtools* package provides a convenient and flexible way to access all aspects of the aligned reads. Frequently we are interested in only some information in the files, e.g., in ChIP- or RNA-seq we are interested in where reads align, but not in necessarily in the sequence of the reads themselves. The following exercise illustrates how reads can be input and manipulated.

Exercise 5
Load the GenomicRanges package, and consult the help page for the function `readGapppedAlignments`.

```
> library(GenomicRanges)
> ?readGappedAlignments
```

Create a variable pointing to a BAM file containing a subset reads from the pasilla *experiment.*

```
> fl <- system.file(package="SeattleAdvancedWorkshop2012Data",
+                   "extdata", "treated1_subset.bam",
+                   mustWork=TRUE)
```

Create a `param` *object use the function* `ScanBamParam` *and specifying the* `which` *function argument such that it will select only reads in the ranges defined by* `ex`*, created in the previous exercise. Use this to input a portion of the bam file. What is a cigar?*

```
> param <- ScanBamParam(which=ex)
> ga <- readGappedAlignments(fl, param=param)
```

Use the `seqnames` *accessor and the* `table` *function to summarize the reads on chromosomes 3L and X.*

```
> table(seqnames(ga))
```

Use `countOverlaps` *to determine the number of reads overlapping each of the genes in* `ex`.

```
> countOverlaps(ex, ga)
```

As an advanced exercise, use the BSgenome package, the `countOverlaps` *and* `scanBam` *functions, and reads from the BAM file to summarize the relationship between alignments and GC content.*

References

[1] A.˜N. Brooks, L.˜Yang, M.˜O. Duff, K.˜D. Hansen, J.˜W. Park, S.˜Dudoit, S.˜E. Brenner, and B.˜R. Graveley. Conservation of an RNA regulatory map between Drosophila and mammals. *Genome Research*, pages 193–202, 2011.

[2] J.˜M. Chambers. *Software for Data Analysis: Programming with R.* Springer, New York, 2008.

[3] P.˜Dalgaard. *Introductory Statistics with R.* Springer, 2nd edition, 2008.

[4] R.˜Gentleman. *R Programming for Bioinformatics.* Computer Science & Data Analysis. Chapman & Hall/CRC, Boca Raton, FL, 2008.

[5] N.˜Matloff. *The Art of R Programming.* No Starch Pess, 2011.