

Introduction to *R* and *Bioconductor*

Martin Morgan (mtmorgan@fredhutch.org)
Fred Hutchinson Cancer Research Center
Seattle, WA, USA

15 June, 2015

R: Statistical Computing Environment

- ▶ Vectors – logical, integer, numeric, character, ...
 - ▶ `list()` – contains other vectors (recursive)
 - ▶ `factor()`, `NA` – statistical concepts
 - ▶ Can be *named* – `c(Germany=1, Argentina=0)`
- ▶ `matrix()`, `array()` – a vector with a 'dim' attribute.
- ▶ `data.frame()` – like spreadsheets; list of equal length vectors.
 - ▶ Homogenous types within a column, heterogenous types across columns.
 - ▶ An example of an *R class*.
- ▶ Other classes – more complicated arrangement of vectors.
 - ▶ Examples: the value returned by `lm()`; the *DNASTringSet* class used to hold DNA sequences.
 - ▶ plain, 'accessor', 'generic', and 'method' functions
- ▶ Packages – base, recommended, contributed.

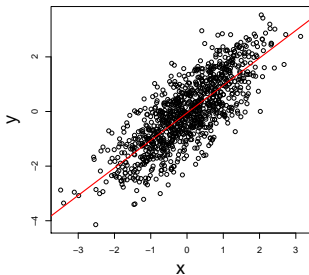
R: Statistical Computing Environment

```
> 1 + 2                                # calculator
[1] 3

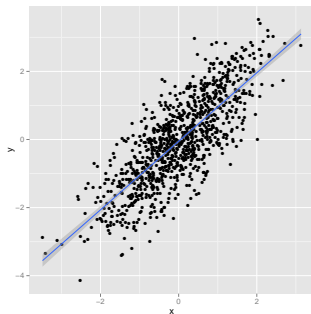
> x <- rnorm(1000)                     # vectors, statistical
> y <- x + rnorm(1000, sd=.8)          # vectorized calculation
> df <- data.frame(x=x, y=y)           # object construction
> fit <- lm(y ~ x, df)                  # linear model, formula
> class(fit)                           # discovery
[1] "lm"
```

R: Statistical Computing Environment

```
> plot(y ~ x, df, cex.lab=2)
> abline(fit, col="red",
+        lwd=2)
```



```
> library(ggplot2)
> ggplot(df, aes(x, y)) +
+   geom_point() +
+   stat_smooth(method="lm")
```



R: programming concepts

- ▶ Functions – built-in (e.g., `rnorm()`); user-defined
- ▶ Subsetting – logical, numeric, character; `df[df$x > 0,]`
- ▶ Iteration – over vector elements, `lapply()`, `mapply()`, `apply()`, ...; e.g., `lapply(df, mean)`

R: help!

- ▶ `?data.frame`
- ▶ `methods(lm)`, `methods(class=class(fit))`
- ▶ `? "plot<tab>"`
- ▶ `help(package="Biostrings")`
- ▶ `vignette(package="GenomicRanges")`
- ▶ [StackOverflow](#); R-help mailing list

“Hey, can you help me with this? I tried...”

Bioconductor

Analysis & comprehension of high-throughput genomic data

- ▶ > 12 years old; 1024 packages; widely used
- ▶ Sequencing (RNAseq, ChIPseq, variants, copy number, ...), microarrays, flow cytometry, proteomics, ...
- ▶ <http://bioconductor.org>,
<https://support.bioconductor.org>

Themes

- ▶ Interoperable – classes to work with genome-scale data, shared (where possible!) across packages
- ▶ Usable – package vignettes, man pages, examples, ...
- ▶ Reproducible – ‘release’ and ‘devel’ versions, updated every 6 months

Bioconductor: GenomicRanges

```
> gr = exons(TxDb.Hsapiens.UCSC.hg19.knownGene); gr
```

GRanges with 289969 ranges and 1 metadata column:

	seqnames	ranges	strand	exon_id
	<Rle>	<IRanges>	<Rle>	<integer>
[1]	chr1	[11874, 12227]	+	1
[2]	chr1	[12595, 12721]	+	2
[3]	chr1	[12613, 12721]	+	3
...
[289967]	chrY	[59358329, 59359508]	-	277748
[289968]	chrY	[59360007, 59360115]	-	277749
[289969]	chrY	[59360501, 59360854]	-	277750

seqinfo: 93 sequences (1 circular) from hg19 genome

GRanges

```
length(gr); gr[1:5]  
seqnames(gr)  
start(gr)  
end(gr)  
width(gr)  
strand(gr)
```

DataFrame

```
mcols(gr)  
gr$exon_id
```

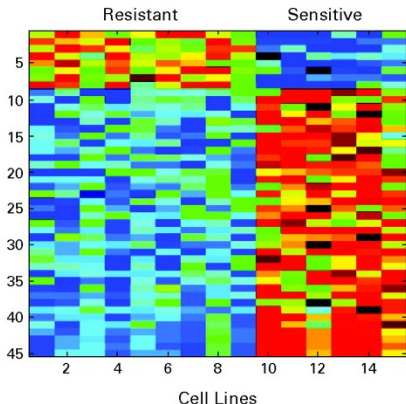
Seqinfo

```
seqlevels(gr)  
seqlengths(gr)  
genome(gr)
```

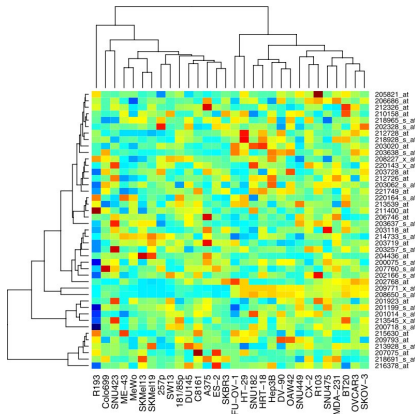
- ▶ Data: aligned reads, called peaks, SNP locations, CNVs, ...
- ▶ Annotation: gene models, variants, regulatory regions, ...
- ▶ `findOverlaps()`, `nearest()`, and many other useful range-based operations.

Bioconductor: SummarizedExperiment motivation

Cisplatin-resistant non-small-cell lung cancer gene sets



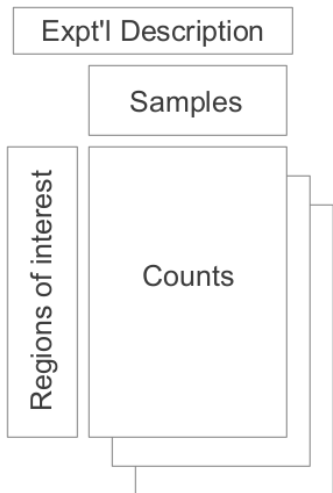
Hsu *et al.* 2007 J Clin Oncol 25:
4350-4357 (retracted)



Baggerly & Coombes 2009 Ann
Appl Stat 3: 1309-1334

Coordinated, programmatic manipulation of feature, sample, and assay data

Bioconductor: SummarizedExperiment



Regions of interest \times samples

- ▶ `assay()` – matrix, e.g., counts of reads overlapping regions of interest.
- ▶ `rowData()` – regions of interest as `GRanges` or `GRangesList`
- ▶ `colData()` – `DataFrame` describing samples.

```
> assay(se)[,se$Treatment == "Control"] # Control counts
```

Bioconductor: a fun demo of *GRanges* interoperability

GenomicFeatures And 'annotation' packages to represent gene models as *GRanges*.

GenomicAlignments To input aligned reads as *GRanges*.

Gviz For visualization.

shiny For interactivity.

Bioconductor: Resources

<http://bioconductor.org>

- ▶ Packages – biocViews, landing pages (e.g., [AnnotationHub](#))
- ▶ Course & conference material; work flows; publications
- ▶ Developer resources

<https://support.bioconductor.org>

- ▶ Question & answer forum for users; usually fast, expert, friendly responses
- ▶ Contributed tutorials, news

Citations

- ▶ Huber et al. (2015) Orchestrating high-throughput genomic analysis with *Bioconductor*. Nature Methods 12:[115-121](#).
- ▶ Lawrence et al. (2013) Software for Computing and Annotating Genomic Ranges. PLoS Comput Biol 9(8):[e1003118](#).