

R / Bioconductor for Integrative Genomic Analysis

Martin Morgan (mtmorgan@fredhutch.org)
Fred Hutchinson Cancer Research Center

15 January 2015

Abstract – *Bioconductor* is a collection of almost 1000 packages for the analysis & comprehension of high-throughput genomic data. This general talk starts with a description of *Bioconductor* principles and their translation to software. We then discuss particular challenges and solutions for applying *R* to large-scale data, and illustrate approaches using the GenomicRanges infrastructure. The presentation concludes with interesting challenges of data integration and analysis facing *R*'s use in emerging areas of genomics and medicine.

Outline: *R* / *Bioconductor* for Integrative Analysis

1. The *Bioconductor* project
2. High-throughput sequencing
3. Genomic Ranges
4. Large data
5. Data integration

Bioconductor

Goal Analysis and comprehension of high-throughput genomic data

- Focus**
- ▶ Sequencing; RNA-Seq, ChIP-Seq, Variants, ...
 - ▶ Expression and other microarrays; flow cytometry; proteomics, imaging

- Themes**
- ▶ Contributions from 'core' members and (primarily academic) user community
 - ▶ Based on the *R* programming language – statistics, visualization, interoperability
 - ▶ Reproducible – data structures, scripts, *vignettes*, packages
 - ▶ Interoperable – formal classes, dependencies on 'core' packages
 - ▶ Open source / open development

Why *Bioconductor*?

A *community* of users and developers.

- ▶ Extensive & interoperable
- ▶ Statistical (volume, technology, experimental design, population samples)
- ▶ Reproducible: long-term, multi-participant science
- ▶ Leading edge: embrace novel technologies and analysis
- ▶ Accessible: affordable, transparent, usable (e.g., vignettes & man pages)

Huber et al., Orchestrating high-throughput genomic analysis with *Bioconductor*. *Nature Methods*: soon!

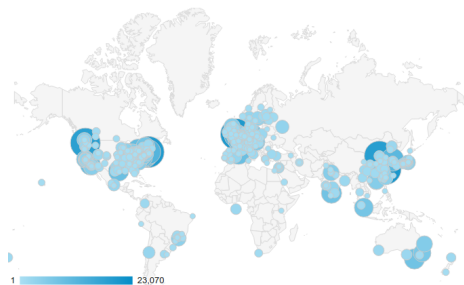
Why *Bioconductor*?

More than a software archive.

- ▶ Build on relevant software, e.g.,
 - ▶ *GenomicRanges* for efficient interoperability; *ExpressionSet* / *SummarizedExperiment* for genetic / phenotypic integration...
 - ▶ I/O via *rtracklayer*, *Rsamtools*, *illuminaio*, ...
 - ▶ Resource access via *biomaRt*, *GEOquery*, ...
- ▶ Commit to long-term support
 - ▶ e.g., *affy* in use 10 years after introduction.
 - ▶ Comprehensive documentation coupled with traditional scientific publications
 - ▶ Engage users via support forum, foster productive collaborations
- ▶ Enable transitions
 - ▶ User to developer
 - ▶ Student to professional

Driving principle: analysis & comprehension of high-throughput genomic data

Project status (December, 2014)



2014 [web site](#) visitors, by city

- ▶ 320,000 unique IP address package downloads / year
- ▶ 1,300 [support site](#) contributors / year, 8,200 visitors / month
- ▶ 10,500 PubMed Central mentions of 'Bioconductor';
≈ 22,000 citations to *Bioconductor* packages
- ▶ At least 12 of 15 initial TCGA publications
- ▶ Funding from US NIH & NSF, and (soon!) EC

High-Throughput Sequencing (HTS)

Questions

- ▶ Which genes are differentially expressed in cancer versus normal tissue?
- ▶ Which transcription factors are regulating gene expression?
- ▶ What single nucleotide polymorphisms are present in a population / associated with a disease?

Sample sizes

- ▶ Designed experiments – e.g., 10's or 100's of samples
- ▶ Cohorts – e.g., 100's or 1000's of patients
- ▶ Populations – 1000's - 10000's of individuals

Attributes

- ▶ 10,000's of genes
- ▶ Millions of variants

HTS: Differential Expression Analysis

E.g., Gene differential expression

- ▶ Human genome: 22 autosomes, 2 sex chromosomes; 3 billion nucleotides of DNA
- ▶ DNA transcribed to mRNA, mRNA translated to proteins
- ▶ A 'gene': known ranges on the genome that encode proteins
- ▶ Roughly, highly expressed genes produce more mRNA

Protocol

- ▶ Isolate mRNA from tissue, reverse-transcribe to cDNA
- ▶ Fragment and then sequence cDNA – 10M - 100M fragments
- ▶ Align sequenced fragments to reference genome
- ▶ Summarize (count) aligned fragments in each gene

HTS: Differential Expression Analysis

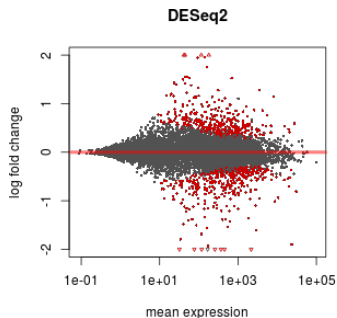
Summarized data

$$\begin{bmatrix} X_{11} & X_{12} & \dots & X_{1n} \\ X_{21} & X_{22} & \dots & X_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ X_{p1} & X_{p2} & \dots & X_{pn} \end{bmatrix}$$

- ▶ Array of counts of reads aligned to p genes in n samples.

Task

- ▶ Fit a linear model to each row, $\text{Count} \sim \text{Treatment}$



HTS: Differential Expression Analysis

Summarized data

$$\begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{p1} & x_{p2} & \dots & x_{pn} \end{bmatrix}$$

- ▶ Array of counts of reads aligned to p genes in n samples.

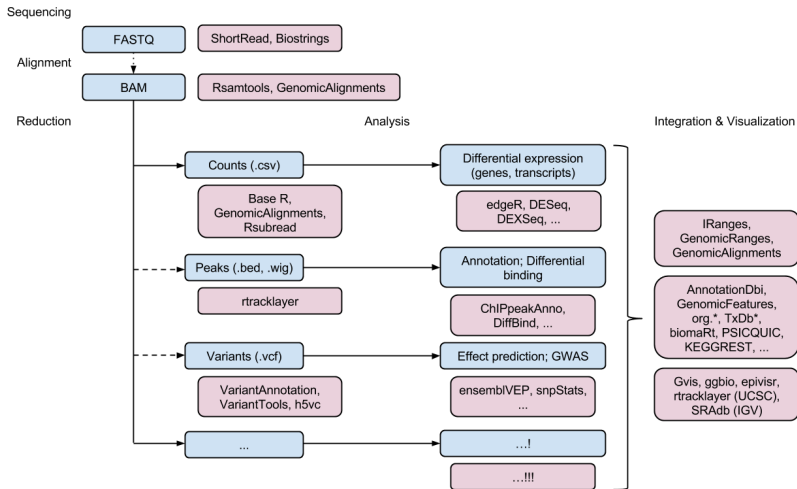
Task

- ▶ Fit a linear model to each row, $\text{Count} \sim \text{Treatment}$

Challenges

- ▶ $p \gg n$
- ▶ Filtering (?)
- ▶ Sample normalization – technical variation between columns
- ▶ Negative binomial error model
- ▶ Shared experimental design – *moderated* test statistics
- ▶ Batch effects

HTS: Package Ecosystem



Genomic Ranges

A central concept

- ▶ Chromosome, start, end, strand provide coordinates specifying where in the genome a range occurs
- ▶ Describes *data*, e.g., aligned reads, and *annotation*, e.g., locations of genes

Many useful operations are based on genomic ranges

- ▶ E.g., reduction of aligned reads to a matrix of counts represents a simple tally of the number of overlaps between genomic ranges describing aligned reads and genomic ranges described gene locations.

Software

- ▶ *GenomicRanges*, *GenomicAlignments*, *GenomicFeatures*
- ▶ *GRanges*, *GRangesList*

Genomic Ranges: GRanges

```
> 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

seqlengths:

chr1	chr2 ...	chrUn_g1000249
249250621	243199373 ...	38502

GRanges

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

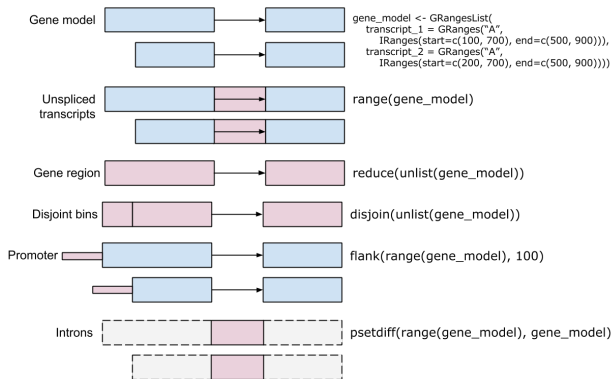
DataFrame

mcols(gr)
gr\$exon_id

Seqinfo

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

Genomic Ranges



A useful [summary table](#) of genomic ranges operations is in PLOS Computational Biology [10.1371/journal.pcbi.1003118](https://doi.org/10.1371/journal.pcbi.1003118).

Genomic Ranges

Building blocks for range-based data structures

- ▶ *GAlignments*, *GAlignmentsList* (*GenomicAlignments*)
- ▶ *SummarizedExperiment* (*GenomicRanges*)
- ▶ *VCF* (*VariantAnnotation*)

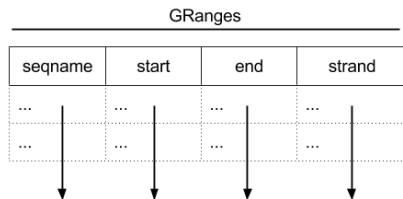
Example

- ▶ What genic regions (coding, intron, 5' or 3' UTR, promoter, ...) do SNPs occur in?

```
library(VariantAnnotation)
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
param <- ScanVcfParam(info=NA, geno=NA)
vcf <- readVcf("my.vcf", "hg19", param)
locateVariants(vcf, TxDb.Hsapiens.UCSC.hg19.knownGene)
```


Genomic Ranges: *GRanges* Implementation

- ▶ Recall: *R* works well on vectors; object creation is expensive
- ▶ *GRanges* class models *columns* of data; one class instance for millions of ranges.
- ▶ Vector-like *API* – `length`, `[[`, `[` returns number and subset of ranges
- ▶ *DataFrame metadata* associated with ranges



Genomic Ranges: *GRangesList*

```
> gr1 = exonsBy(TxDb.Hsapiens.UCSC.hg19.knownGene, "tx", use.names=TRUE); gr1
```

```
GRangesList of length 82960:
```

```
$uc001aaa.3
```

```
GRanges with 3 ranges and 3 metadata columns:
```

	seqnames	ranges	strand	exon_id	exon_name	exon_rank
	<Rle>	<IRanges>	<Rle>	<integer>	<character>	<integer>
[1]	chr1	[11874, 12227]	+	1	<NA>	1
[2]	chr1	[12613, 12721]	+	3	<NA>	2
[3]	chr1	[13221, 14409]	+	5	<NA>	3

```
GRangesList  
(list of GRanges)  
length(gr1)  
gr1[1:3]  
shift(gr1, 1)  
range(gr1)
```

```
$uc010nxq.1
```

```
GRanges with 3 ranges and 3 metadata columns:
```

	seqnames	ranges	strand	exon_id	exon_name	exon_rank
[1]	chr1	[11874, 12227]	+	1	<NA>	1
[2]	chr1	[12595, 12721]	+	2	<NA>	2
[3]	chr1	[13403, 14409]	+	6	<NA>	3

```
GRanges  
gr1[[2]]  
gr1[["uc010nxq.1"]]
```

```
$uc010nxr.1
```

```
GRanges with 3 ranges and 3 metadata columns:
```

	seqnames	ranges	strand	exon_id	exon_name	exon_rank
[1]	chr1	[11874, 12227]	+	1	<NA>	1
[2]	chr1	[12646, 12697]	+	4	<NA>	2
[3]	chr1	[13221, 14409]	+	5	<NA>	3

Two kinds of fun!

```
introns =  
  psetdiff(range(gr1), gr1)
```

```
grr = unlist(gr1)  
## transform grr, then...  
gr1 = relist(grr, gr1)
```

'flesh' 'skeleton'

```
...  
<82957 more elements>
```

```
---
```

```
seqlengths:
```

chr1	chr2 ...	chrUn_gl000249
249250621	243199373 ...	38502

Genomic Ranges: *GRangesList* Implementation

- ▶ List-like where each element of the list is must be a *GRanges*
- ▶ Implementation: a single *GRanges* instance, and a *partitioning* describing how ranges are grouped into list elements
- ▶ Only two objects
- ▶ Some operations can be very fast – unlist, transform, relist.

GRangesList				
GRanges				Partition
seqname	start	end	strand	
...	1
...	1
				2
				2
				2
				3

Strategies for Large Data

Memory management

- ▶ Restrict input to relevant 'columns', e.g., `readGAlignments` inputs only columns necessary to describe geometry of alignment.
- ▶ Select relevant rows, e.g., `ScanBamParam(which=...)`
- ▶ Iterate: read in and operate on successive chunks – e.g.,
`open(BamFile(..., yieldSize=1e7)); reduceByYield(...)`

Speed

- ▶ Efficient *R* code – 10-100× speed-up
- ▶ Parallel evaluation – 2-10× speed-up
- ▶ Often implies memory management
- ▶ *BiocParallel*, *GenomicFiles*

Integrative Analysis: Annotation

- ▶ Gene identifiers (e.g., *org.Hs.eg.db*) and models (e.g., *TxDb.Hsapiens.UCSC.hg19.knownGene*)
- ▶ Web-based resources (e.g., *biomaRt*, *KEGGREST*, *UniProt.ws*)
- ▶ Whole-genome annotations via *AnnotationHub*, e.g., Ensembl, UCSC, *ad hoc*

Integrative Analysis: *AnnotationHub*

File-based resources, e.g., UCSC *liftOver* files

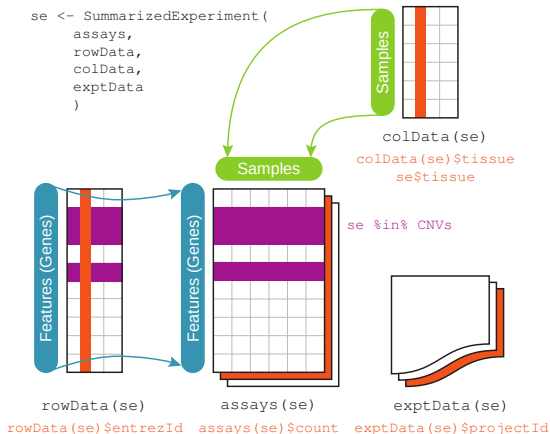
```
## hg19SNPs <- GRanges(...)
library(AnnotationHub)
hub <- AnnotationHub()
chain <- query(hub, 'hg19ToHg38')[[1]]
hg38SNPs <- liftOver(hg19SNPs, chain)
```

Annotation-style resources, e.g., [grasp2](#)

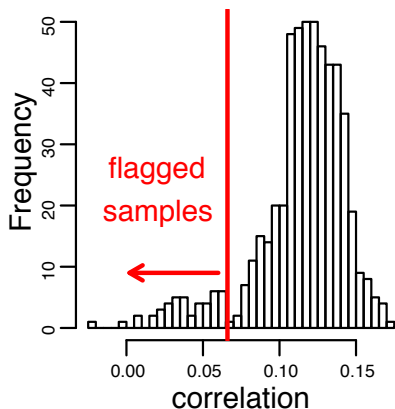
```
library(grasp2db) # Annotation package,
                  # 6 Gb AnnotationHub resource
d <- GRASP2()     # dplyr instance
hispanic <- tbl(d, "count") %>%
  filter(Population=="Hispanic")
semi_join(tbl(d, "variant"), hispanic)
```

Integrative Analysis: *SummarizedExperiment* / *ExpressionSet*

- ▶ Co-ordinate subsetting of 'data' and row (e.g., genomic location) or column (e.g., sample treatment) metadata



Integrative Analysis: Diverse Data Types



TCGA Ovarian gene expression /
copy number correlation

- ▶ Co-ordinated management of diverse data types
- ▶ In-memory and on-disk
- ▶ e.g., Identifier / genomic ranges conversion, `x[i, ,]`
- ▶ e.g., `j = complete.cases(x$mRNA, x$miRNA); x[, j,]`
- ▶ e.g., data type selection, `x[, , c("mRNA", "miRNA")]`
- ▶ Curated collections of public integrated data sets

Future events

- ▶ Computational Statistics for Genome Biology (CSAMA), 15-19 June, Brixen / Bressanone, Italy
- ▶ *useR!*, 1-3 July, Aalborg, Denmark
- ▶ BioC 2015, 20 - 22 July, Seattle, WA USA

Acknowledgments

Core (Seattle): Sonali Arora, Marc Carlson, Nate Hayden, Valerie Obenchain, Hervé Pagès, Paul Shannon, Dan Tenenbaum.

Technical Advisory Board: Vincent Carey, Aedin Culhane, Sean Davis, Robert Gentleman, Kasper Hansen, Wolfgang Huber, Rafael Irizarry, Levi Waldron.

Scientific Advisory Board: Paul Flicek, Simon Tavaré, Simon Urbanek.