April, 18th 2013 – IJM BioInfo Club

An introduction to Bioconductor

OPEN SOURCE SOFTWARE FOR BIOINFORMATICS

Claire Vandiedonck MCF Paris Diderot
claire.vandiedonck@inserm.fr

Université Paris Diderot
Faculté de médecine – Site Villemin
10 avenue de Verdun, 75010 Paris

INSERM UMRS-958
Génétique du diabète

R and bioconductor

R – http://r-project.org

- Open-source, statistical programming language
- Widely used in academia, finance, pharma…
- Core language, 'base' and > 3000 contributed packages
- Interactive sessions, scripts, packages in the CRAN ("Comprehensive R Archive Network")

Bioconductor – http://bioconductor.org

- Analysis and comprehension of high-throughput genomic data
- Open-source & open development software project
- Based primarily on the R programming language
- > 10 years old, > 550 packages

Gentleman RC et al. *Genome Biology* 2004, Vol 5, 10:R80

The Bioconductor project is an initiative for the collaborative creation of extensible software for computational biology and bioinformatics. The goals of the project include: fostering collaborative development and widespread use of innovative software, reducing barriers to entry into interdisciplinary scientific research, and promoting the achievement of remote reproducibility of research results. We describe details of our aims and methods, identify current challenges, compare Bioconductor to other open bioinformatics projects, and provide working examples.

The Bioconductor project

For both statisticians and biologists

- Creation of an extensible software for computational biology and bioinformatics
- Combining computational and statistical needs for many biological processes
- Durable and flexible software development
    - Transparency
    - Reproducibility
    - Efficiency of development
    - Workflows

Organization

- Started in 2001
- Overseen by a core team
- An advisory board
- Annual reports
- Based at the Fred Hutchinson Cancer Research Center
- Mirrors in USA, Brazil, Germany, UK, Japan, China, Australia
- A cloud version even exists!

A united community

Events: courses, developer meetings, workshops

http://www.bioconductor.org/help/events/

Courses materials

http://www.bioconductor.org/help/course-materials/

Two Mailing lists

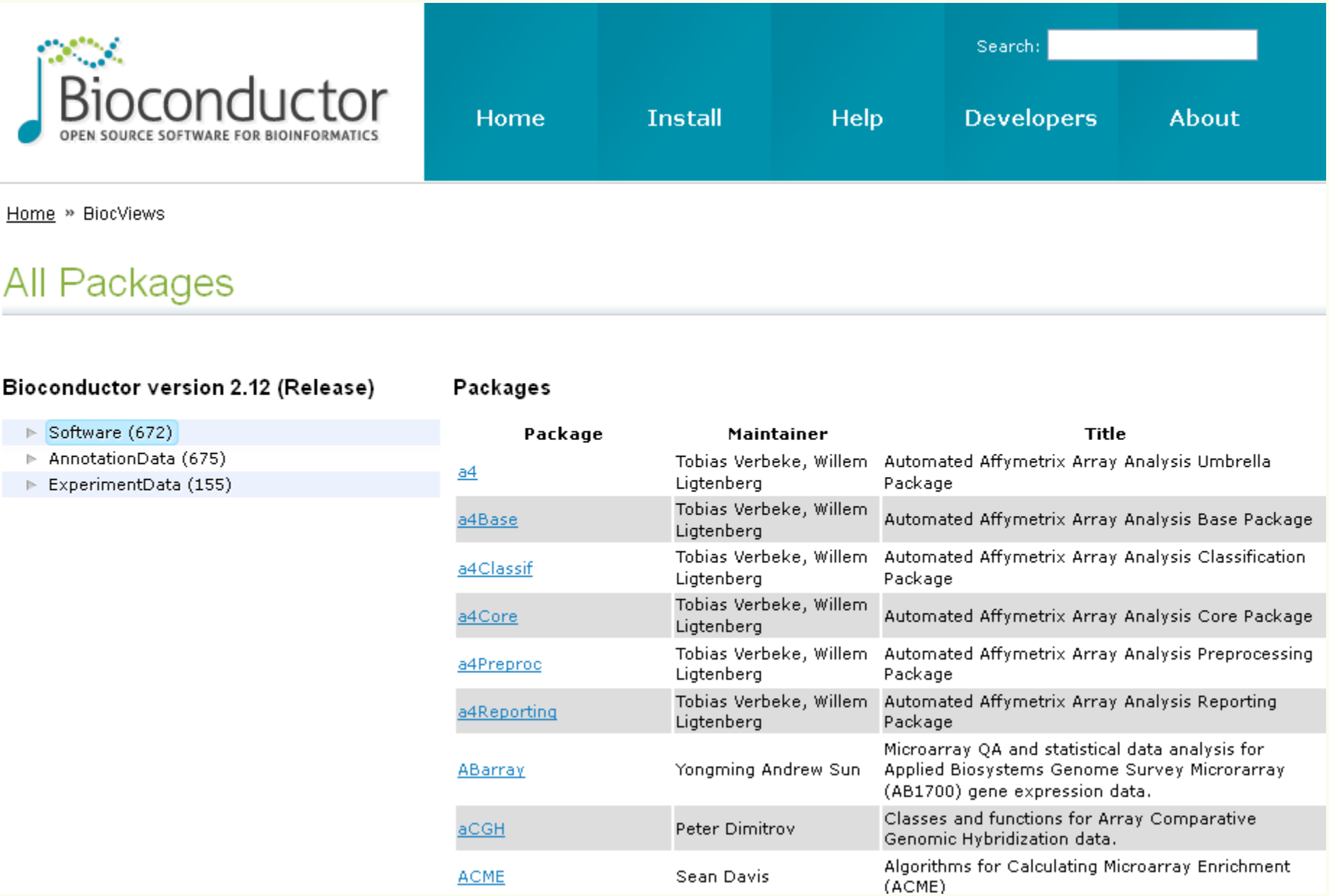
- **Users:** https://stat.ethz.ch/mailman/listinfo/bioconductor
- **Developers:** https://stat.ethz.ch/mailman/listinfo/bioc-devel

And extra sources:

http://manuals.bioinformatics.ucr.edu/home/R_BioCondManual

http://watson.nci.nih.gov/~sdavis/tutorials/

What's in Bioconductor?



Many packages in version 2.12

3 main Components

Software (672)

- Annotation (93)
- AssayDomains (274)
- AssayTechnologies (415)
- Bioinformatics (460)
- BiologicalDomains (102)
- Infrastructure (187)

Annotation Data (675)

- ChipManufacturer (350)
- ChipName (194)
- CustomArray (2)
- CustomCDF (16)
- CustomDBSchema (10)
- FunctionalAnnotation (10)
- Organism (470)
- PackageType (400)
- SequenceAnnotation (2)

Experiment Data (155)

- Cancer (26)
- ChIPchipData (1)
- ChIPseqData (4)
- EColiData (1)
- HapMap (7)
- HighThroughputSequencingData (4)
- HIV (1)
- MassSpectrometryData (6)
- NormalTissue (2)
- RNAExpressionData (6)
- RNAseqData (13)
- StemCells (1)
- Yeast (9)

6 Workflows for

- Oligonucleotide arrays
- High-throughput sequencing
- Annotations
- Variants
- Flow cytometry
- Binding sites of transcription factors

A semi-annual release

Two coexisting versions both designed to work with a specific R version

- a released version
- a development version

Current: Bioconductor 2.12 (2013-04-04) with R 3.0 (2013-04-03)

Previous versions archived for use with Bioconductor (R)

- 2.11 (2.15)
- 2.10 (2.15)
- 2.9 (2.14)
- 2.8 (2.13)
- 2.7 (2.12)
- 2.6 (2.11)
- 2.5 (2.10)

etc…

Installing Bioconcuctor

First install the Bioconductor installer package:

```
source("http://bioconductor.org/biocLite.R")

sessionInfo()
R version 3.0.0 (2013-04-03)
…
other attached packages:
[1] BiocInstaller_1.10.0
```

Then you install the minimum set of packages:

```
biocLite()

BioC_mirror: http://bioconductor.org
Using Bioconductor version 2.12 (BiocInstaller 1.10.0), R version
  3.0.0.
Installing package(s) 'Biobase' 'IRanges' 'AnnotationDbi'
also installing the dependencies 'BiocGenerics', 'DBI', 'RSQLite'
…
package 'BiocGenerics' successfully unpacked and MD5 sums checked
package 'DBI' successfully unpacked and MD5 sums checked
package 'RSQLite' successfully unpacked and MD5 sums checked
package 'Biobase' successfully unpacked and MD5 sums checked
package 'IRanges' successfully unpacked and MD5 sums checked
package 'AnnotationDbi' successfully unpacked and MD5 sums checked
```

A bioconductor package

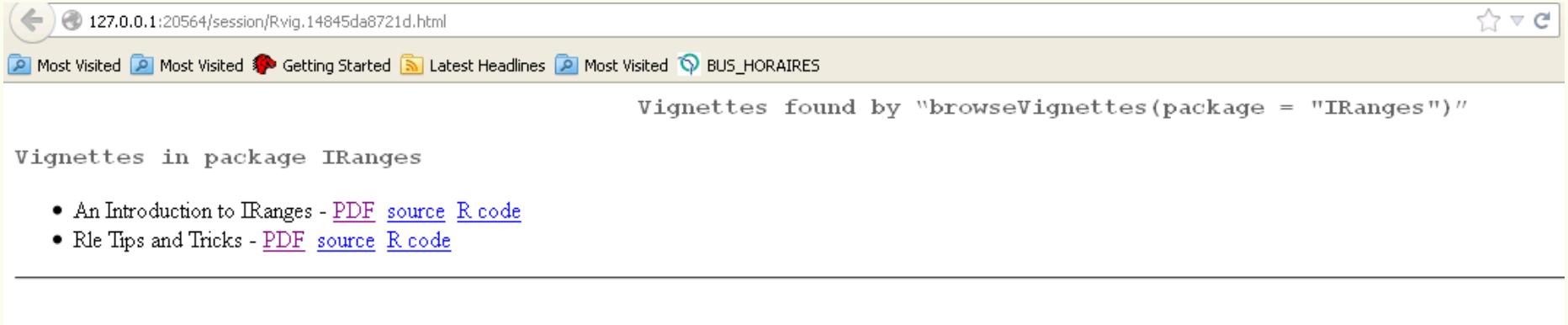
Package documentation

Each Bioconductor package contains at least one vignette = a document that provides a task-oriented description of package functionality

Vignettes contain executable examples and are intended to be used interactively

```
browseVignettes(package = "IRanges")
```

-> opens it a web browser with links to the vignette PDF as well as a plain-text R file containing the code used in the vignette



IRanges

Infrastructure for manipulating intervals on sequences

Bioconductor version: Release (2.12)

The package provides efficient low-level and highly reusable S4 classes for storing ranges of integers, RLE vectors (Run-Length Encoding), and, more generally, data that can be organized sequentially (formally defined as Vector objects), as well as views on these Vector objects. Efficient list-like classes are also provided for storing big collections of instances of the basic classes. All classes in the package use consistent naming and share the same rich and consistent "Vector API" as much as possible.

Author: H. Pages, P. Aboyoun and M. Lawrence

Maintainer: Bioconductor Package Maintainer <maintainer at bioconductor.org>

To install this package, start R and enter:

```
source("http://bioconductor.org/biocLite.R")
biocLite("IRanges")
```

To cite this package in a publication, start R and enter:

```
citation("IRanges")
```

Documentation

PDF	R Script	An Introduction to IRanges
PDF	R Script	Rle Tips and Tricks
PDF		Reference Manual
Text		NEWS

Details

biocViews	DataRepresentation, Infrastructure, Software
Version	1.18.0
In Bioconductor since	BioC 2.3 (R-2.8)
License	Artistic-2.0
Depends	R (>= 2.8.0), methods, utils, stats, BiocGenerics(>= 0.5.6)
Imports	methods, utils, stats, BiocGenerics, stats4
Suggests	GenomicRanges, RUnit, BSgenome.Celegans.UCSC.ce2
System Requirements	
URL	
Imports Me	AnnotationHub, BayesPeak, biomvRCNS, Biostrings, BiSeq, BSgenome, bsseq, bumphunter, casper, ChIPpeakAnno, chipseq, chroGPS, cn.mops, CSAR, DASiR, DECIPHER, deepSNV, DESeq2, DirichletMultinomial, easyRNASeq, epigenomix,

ShortRead, SNPlocs.Hsapiens.dbSNP.20090506, SNPlocs.Hsapiens.dbSNP.20100427, SNPlocs.Hsapiens.dbSNP.20110815, SNPlocs.Hsapiens.dbSNP.20120608, SomatiCA, SplicingGraphs, triform, VariantAnnotation, VariantTools, xmapcore

annmap, AnnotationDbi, ArrayExpressHTS, BayesPeak, Biostrings, BiSeq, BitSeq, CAGEr, cqdv17, charm, ChIPpeakAnno, chipseq, ChIPsim, ChromHeatMap, cn.mops, copynumber, DECIPHER, DESeq2, EDASeq, ensemblVEP, epigenomix, fastseq, flowQ, FunciSNP, gCMAP, qcrma, GenomicFeatures, GenomicRanges, qqbio, qirafe, qmapR, Gviz, HTSeqGenie, MEDIPS, methVisual, methylAnalysis, MethylSeekR, MM, mosaics, motifRG, MotIV, MSnbase, NarrowPeaks, nucleR, oligoClass, OTUbase, pd.081229.hg18.promoter.medip.hx1, pd.2006.07.18.hg18.refseq.promoter, pd.2006.07.18.mm8.refseq.pro, pd.2006.10.31.rn34.refseq.promoter, pd.aq, pd.aragene.1.0.st, pd.aragene.1.1.st, pd.atdschip.tiling, pd.ath1.121501, pd.barley1, pd.bovgene.1.0.st, pd.bovgene.1.1.st, pd.bovine, pd.bsubtilis, pd.cangene.1.0.st, pd.cangene.1.1.st, pd.canine, pd.canine.2, pd.celeg, pd.charm.hg18.example, pd.chicken, pd.citrus, pd.cotton, pd.cyngene, pd.cyngene.1.1.st, pd.cyrgene.1.0.st, pd.cyrgene.1.1.st, pd.cytogenet, pd.drosgenome1, pd.drosophila.2, pd.e.coli.2, pd.ecoli, pd.ecoli.asv2, pd.equgene.1.0.st, pd.equgene.1.1.st, pd.feinberg.hg18.me.hx1, pd.feinberg.mm8.me.hx1, pd.felgene.1.0.st, pd.felgene.1.1.st, pd.genomewidesnp.5, pd.genomewidesnp.6, pd.hc.g110, pd.hg.focus, pd.hg.u133.plus.2, pd.hg.u133a, pd.hg.u133a.2, pd.hg.u133a.tag, pd.h, pd.hg.u219, pd.hg.u95a, pd.hg.u95av2, pd.hg.u95b, pd.hg.u95c, pd.hg, pd.hg.u95e, pd.hg18.60mer.expr, pd.ht.hg.u133.plus.pm, pd.ht.hg.u13, pd.ht.mg.430a, pd.hu6800, pd.huex.1.0.st.v2, pd.hugene.1.0.st.v1, pd.hugene.1.1.st.v1, pd.hugene.2.0.st, pd.hugene.2.1.st, pd.maize, pd.mapping250k.nsp, pd.mapping250k.sty, pd.mapping50k.hind240, pd.mapping50k.xba240, pd.medicago, pd.mg.u74a, pd.mg.u74av2, pd, pd.mg.u74bv2, pd.mg.u74c, pd.mg.u74cv2, pd.mirna.1.0, pd.mirna.2.0, pd.mirna.3.0, pd.mirna.3.1, pd.moe430a, pd.moe430b, pd.moex.1.0.st, pd.mogene.1.0.st.v1, pd.mogene.1.1.st.v1, pd.mouse430.2, pd.mouse, pd.mu11ksuba, pd.mu11ksubb, pd.ovigene.1.0.st, pd.ovigene.1.1.st, pd.pae.g1a, pd.plasmodium.anopheles, pd.poplar, pd.porcine, pd.porge, pd.porgene.1.1.st, pd.rae230a, pd.rae230b, pd.raex.1.0.st.v1, pd.ragene.1.0.st.v1, pd.ragene.1.1.st.v1, pd.rat230.2, pd.rcngene.1.1, pd.rg.u34a, pd.rg.u34b, pd.rg.u34c, pd.rhegene.1.0.st, pd.rhegene.1.1, pd.rhesus, pd.rice, pd.rjpgene.1.1.st, pd.rn.u34, pd.s.aureus, pd.soybe, pd.soygene.1.1.st, pd.sugar.cane, pd.tomato, pd.u133.x3p, pd.vitis.vi, pd.wheat, pd.x.laevis.2, pd.x.tropicalis, pd.xenopus.laevis, pd.yeast.2, pd.yg.s98, pd.zebgene.1.0.st, pd.zebgene.1.1.st, pd.zebrafish, pdInfoB, PICS, prebs, QuasR, R453Plus1Toolbox, Rcade, REDseq, Repitools, r, rMAT, rnaSeqMap, Rolexa, Rsamtools, rSFFreader, RSVSim, rtracklaye, segmentSeq, ShortRead, SNPlocs.Hsapiens.dbSNP.20090506, SNPlocs.Hsapiens.dbSNP.20100427, SNPlocs.Hsapiens.dbSNP.201, SNPlocs.Hsapiens.dbSNP.20110815, SNPlocs.Hsapiens.dbSNP.201, SNPlocs.Hsapiens.dbSNP.20120608, SomatiCA, SplicingGraphs, Tra, triform, TSSi, VanillaICE, VariantAnnotation, VariantTools, waveTiling

Suggests Me	BiocGenerics, HilbertVis, HilbertVisGUI, MiRaGE, Repitools, SNPchip, yeastRNASeq

Package Downloads

Package Source	IRanges_1.18.0.tar.gz
Windows Binary	IRanges_1.18.0.zip (32- & 64-bit)
Mac OS X 10.6 (Snow Leopard)	IRanges_1.18.0.tgz
Package Downloads Report	Download Stats

Installing a bioconductor package

Getting the library path where the package is installed:

```
.libPaths()
[1] "/home/vandiedo/R/x86_64-pc-linux-gnu-library/2.11"
[2] "/usr/local/lib/R/site-library"
...
```

Checking whether the package already exists in the path:

```
list.files(.libPaths()[1])
[1] "AnnotationDbi" "base" "Biobase" "BiocGenerics"  "BiocInstaller"
...
```

Installing the package -> it automatically adapts to your R version

```
source("http://bioconductor.org/biocLite.R")
biocLite("affy")
BioC_mirror: http://bioconductor.org
Using Bioconductor version 2.12 (BiocInstaller 1.10.0), R version 3.0.0.
Installing package(s) 'affy'
also installing the dependencies 'affyio', 'preprocessCore', 'zlibbioc'
...
```

Loading the package from the desired path

```
library(affy,lib.loc=.libPaths()[1])
Loading required package: BiocGenerics
Loading required package: parallel
```

R data types in bioconductor

The main R objects:

- Vectors of logical, integer, numeric, complex, character, raw types
- Statistical concepts such as factors
- More complicated data structure: matrix, data.frame, list

In Bioconductor:

The classes are structured around an object-oriented programming system of formal classes and methods S4 proposed by John Chambers

Why?

Lists are contrived and have limited functionalities

Real object-oriented standards are better, especially with large complex biological data objects

For easier coding, to secure reliable package interoperability

Methods are defined both generically to specify the basic contract and behaviour and specifically to cater for objects of particular classes

The S4 class system

A class provides a software abstraction of a real world object. It reflects how we think about certain objects and what information they should contain

Classes are defined to have specified structures in terms of slots. These are like the components in a list. They contain the relevant data.

An object is an instance of a class

A class defines the structure and inheritance relationships of objects

Implemented in the *methods* R package

```
> sessionInfo()
R version 3.0.0 (2013-04-03)
...
attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

> ls("package:methods")
  [1] "addNextMethod"               "allGenerics"                 "allNames"
  [4] "Arith"                       "as"                          "as<-"
  [7] "asMethodDefinition"          "assignClassDef"              "assignMethodsMetaData"
...
[214] "unRematchDefinition"         "validObject"                 "validSlotNames"
```

Accessing slots

The slots in an object can be accessed in several ways

- Example:

The class for microarray expression data is ExpressionSet

The slot in an Expression Set object containing the matrix of expression values is named exprs

If upp1Eset is an ExpressionSet object, the exprs slot can be accessed by any one of the following:

```
upp1Eset@exprs
exprs(upp1Eset)
slot(upp1Eset, "exprs")
```

slotNames(upp1eset) lists all the slots in this object

Example of an S4 object

```
library(IRanges)
mydata <- IRanges(start=c(101, 25), end=c(110, 80))
mydata
IRanges of length 2
    start end width
[1]   101 110    10
[2]    25  80    56

str(mydata)
Formal class 'IRanges' [package "IRanges"] with 6 slots
  ..@ start          : int [1:2] 101 25
  ..@ width          : int [1:2] 10 56
  ..@ NAMES          : NULL
  ..@ elementType    : chr "integer"
  ..@ elementMetadata: NULL
  ..@ metadata       : list()
```

The Object-Oriented methods

A method is a function that performs an action on an object

Methods define how a particular function should behave depending on the class of its arguments

Methods allow computations to be adapted to particular data types, i.e. classes

Associated to any object is a list of the methods that can be applied to it

The classes and methods implemented in BioC packages can be hard to document, especially when the class hierarchy is complicated

For the end-user: it's mostly transparent. But when something goes wrong, error messages issued by the S4 class system can be hard to understand. Also it can be hard to find the documentation for a specific method

A Practical example

Workflow of affymetrix microarray analysis:

24 arrays:
4 genetic makeups (WT, KO1, KO2, KO3) x 2 tissues (liver or spleen) x 3 mice per group

mouse Gene1.0 ST Arrays
770,317 probes corresponding to 28,853 genes (designed on UCSC mm8, NCBI build 36)

=> We will use microarray (affy, vsn, limma) and annotation (biomaRt) packages to identify differentially expressed genes on the latest annotations (GRCm38, mm10 /Dec 2011)

```
## 1. loading libraries
library(affy)
library(limma)
library(biomaRt)

sessionInfo()
#R version 2.15.0 (2012-03-30)
#Platform: x86_64-pc-linux-gnu (64-bit)
#
#locale:
# [1] LC_CTYPE=en_GB.UTF-8       LC_NUMERIC=C
# [3] LC_TIME=en_GB.UTF-8        LC_COLLATE=en_GB.UTF-8
# [5] LC_MONETARY=en_GB.UTF-8    LC_MESSAGES=en_GB.UTF-8
# [7] LC_PAPER=C                 LC_NAME=C
# [9] LC_ADDRESS=C               LC_TELEPHONE=C
#[11] LC_MEASUREMENT=en_GB.UTF-8 LC_IDENTIFICATION=C
#
#attached base packages:
#[1] stats     graphics  grDevices utils     datasets  methods
base
#
#other attached packages:
#[1] biomaRt_2.14.0     limma_3.14.4       affy_1.36.1
Biobase_2.18.0
#[5] Biostrings_2.26.3  IRanges_1.16.6     BiocGenerics_0.4.0
#
#loaded via a namespace (and not attached):
#[1] affyio_1.24.0         BiocInstaller_1.8.3   parallel_2.15.0
#[4] preprocessCore_1.12.0 RCurl_1.5-0           stats4_2.15.0
#[7] tools_2.15.0          XML_3.9-4             zlibbioc_1.2.0
```

```
## 2. loading libraries
list.files(path="/projects/MHC/Claire/XYZ/First/")## I check the path of the
directory  containing the 24 .CEL files
# [1] "Y_L10_MoGene.CEL" "Y_L11_MoGene.CEL" "Y_L12_MoGene.CEL"
# [4] "Y_L1_MoGene.CEL"  "Y_L2_MoGene.CEL"  "Y_L3_MoGene.CEL"
# [7] "Y_L4_MoGene.CEL"  "Y_L5_MoGene.CEL"  "Y_L6_MoGene.CEL"
#[10] "Y_L7_MoGene.CEL"  "Y_L8_MoGene.CEL"  "Y_L9_MoGene.CEL"
#[13] "Y_S10_MoGene.CEL" "Y_S11_MoGene.CEL" "Y_S12_MoGene.CEL"
#[16] "Y_S1_MoGene.CEL"  "Y_S2_MoGene.CEL"  "Y_S3_MoGene.CEL"
#[19] "Y_S4_MoGene.CEL"  "Y_S5_MoGene.CEL"  "Y_S6_MoGene.CEL"
#[22] "Y_S7_MoGene.CEL"  "Y_S8_MoGene.CEL"  "Y_S9_MoGene.CEL"
```

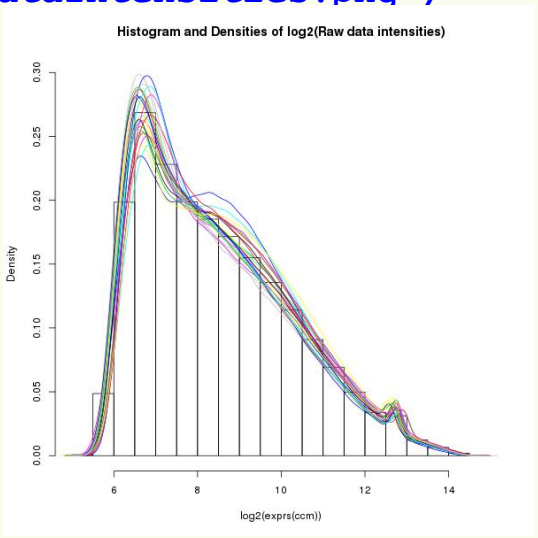
```
Outside of R getting the CDF library files with most recent Ensembl annotations
wget
http://brainarray.mbni.med.umich.edu/Brainarray/Database/CustomCDF/16.0.0/ensg.downl
oad/mogene10stmmensgprobe_16.0.0.tar.gz
wget
http://brainarray.mbni.med.umich.edu/Brainarray/Database/CustomCDF/16.0.0/ensg.downl
oad/mogene10stmmensgprobe_16.0.0.tar.gz
R CMD INSTALL mogene10stmmensgprobe_16.0.0.tar.gz
R CMD INSTALL R CMD INSTALL mogene10stmmensgcdf_16.0.0.tar.gz
```

```
xyz <- ReadAffy(celfile.path ="/projects/MHC/Claire/XYZ/First/", cdfname =
"mogene10stmmensgcdf")
```

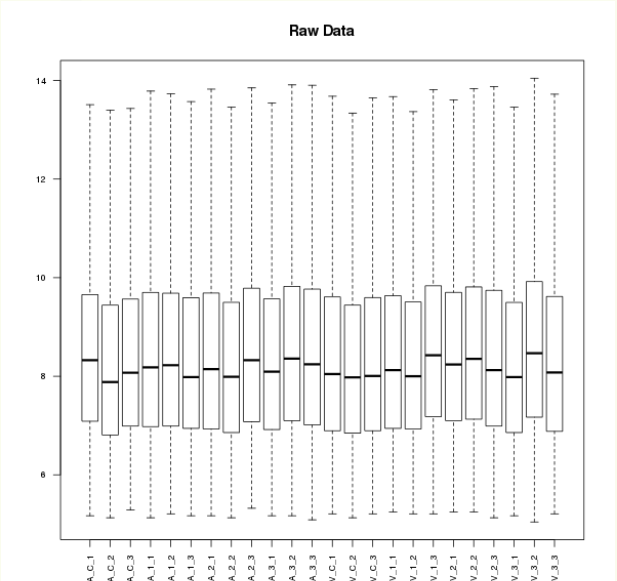
```
dim(exprs(xyz))
#[1] 1102500      24  ## there are 24 arrays and  1102500 probes

head(exprs(xyz))
#  Y_L10_MoGene.CEL   Y_L11_MoGene.CEL   Y_L12_MoGene.CEL
#1             5489               6671               5224
#2              195                191                282
#3             5519               6879               5453
#4              149                193                246
#5              181                101                206
#6              127                123                251 ...

xyz.matrix <- exprs(xyz)

sinfo
#   Array_Identifier Genotype Invalidation     Tissue
#1          Y_L1        C        FALSE        L
#2          Y_L2        1         TRUE        L
#3          Y_L3        2         TRUE        L
#4          Y_L4        3         TRUE        L
#5          Y_L5        C        FALSE        L
#6          Y_L6        1         TRUE        L
#7          Y_L7        2         TRUE        L     ...

# Then I reodered samples logically
```

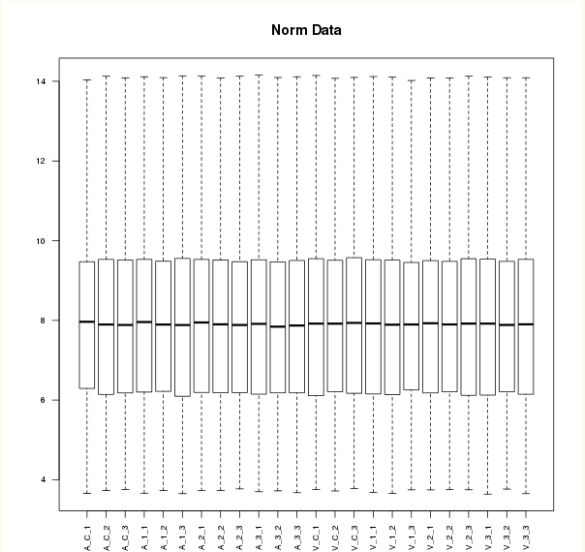
```
## 3.normalization

xyz.rma <- rma(xyz)

dim(exprs(xyz.rma))
#[1] 22312    24

head(exprs(xyz.rma))
#                      Y_L10_MoGene.CEL    Y_L11_MoGene.CEL
#ENSMUSG00000000001_at          11.016042          10.990287
#ENSMUSG00000000003_at           4.259191           4.324463
#ENSMUSG00000000028_at           8.039358           7.947703
#ENSMUSG00000000031_at          10.940150          11.661877
#ENSMUSG00000000037_at           7.845211           7.48773
#ENSMUSG00000000049_at           5.495855           5.591119 ...
```

```
## 4. some QCs
png(height=600, width=600, file="Histogram_Densities_RawDataIntensities.png")
hist(log2(exprs(xyz)), freq=F, ylim=c(0,0.3))
for(i in 2:dim(exprs(xyz))[2]){
lines(density(log2(exprs(xyz))[,i]), col=i, )}
dev.off()
```




```
colours=hsv(seq(0,1,length=24),0.6,1)
filename <- paste("BoxPlots_Raw_WithOutliers_",Sys.Date(),".png", sep="" )
png(height=600, width=600, file=filename, bg="transparent")
boxplot(as.data.frame(log2(xyz.matrix_ordered), na.rm=T), las=2, cex.axis=0.7,
   names=sinfo_ordered2$ID, outline=F, main="Raw Data", col=colours)
dev.off()
```





```
## 5. differential expression

xyz.groups <- paste(sinfo$Genotype, sinfo$Vaisseau, sep = ".")
xyz.groups <- factor(xyz.groups, levels = c("C.L", "K1.L", "K2.L", "K3.L","C.S",
    "K1.S", "K2.S", "K3.S"))


xyz.design <- model.matrix(~0+xyz.groups)
colnames(xyz.design) <- levels(xyz.groups)
xyz.design
#   C.L K1.L K2.L K3.L C.S K1.S K2.S K3.S
#1    1    0    0    0   0    0    0    0
#2    0    1    0    0   0    0    0    0
#3    0    0    1    0   0    0    0    0
#4    0    0    0    1   0    0    0    0
#5    1    0    0    0   0    0    0    0
#6    0    1    0    0   0    0    0    0
#7    0    0    1    0   0    0    0    0
#8    0    0    0    1   0    0    0    0
#9    1    0    0    0   0    0    0    0

...
#24   0    0    0    0   0    0    0    1
#attr(,"assign")
#[1] 1 1 1 1 1 1 1 1
#attr(,"contrasts")
#attr(,"contrasts")$xyz.groups
#[1] "contr.treatment"
```

```
contrast.matrix  <- makeContrasts(
                    KvsC = "(K1.L-C.L)+(K2.L-C.L)+(K3.L-C.L)+(K1.S-C.S)+(K2.S-
   C.S)+(K3.S-C.S)",
                    KvsC.L = "(K1.L-C.L)+(K2.L-C.L)+(K3.L-C.L)",
                    KvsC.S = "(K1.S-C.S)+(K2.S-C.S)+(K3.S-C.S)",
                    K1vsC = "(K1.L-C.L)+(K1.S-C.S)" ,
                    K2vsC = "(K2.L-C.L)+ (K2.S-C.S)",
                    K3vsC = "(K3.L-C.L)+(K3.S-C.S)",
                    K1vsC.L = "(K1.L-C.L)",
                    K2vsC.L = "(K2.L-C.L)",
                    K3vsC.L = "(K3.L-C.L)",
                    K1vsC.S = "(K1.S-C.S)",
                    K2vsC.S = "(K2.S-C.S)",
                    K3vsC.S = "(K3.S-C.S)",
                    LvsV = "(C.L-C.S) + (K1.L-K1.S) + (K2.L-K2.S) + (K3.L - K3.S)",
                    levels = xyz.design
                    )

contrast.matrix
   #    Contrasts
#Levels KvsC KvsC.L KvsC.S K1vsC K2vsC K3vsC K1vsC.L K2vsC.L K3vsC.L K1vsC.S K2vsC.S K3vsC.S LvsV
#  C.L    -3     -3      0    -1    -1    -1      -1      -1      -1       0       0       0    1
#  K1.L    1      1      0     1     0     0       1       0       0       0       0       0    1
#  K2.L    1      1      0     0     1     0       0       1       0       0       0       0    1
#  K3.L    1      1      0     0     0     1       0       0       1       0       0       0    1
#  C.S    -3      0     -3    -1    -1    -1       0       0       0      -1      -1      -1   -1
#  K1.S    1      0      1     1     0     0       0       0       0       1       0       0   -1
#  K2.S    1      0      1     0     1     0       0       0       0       0       1       0   -1
#  K3.S    1      0      1     0     0     1       0       0       0       0       0       1   -1
```

```
fit <- lmFit(matrix.xyz.rma2, design = xyz.design)
fit2 <- contrasts.fit(fit, contrast.matrix)
fit2 <- eBayes(fit2)
```

```
toptable(fit2, genelist=rownames(matrix.xyz.rma2))
#                          ID     logFC         t      P.Value    adj.P.Val         B
#1246  ENSMUSG00000009687_at  3.877709 10.044024 6.118691e-09 0.0001365202 10.079018
#20045 ENSMUSG00000075602_at  4.255918  9.232069 2.283692e-08 0.0002547686  8.980976
#14371 ENSMUSG00000049939_at -2.763483 -8.281176 1.180973e-07 0.0006883368  7.572507
#772   ENSMUSG00000004952_at  6.696365  8.256623 1.234021e-07 0.0006883368  7.534306
#7478  ENSMUSG00000029994_at  1.508051  8.123720 1.567495e-07 0.0006994791  7.325883
#7600  ENSMUSG00000030208_at  2.093850  7.919806 2.272714e-07 0.0008451467  7.000681
#12077 ENSMUSG00000041592_at -1.835874 -7.308785 7.150926e-07 0.0018511419  5.986400
#6325  ENSMUSG00000027962_at  3.220156  7.250085 8.004515e-07 0.0018511419  5.885790
#2237  ENSMUSG00000020038_at -1.962155 -7.146721 9.773770e-07 0.0018511419  5.707271
#19247 ENSMUSG00000073418_at  5.206791  7.119184 1.031034e-06 0.0018511419  5.659421


## and for each contrast...
```

```
## 6. getting corresponding gene names and annotation data
# library(biomaRt)

listMarts()
#                                biomart    version
#1                              ensembl    ENSEMBL GENES 69 (SANGER UK)
#2                                  snp    ENSEMBL VARIATION 69 (SANGER UK)
#3                  functional_genomics    ENSEMBL REGULATION 69 (SANGER UK)
#4                                 vega    VEGA 49  (SANGER UK)
#...

ensembl.mus = useMart("ensembl", dataset = "mmusculus_gene_ensembl")

listAttributes(ensembl.mus)
#                   name                                   description
#1         ensembl_gene_id                                Ensembl Gene ID
#2         ensembl_transcript_id                          Ensembl Transcript ID
#3         ensembl_peptide_id                             Ensembl Protein ID
#4         ensembl_exon_id                                Ensembl Exon ID
# …

listFilters(ensembl.mus)
##             name                                       description
##1           chromosome_name                            Chromosome name
##2           start                                      Gene Start (bp)
##3           end                                        Gene End (bp)
##4           band_start                                 Band Start
##5           band_end                                   Band End
## ...
```

```
for (i in 1:13)
    {
    a <- topTable(fit2, genelist=rownames(matrix.xyz.rma2), coef = i, n = 1E4, p =
   5E-2)
    a$ID <- gsub("_at", "", a$ID)
    if (dim(a)[1] == 0) next
    b <- getBM(attributes = c("ensembl_gene_id", "mgi_symbol", "mgi_id",
   "entrezgene", "chromosome_name", "start_position", "end_position",
   "description"), filters ="ensembl_gene_id", values = a$ID, mart = ensembl.mus)
    m <- merge(a, b, by.x = "ID", by.y = "ensembl_gene_id", sort = F)
    fileName <- paste("output", names(topTable(fit2))[1+i], "txt", sep = ".")
    write.table(m, file = fileName, "quote" = F, row.names = F, sep = "\t")
    }
```