Course › Section 1: Biology b... › A Review of What ... › Gene Expression As...

Gene Expression Assessment

We have shown how Bioconductor provides resources for studying the human genome sequence as well as SNPs. Bioconductor also provides resources that permit us to obtain information about genes. We will see how these databases can be quite complex. But before learning about these here we present some experimental data.

The driving force behind the formation of the Bioconductor project was the emergence of high throughput measurement of gene expression data. Unlike genome sequence and SNP data, gene expression data varies from cell to cell, from tissue to tissue and from individual to individual. Statistical techniques such as those implemented in R were natural tools to parse out variability and perform statistical inference. Furthermore, the ambitious use of newly invented technologies added issues of measurement error and bias to already difficulty challenge.

Note that in the previous assessments we focused on *static* database information: genome sequence and SNPs. In previous courses we have seen the '`tissuesGeneExpression`' data which is experimental data measured with microarrays. If you have not installed it you can do it like this:

```
library(devtools)
install_github("genomicsclass/tissuesGeneExpression")
```

You can load the data:

```
library(tissuesGeneExpression)
data(tissuesGeneExpression)
head(e[,1:5])
table(tissue)
```

The rows of the matrix e are the features, in this case representing genes, and the columns are the samples. The entries of the matrix are gene expression measurements (in log scale) obtained using a microarray technology.

Once the `tissuesGeneExpression` package is loaded, and `data(tissuesGeneExpression)` is run, you have object `e`, `tab`, and `tissue` in your workspace. You can work with them separately but it is preferable in Bioconductor to unify them in an object. In 2017, the preferred object type is SummarizedExperiment. Let's perform the computations and then discuss their utility.

```
library(SummarizedExperiment)
tissSE = SummarizedExperiment(list(rma=e))
colData(tissSE) = DataFrame(tab)
```

Here's a schematic describing the object structure:

SE schema

The storage concept for the assay results is that we have a matrix with rows corresponding to features (in this case, genes) and columns corresponding to samples (in this case, extracts from particular tissues). The basic idea is that we unite metadata about experimental samples (colData) with metadata about features (rowData) and the numerical assay data in a single object. We also define methods such as "[" so that the R expression `X[G, S]` for SummarizedExperiment `X` defines a new SummarizedExperiment with features restricted to those specified in G, and samples restricted to those specified in

S. When we want the numerical values for all features and samples in a SummarizedExperiment `X`, we use `assay(X)`. This approach reduces the risk of mismatches between sample characteristic data and assay data, as selections are coordinated through the underlying code for `[`.

Localization of expression to tissues

0/1 point (graded)

Look at the data for the feature with ID "209169_at". You can index the rows of

```
assay(tissSE)
```

directly with this character string. For example,

```
 mean(assay(tissSE["209169_at",]))
```

is about 7.26

Which of the following best describes the data? (Hint: stratify assay data for the feature by tissue and create boxplots)

○ This is human data and this gene has the same sequence across all tissues thus there is no difference in gene expression

○ This gene is expressed in the brain but not the other tissues

○ This gene is differentially expressed between all tissues

● The individual to individual variability is much larger than the difference between tissues

✗

Submit You have used 2 of 2 attempts

Comparing genes for tissue-specificity

0/1 point (graded)

Below is a vector of 6 IDs which index features of 'tissSE':

IDs = c("201884_at", "209169_at", "206269_at", "207437_at", "219832_s_at", "212827_at")

Which of the following ID(s) appear to represent a gene specific to placenta? Be careful when you are picking, to pick the correct name or names. Names often look similar. Also, if you get your guess wrong, you need to uncheck the ones you think are wrong to guess again.

- □ "201884_at"
- ☑ "209169_at"
- ☑ "206269_at"
- □ "207437_at"
- □ "219832_s_at"

□ "212827_at"

✖

Submit You have used 2 of 2 attempts

✖ Incorrect (0/1 point)

Discovery of microarray annotation in Bioconductor

1/1 point (graded)

Note that there is much existing work on gene function and all we have here are identifiers provided by the manufacturer of the machine that makes the measurements. How would we go about finding more information about gene "206269_at" for example? Does it have a known function? Where is it on the genome? What is its sequence? One of the strengths of Bioconductor is that it connects R, an existing comprehensive toolbox for data analysis, with the existing comprehensive databases annotating the genome. We will learn about these powerful resources in this class.

The microarray product used to make the measurements described here is the Affymetirx Human GeneChip HG133A. Search the Bioconductor website and determine which of the following packages provides a connection to gene information:

○ Biobase

○ simpleaffy

○ hgu133a2cdf

● hgu133a.db

○ affy

✓

Submit You have used 1 of 2 attempts

✓ Correct (1/1 point)

Oligo sequences on affymetrix arrays

1/1 point (graded)

The affymetrix technology for mRNA abundance measurement is based on hybridization of highly processed biological sample material to synthetic oligonucleotide "probes" that are on fixed positions of the microarray surface. Bioconductor provides detailed information on the probe and array structure as published by affymetrix.

Install and attach the hgu133aprobe package.

```
library(BiocInstaller)
biocLite("hgu133aprobe")
library(hgu133aprobe)
> head(hgu133aprobe)
                   sequence   x   y Probe.Set.Name Probe.Interrogation.Position
1 CACCCAGCTGGTCCTGTGGATGGGA 467 181      1007_s_at                         3330
2 GCCCCACTGGACAACACTGATTCCT 531 299      1007_s_at                         3443
3 TGGACCCCACTGGCTGAGAATCTGG  86 557      1007_s_at                         3512
4 AAATGTTTCCTTGTGCCTGCTCCTG 365 115      1007_s_at                         3563
5 TCCTTGTGCCTGCTCCTGTACTTGT 207 605      1007_s_at                         3570
6 TGCCTGCTCCTGTACTTGTCCTCAG 593 599      1007_s_at                         3576
  Target.Strandedness
1           Antisense
2           Antisense
3           Antisense
4           Antisense
5           Antisense
6           Antisense
```

The field "sequence" gives 25 base-pair sequences of oligonucleotides that are in the 3' UTR region of the gene associated with the array "probe set".

You will learn how to use this information to check for accuracy of annotation, to assess risk of cross-hybridization, etc. This table is essentially a large data.frame.

How many oligos are used to interrogate samples for gene GCM1, annotated to probe 206269_at? You will need to work with the Probe.Set.Name field of the hgu133aprobe data.frame.

11 ✓

11

Submit You have used 1 of 5 attempts

✓ Correct (1/1 point)

We'll conclude this series with a quick illustration of annotation enhancement of a SummarizedExperiment.

```
library(hgu133a.db)
sym = mapIds(hgu133a.db, keys=rownames(tissSE), column="SYMBOL", keytype="PROBEID")
nm = mapIds(hgu133a.db, keys=rownames(tissSE), column="GENENAME", keytype="PROBEID")
rowData(tissSE) = DataFrame(symbol=sym, genename=nm)
```

To restrict attention to genes with 'phosphatase' in their names, use code like:

```
tissSE[ grep("phosphatase", rowData(tissSE)$genename), ]
```

Counting features in a SummarizedExperiment

1/1 point (graded)

Set up the rowData for tissSE as noted above.

How many features are annotated to genes with 'kinase' in their name?

1067 ✓

1067

Submit You have used 1 of 5 attempts

✓ Correct (1/1 point)