Bringing Bioconductor to caBIG

M. Morgan (mtmorgan@fhcrc.org), N. Li, S. Falcon
R. Gentleman

http://bioconductor.org

R and Bioconductor

R

- Mature open source project, originating in the mid 1990's
- A *programing language* (elements of *Lisp*, *APL*, ...) for statistical computation and visualization
- ≈ 1000 user-contributed packages

Bioconductor

- R packages focusing on high-throughput biological analysis, e.g., microarray gene expression, proteomics
- ≈ 175 user-contributed packages, active user community, established open-source protocols
- 1000's of potential analytic services

A Bioconductor session

```
> library(affy)

Loading required package: Biobase
Loading required package: tools
Loading required package: affyio

> library(vsn)
> data(sample.exprSet)
> sample.exprSet

Expression Set (exprSet) with
        500 genes
        26 samples
        phenoData object with 3 variables and 26 cases
        varLabels
                sex: Female/Male
                type: Case/Control
                score: Testing Score

> vsnSet <- vsn(exprs(sample.exprSet))
> quantSet <- normalize.quantiles(exprs(sample.exprSet))
```

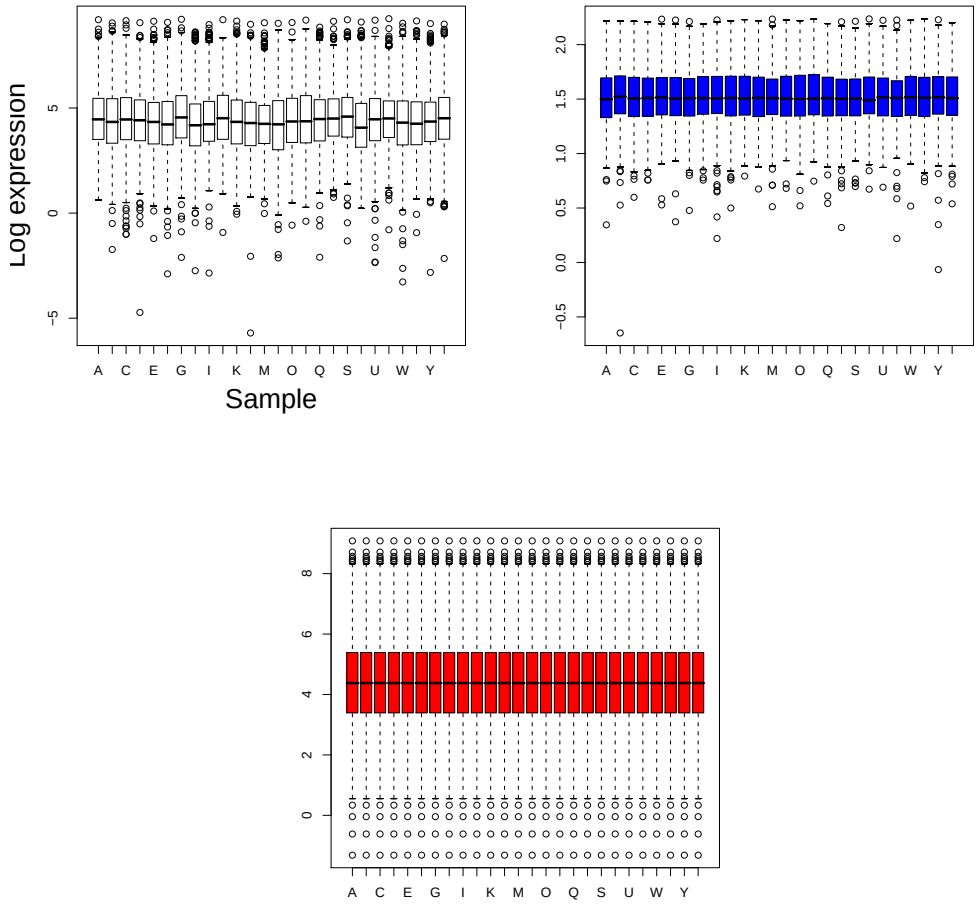

Figure 1: Original, variance-stabilized, and quantile-normalized expression arrays

Bioconductor as analytic service

Challenges

- Objects defined by formal and informal classes
- No argument or return type information
- No robust R web services infrastructure

Overall strategy

- Wrap R data objects and functions in Java
- Make Java available as web services
- Add semantic annotation for caGRID analytic services

From R to Java

TypeInfo

- Add information about argument and return type
- Ensure functions obey type specification

RWebServices

- Java beans represent arbitrary R data objects
- Java methods encapsulate R functions
- Organize objects and methods into class hierarchy
- R $\Leftrightarrow$ Java object conversion

SJava

- R/Java server and client
- Additional R $\Leftrightarrow$ Java object conversion

From Java to (Apache) web service

Starting point

- Java-enabled R objects and functions

Standard Java tools

- Introspection to create WSDL, XSD, etc.
- Installing Java objects as web services
- Finding and invoking web services

Software layers

- Java, SJava, R, Bioconductor

Discussion points: setting the stage

From Java web service to analytic service

- A likely path to exposing analytic service
- What tools will make this step easy?

Semantic annotation of function arguments

- 'Tuning' arguments are numerous and idiosyncratic
- How can these be annotated efficiently?

From web to analytic service

Starting point

- Web services enabled Java

Automate semantic annotation as much as possible

- Introspection to collate arguments
- Programmatic semantic annotation

Challenges to exposing as analytic service

- Need programatic annotation tools
- Must be simple, to encourage annotation of diverse packages

Semantic annotation of arguments

- Recall: *many* functions (e.g., 115 in affy) and packages, each function with several arguments

```
> str(args(vsn))
function (intensities, lts.quantile = 0.5, verbose = interactive(),
    niter = 10, cvg.check = NULL, describe.preprocessing = TRUE,
    subsample, pstart, strata)
```

Arguments like `intensities`

- Matrix of expression values
- Useful for semantic interoperability
- Needs to be described with common vocabulary

Arguments like `lts.quantile`, `niter`, `cvg.check`

- 'Tuning' parameters with precise definitions, but...
- Unlikely to facilitate semantic interoperability

Discussion points

What tools ease the transition from web to analytic service?

- Existing facilities?
- Development of new technologies?

How can we efficiently annotate function arguments?

- Set to default and excluded from web service creation?
- Identify established vocabulary for tuning parameters?
- Find an alternative to semantic annotation for 'one-off' parameters?