Part V. Analysis and presentation via web interfaces

genefilter, **multtest**, and **annotate** packages

Sandrine Dudoit and

Robert Gentleman

Outline

- **`genefilter`** package
- **`multtest`** package
- **`annotate`** package
  - annotation data packages;
  - matching IDs using environments;
  - searching and processing queries from WWW databases
    - LocusLink,
    - GenBank,
    - PubMed;
  - HTML reports.

Combining data across arrays

Data on *G* genes for *n* arrays

→ *G x n* genes-by-arrays data matrix

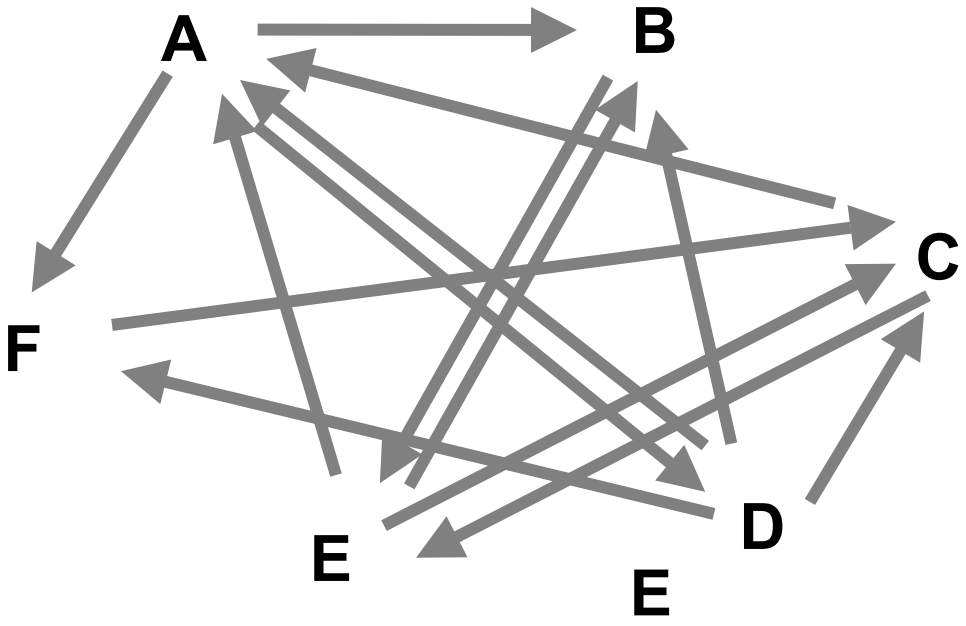
Arrays

Genes		Array1	Array2	Array3	Array4	Array5	...
	Gene1	0.46	0.30	0.80	1.51	0.90	...
	Gene2	-0.10	0.49	0.24	0.06	0.46	...
	Gene3	0.15	0.74	0.04	0.10	0.20	...
	Gene4	-0.45	-1.03	-0.79	-0.56	-0.32	...
	Gene5	-0.06	1.06	1.35	1.09	-1.09	...
	...	...	...	...	...	...	

M = $\log_2$(Red intensity / Green intensity)
expression measure, e.g. RMA.

Combining data across arrays

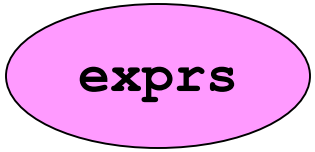
… but the columns have structure, determined by the experimental design.



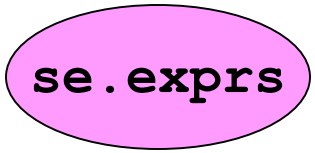
Combining data across arrays

- *cDNA array factorial experiment*. Each column corresponds to a pair of mRNA samples with different drug x dose x time combinations.
- *Clinical trial.* Each column corresponds to a patient, with associated clinical outcome, such as survival and response to treatment.
- Linear models and extensions thereof can be used to effectively combine data across arrays for complex experimental designs.

Biobase: exprSet class

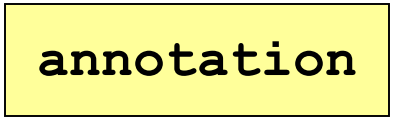


Matrix of expression measures, genes x samples



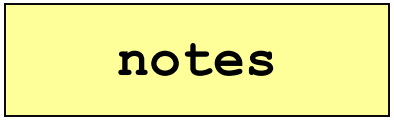
Matrix of SEs for expression measures

Sample level covariates, instance of class **phenoData**



Name of annotation data

Object of class MIAME



Any notes

Gene filtering

- A very common task in microarray data analysis is gene-by-gene selection.
- Filter genes based on
  - data quality criteria, e.g. absolute intensity or variance;
  - subject matter knowledge;
  - their ability to differentiate cases from controls;
  - their spatial or temporal expression pattern.
- Depending on the experimental design, some highly specialized filters may be required and applied sequentially.

Gene filtering

- *Clinical trial.* Filter genes based on association with survival, e.g. using a Cox model.
- *Factorial experiment.* Filter genes based on interaction between two treatments, e.g. using 2-way ANOVA.
- *Time-course experiment*. Filter genes based on periodicity of expression pattern, e.g. using Fourier transform.

`genefilter` package

- The **`genefilter`** package provides tools to sequentially apply filters to the rows (genes) of a matrix.
- There are two main functions, **`filterfun`** and **`genefilter`**, for assembling and applying the filters, respectively.
- Any number of functions for specific filtering tasks can be defined and supplied to **`filterfun`**.

  E.g. Cox model p-values, coefficient of variation.

`genefilter`: separation of tasks

1. Select/define functions for specific filtering tasks.
2. Assemble the filters using the `filterfun` function.
3. Apply the filters using the `genefilter` function → a logical vector, **`TRUE`** indicates genes that are retained.
4. Apply that vector to the **`exprSet`** to obtain a microarray object for the subset of interesting genes.

`genefilter`: supplied filters

Filters supplied in the package

- **`kOverA`** – select genes for which k samples have expression measures larger than A.
- **`gapFilter`** – select genes with a large IQR or gap (jump) in expression measures across samples.
- **`ttest`** – select genes according to t-test nominal p-values.
- **`Anova`** – select genes according to ANOVA nominal p-values.
- **`coxfilter`** – select genes according to Cox model nominal p-values.

`genefilter`: writing filters

- It is very simple to write your own filters.
- You can use the supplied filtering functions as templates.
- The basic idea is to rely on lexical scope to provide values (bindings) for the variables that are needed to do the filtering.

`genefilter`: How to?

1. First, build the filters

```
f1 <- anyNA
f2 <- kOverA(5, 100)
```

2. Next, assemble them in a filtering function

```
ff <- filterfun(f1,f2)
```

3. Finally, apply the filter

```
wh <- genefilter(exprs(DATA), ff)
```

4. Use **`wh`** to obtain the relevant subset of the data

```
mySub <- DATA[wh,]
```

Differential gene expression

- Identify genes whose expression levels are associated with a response or covariate of interest
  - clinical outcome such as survival, response to treatment, tumor class;
  - covariate such as treatment, dose, time.
- Estimation: estimate effects of interest and variability of these estimates.

  E.g. slope, interaction, or difference in means in a linear model.
- Testing: assess the statistical significance of the observed associations.

Multiple hypothesis testing

- When testing for each gene the null hypothesis of no differential expression, e.g. using a t- or F-statistic, two types of errors can be committed.
- Type I error or false positive
  - say that a gene is differentially expressed when it is not,
  - reject a *true null* hypothesis.
- Type II error or false negative
  - fail to identify a truly differentially expressed gene,
  - fail to reject a *false null* hypothesis.

Multiple hypothesis testing

- Large multiplicity problem: thousands of hypotheses are tested simultaneously!
  - Increased chance of false positives.
  - E.g. chance of at least one p-value < α for G independent tests is $1-(1-\alpha)^G$
    and converges to one as G increases.
    For G=1,000 and $\alpha = 0.01$, this chance is 0.9999568!
  - Individual p-values of 0.01 no longer correspond to significant findings.

- Need to adjust for multiple testing when assessing the statistical significance of the observed associations.

Multiple hypothesis testing

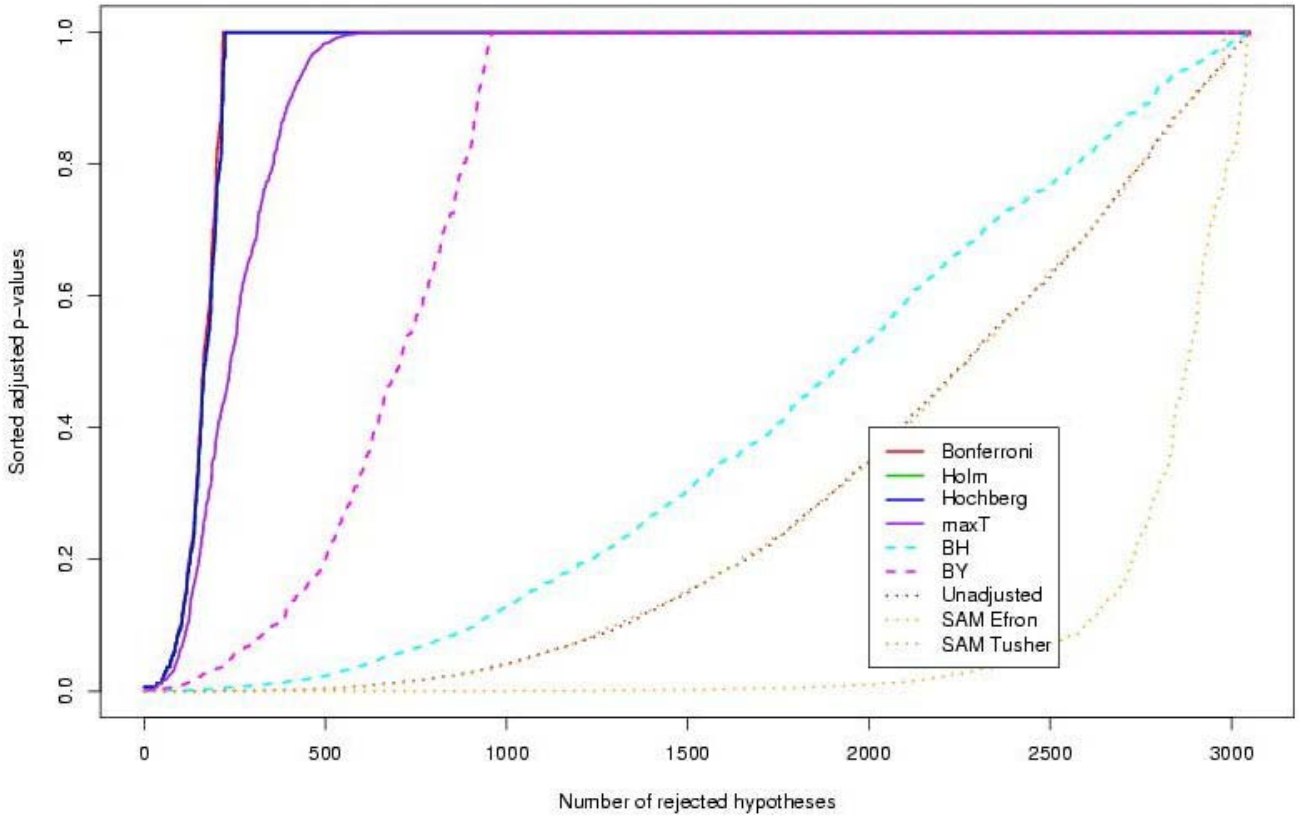
- Define an appropriate Type I error or false positive rate.
- Develop multiple testing procedures that
  - provide strong control of this error rate,
  - are powerful (few false negatives),
  - take into account the joint distribution of the test statistics.
- Report adjusted p-values for each gene which reflect the overall Type I error rate for the experiment.
- Resampling methods are useful tools to deal with the unknown joint distribution of the test statistics.

`multtest` package

- Multiple testing procedures for controlling
  - Family-Wise Error Rate - FWER: Bonferroni, Holm (1979), Hochberg (1986), Westfall & Young (1993) maxT and minP;
  - False Discovery Rate - FDR: Benjamini & Hochberg (1995), Benjamini & Yekutieli (2001).
- Tests based on t- or F-statistics for one- and two-factor designs.
- Permutation procedures for estimating adjusted p-values.
- Fast permutation algorithm for minP adjusted p-values.
- Documentation: tutorial on multiple testing.

`multtest` package

Sorted adjusted p-values for different multiple testing procedures
Golub et al. (1999) ALL AML data



- FWER control
  solid lines
- FDR control
  dashed lines
- PCER control
  dotted lines

`annotate` package

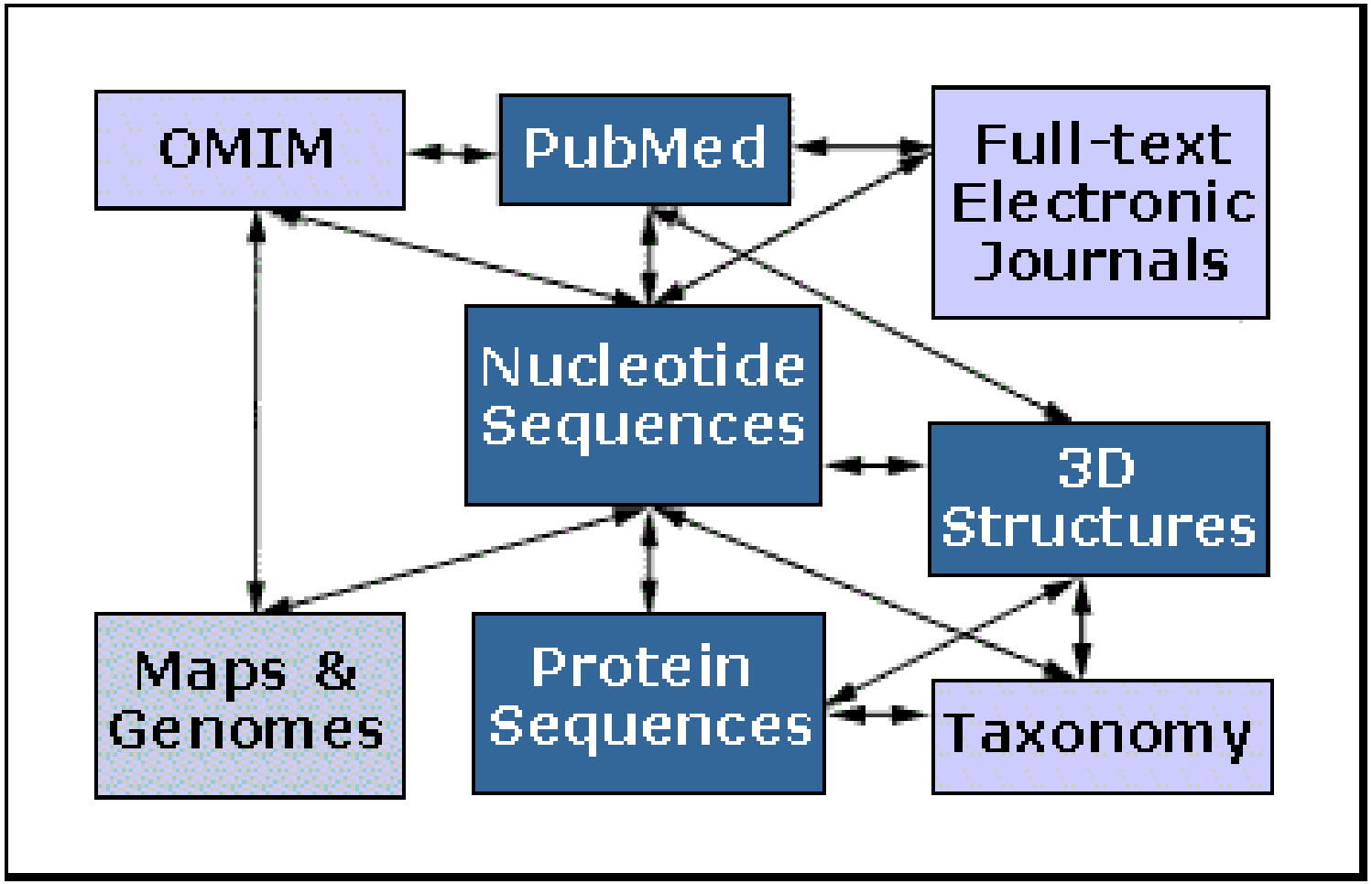
- One of the largest challenges in analyzing genomic data is associating the experimental data with the available metadata, e.g. sequence, gene annotation, chromosomal maps, literature.
- The `annotate` package provides some tools for carrying this out.
- These are very likely to change, evolve and improve, so please check the current documentation - things may already have changed!

WWW resources

- Nucleotide databases: e.g. GenBank.
- Gene databases: e.g. LocusLink, UniGene.
- Protein sequence and structure databases: e.g. SwissProt, Protein DataBank (PDB).
- Literature databases: e.g. PubMed, OMIM.
- Chromosome maps: e.g. NCBI Map Viewer.
- Pathways: e.g. KEGG.
- Entrez is a search and retrieval system that integrates information from databases at NCBI (National Center for Biotechnology Information).

NCBI Entrez

www.ncbi.nlm.nih.gov/Entrez



`annotate`: matching IDs

Important tasks

- Associate manufacturers probe identifiers (e.g. Affymetrix IDs) to other available identifiers (e.g. gene symbol, PubMed PMID, LocusLink LocusID, GenBank accession number).
- Associate probes with biological data such as chromosomal position, pathways.
- Associate probes with published literature data via PubMed.

`annotate`: matching IDs

Affymetrix identifier HGU95A chips	`"41046_s_at"`
LocusLink, LocusID	`"9203"`
GenBank accession #	`"X95808"`
Gene symbol	`"ZNF261"`

PubMed, PMID	`"10486218"`
	`"9205841"`
	`"8817323"`

Chromosomal location	`"X", "Xq13.1"`

`annotate`: database searches and report generation

- Provide tools for searching and processing information from various biological databases.
- Provide tools for regular expression searching of PubMed abstracts.
- Provide nice HTML reports of analyses, with links to biological databases.

Annotation data packages

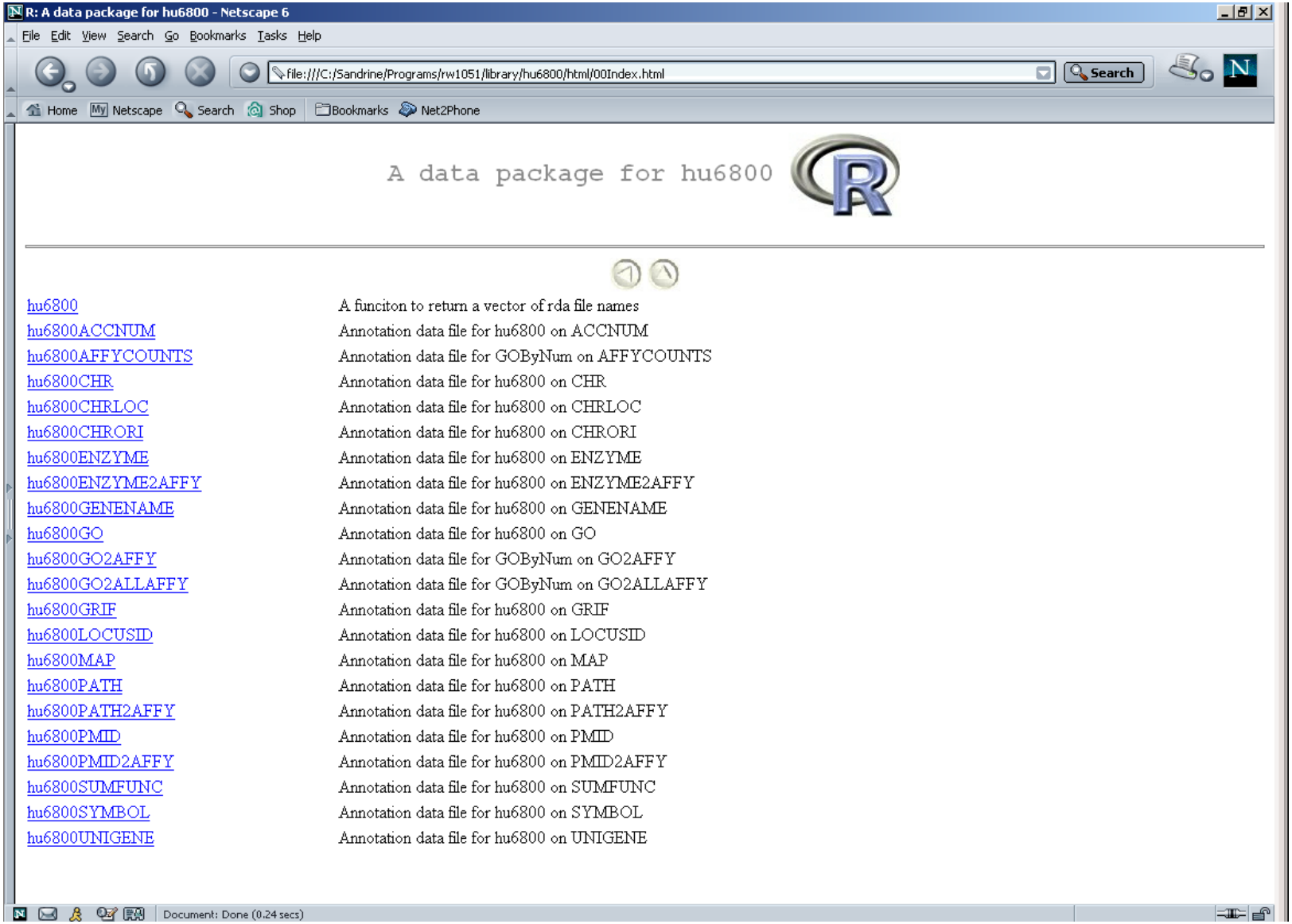
- The Bioconductor project has started to deploy packages that contain only data.

  E.g. **`hgu95a`** package for Affymetrix HGU95A GeneChips series, also, **`hgu133a`**, **`hu6800`**, **`mgu74a`**, **`rgu34a`**.
- These packages contain many different mappings to interesting data.
- They are available from the Bioconductor website and also using **`update.packages`**.

Annotation data packages

- Maps to GenBank accession number, LocusLink LocusID, gene symbol, gene name, UniGene cluster.
- Maps to chromosomal location: chromosome, cytoband, physical distance (bp), orientation.
- Maps to KEGG pathways, enzymes, Gene Ontology Consortium (GO).
- Maps to PubMed PMID.
- These packages will be updated and expanded regularly as new or updated data become available.

hu6800 data package



`annotate`: matching IDs

- Much of what `annotate` does relies on matching symbols.
- This is basically the role of a hash table in most programming languages.
- In R, we rely on environments (they are similar to hash tables).
- The annotation data packages provide R environment objects containing key and value pairs for the mappings between two sets of probe identifiers.
- Keys can be accessed using the R `ls` function.
- Matching values in different environments can be accessed using the `get` or `multiget` functions.

annotate: matching IDs

```
> library(hgu95a)
> get("41046_s_at", env = hgu95aACCNUM)
[1] "X95808"
> get("41046_s_at", env = hgu95aLOCUSID)
[1] "9203"
> get("41046_s_at", env = hgu95aSYMBOL)
[1] "ZNF261"
> get("41046_s_at", env = hgu95aGENENAME)
[1] "zinc finger protein 261"
> get("41046_s_at", env = hgu95aSUMFUNC)
[1] "Contains a putative zinc-binding
  motif (MYM)|Proteome"
> get("41046_s_at", env = hgu95aUNIGENE)
[1] "Hs.9568"
```

annotate: matching IDs

```
> get("41046_s_at", env = hgu95aCHR)
[1] "X"
> get("41046_s_at", env = hgu95aCHRLOC)
[1] "66457019@X"
> get("41046_s_at", env = hgu95aCHRORI)
[1] "-@X"
> get("41046_s_at", env = hgu95aMAP)
[1] "Xq13.1”
> get("41046_s_at", env = hgu95aPMID)
[1] "10486218" "9205841"  "8817323"
> get("41046_s_at", env = hgu95aGO)
[1] "GO:0003677" "GO:0007275"
```

`annotate`: `chromLoc` class

Location information for one gene

- **`chrom`**: chromosome name.
- **`position`**: starting position of the gene in bp.
- **`strand`**: chromosome strand +/-.

annotate: chromLocation class

Location information for a set of genes

- **species:** species that the genes correspond to.
- **datSource:** source of the gene location data.
- **nChrom:** number of chromosomes for the species.
- **chromNames:** chromosome names.
- **chromLocs:** starting position of the genes in bp.
- **chromLengths:** length of each chromosome in bp.
- **geneToChrom:** hash table translating gene IDs to location.

Function **buildChromClass**

`annotate`: WWW queries

- Functions for querying WWW databases from R rely on the **`openBrowser`** function

```
openBrowser("www.r-roject.org")
```

annotate: GenBank query

www.ncbi.nlm.nih.gov/Genbank/index.html

- Given a vector of GenBank accession numbers or NCBI UIDs, the **genbank** function
  - opens a browser at the URLs for the corresponding GenBank queries;
  - returns an **XMLdoc** object with the same data.

```
genbank("X95808",disp="browser")
```

http://www.ncbi.nih.gov/entrez/query.fcgi?tool=bioconductor&cmd=Search&db=Nucleotide&term=X95808

```
genbank(1430782,disp="data",
    type="uid")
```

`annotate`: LocusLink query

www.ncbi.nlm.nih.gov/LocusLink/

- **`locuslinkByID`**: given one or more LocusIDs, the browser is opened at the URL corresponding to the first gene.

  ```
  locuslinkByID("9203")
  ```
  http://www.ncbi.nih.gov/LocusLink/LocRpt.cgi?l=9203

- **`locuslinkQuery`**: given a search string, the results of the LocusLink query are displayed in the browser.

  ```
  locuslinkQuery("zinc finger")
  ```
  http://www.ncbi.nih.gov/LocusLink/list.cgi?Q=zinc finger&ORG=Hs&V=0

`annotate`: PubMed query

www.ncbi.nlm.nih.gov

- For any gene there is often a large amount of data available from PubMed.
- The `annotate` package provides the following tools for interacting with PubMed
  - `pubMedAbst`: a class structure for PubMed abstracts in R.
  - `pubmed`: the basic engine for talking to PubMed.
- WARNING: be careful you can query them too much and be banned!

annotate: pubMedAbst class

Class structure for storing and processing PubMed abstracts in R

- **authors**
- **abstText**
- **articleTitle**
- **journal**
- **pubDate**
- **abstUrl**

`annotate`: high level tools for PubMed query

- **`pm.getabst`**: download the specified PubMed abstracts (stored in XML) and create a list of **`pubMedAbst`** objects.
- **`pm.titles`**: extract the titles from a set of PubMed abstracts.
- **`pm.abstGrep`**: regular expression matching on the abstracts.

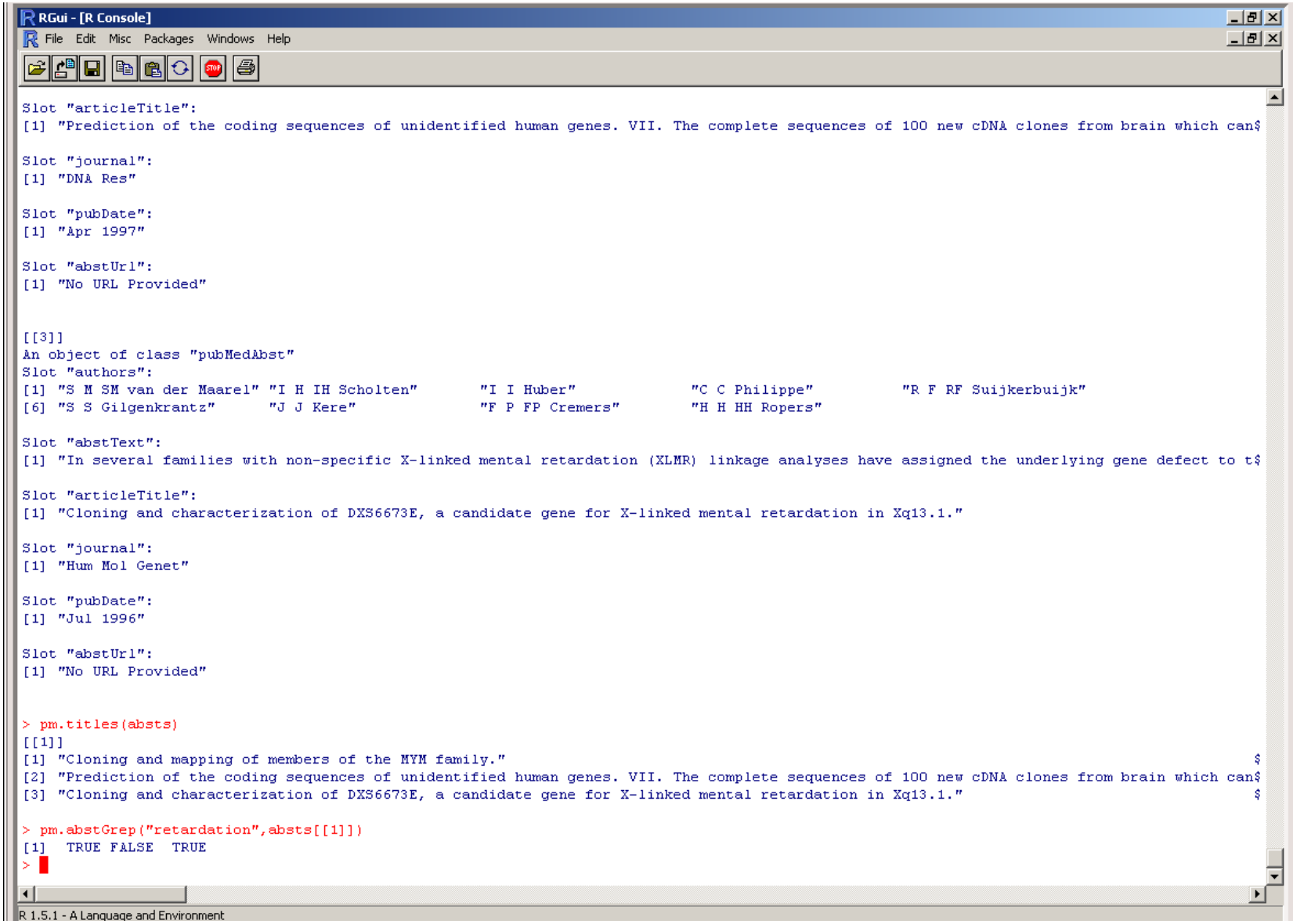
annotate: PubMed example

```
pmid <-get("41046_s_at", env=hgu95aPMID)
pubmed(pmid, disp="browser")
```

http://www.ncbi.nih.gov/entrez/query.fcgi?tool=bioconductor&cmd=Retrieve&db=PubMed&list_uids=10486218%2c9205841%2c8817323

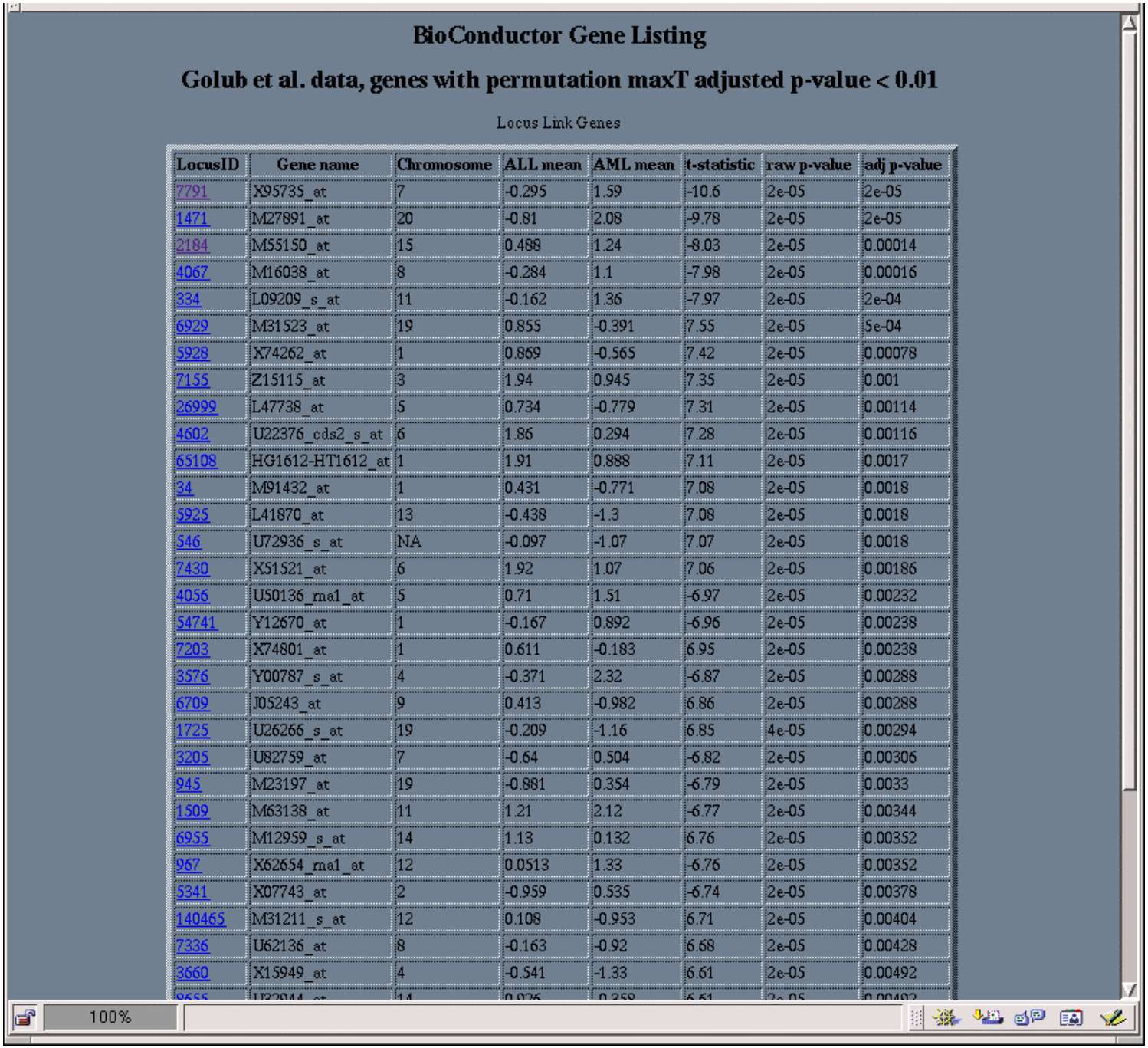
```
absts <- pm.getabst("41046_s_at",
  base="hgu95a")
pm.titles(absts)
pm.abstGrep("retardation",absts[[1]])
```

annotate: PubMed example



`annotate`: data rendering

- A simple interface, `ll.htmlpage`, can be used to generate an HTML report of your results.
- The page consists of a table with one row per gene, with links to LocusLink.
- Entries can include various gene identifiers and statistics.



BioConductor Gene Listing

Golub et al. data, genes with permutation maxT adjusted p-value < 0.01

Locus Link Genes

LocusID	Gene name	Chromosome	ALL mean	AML mean	t-statistic	raw p-value	adj p-value
7791	X95735_at	7	-0.295	1.59	-10.6	2e-05	2e-05
1471	M27891_at	20	-0.81	2.08	-9.78	2e-05	2e-05
2184	M55150_at	15	0.488	1.24	-8.03	2e-05	0.00014
4067	M16038_at	8	-0.284	1.1	-7.98	2e-05	0.00016
334	L09209_s_at	11	-0.162	1.36	-7.97	2e-05	2e-04
6929	M31523_at	19	0.855	-0.391	7.55	2e-05	5e-04
5928	X74262_at	1	0.869	-0.565	7.42	2e-05	0.00078
7155	Z15115_at	3	1.94	0.945	7.35	2e-05	0.001
26999	L47738_at	5	0.734	-0.779	7.31	2e-05	0.00114
4602	U22376_cds2_s_at	6	1.86	0.294	7.28	2e-05	0.00116
65108	HG1612-HT1612_at	1	1.91	0.888	7.11	2e-05	0.0017
34	M91432_at	1	0.431	-0.771	7.08	2e-05	0.0018
5925	L41870_at	13	-0.438	-1.3	7.08	2e-05	0.0018
546	U72936_s_at	NA	-0.097	-1.07	7.07	2e-05	0.0018
7430	X51521_at	6	1.92	1.07	7.06	2e-05	0.00186
4056	U50136_rna1_at	5	0.71	1.51	-6.97	2e-05	0.00232
54741	Y12670_at	1	-0.167	0.892	-6.96	2e-05	0.00238
7203	X74801_at	1	0.611	-0.183	6.95	2e-05	0.00238
3576	Y00787_s_at	4	-0.371	2.32	-6.87	2e-05	0.00288
6709	J05243_at	9	0.413	-0.982	6.86	2e-05	0.00288
1725	U26266_s_at	19	-0.209	-1.16	6.85	4e-05	0.00294
3205	U82759_at	7	-0.64	0.504	-6.82	2e-05	0.00306
945	M23197_at	19	-0.881	0.354	-6.79	2e-05	0.0033
1509	M63138_at	11	1.21	2.12	-6.77	2e-05	0.00344
6955	M12959_s_at	14	1.13	0.132	6.76	2e-05	0.00352
967	X62654_rna1_at	12	0.0513	1.33	-6.76	2e-05	0.00352
5341	X07743_at	2	-0.959	0.535	-6.74	2e-05	0.00378
140465	M31211_s_at	12	0.108	-0.953	6.71	2e-05	0.00404
7336	U62136_at	8	-0.163	-0.92	6.68	2e-05	0.00428
3660	X15949_at	4	-0.541	-1.33	6.61	2e-05	0.00492
9655	U22944_at	14	0.926	0.268	6.61	2e-05	0.00492

100%

`ll.htmlpage` function from **`annotate`** package

genelist.html