Microarray annotation and biological information

Benedikt Brors

Dept. Intelligent Bioinformatics Systems

German Cancer Research Center

b.brors@dkfz.de

dkfz

Why do we need microarray clone annotation?

- Often, the result of microarray data analysis is a list of genes.

- The list has to be summarized with respect to its biological meaning. For this, information about the genes and the related proteins has to be gathered.

- If the list is small (let's say, 1–30), this is easily done by reading database information and/or the available literature.

- Sometimes, lists are longer (100s or even 1000s of genes). Automatic parsing and extracting of information is needed.

- To get complete information, you will need the help of an experienced computational biologist (aka 'bioinformatician'). However, there is a lot that you can do on your own.

Primary databases

- Some information about genes and the encoded proteins is available already from sequence databases, e.g. database accession number, nucleotide and protein sequences, database cross references, and a sequence name that may or may not give a hint to the function. To find a sequence in another database, use sequence comparison tools like BLAST.

- There are large repositories for sequence data, the most prominent being EMBL, GenBank and DDBJ (these 3 are redundant). Because they are so large, nobody cares about the quality of the data. Everybody having internet access can deposit sequence information there. Errors introduced long time ago will stay there forever.

dkfz

GenBank information from NCBI

Curated databases

- In contrast, some databases are *curated*. That means that biologists will get the information first and compare them with literature before it goes into the database. Thus, the database is of high quality, but it takes some time until a newly discovered sequence is entered. Because information is only entered by curators, *annotation* can be unified. Rules can be put in place that say, e.g., that all enzymes cutting off phosphates are called *phosphatases*, not 'phosphate hydrolases'. A very famous curated database is Amos Bairoch's SWISSPROT (http://www.expasy.ch).

SwissProt entry

URL: http://us.expasy.org/cgi-bin/niceprot.pl?P07092

[1] SEQUENCE FROM NUCLEIC ACID.
MEDLINE=88107544; PubMed=3427015; [NCBI, ExPASy, EBI, Israel, Japan]
Sommer J., Gloor S.M., Rovelli G.F., Hofsteenge J., Nick H., Meier R., Monard D.;
"cDNA sequence coding for a rat glia-derived nexin and its homology to members of the serpin superfamily.";
Biochemistry 26:6407-6410(1987).

Comments

- **FUNCTION**: THIS GLYCOPROTEIN PROMOTES NEURITE EXTENSION AND IS A SERINE PROTEASE INHIBITOR WITH ACTIVITY TOWARD THROMBIN, TRYPSIN, AND UROKINASE. BINDS HEPARIN.
- **SUBCELLULAR LOCATION**: Extracellular.
- **SIMILARITY**: BELONGS TO THE SERPIN FAMILY.

Copyright

This SWISS-PROT entry is copyright. It is produced through a collaboration between the Swiss Institute of Bioinformatics and the EMBL outstation – the European Bioinformatics Institute. There are no restrictions on its use by non-profit institutions as long as its content is in no way modified and this statement is not removed. Usage by and for commercial entities requires a license agreement (See http://www.isb-sib.ch/announce/ or send an email to license@isb-sib.ch).

## Cross-references

EMBL	M17784; AAA41209.1; -.[EMBL / GenBank / DDBJ] [CoDingSequence]
PIR	B27496; B27496.
HSSP	P05121; 1A7C. [HSSP ENTRY / PDB]
InterPro	IPR000215; Serpin. Graphical view of domain structure.
Pfam	PF00079; serpin; 1.
SMART	SM00093; SERPIN; 1.
PROSITE	PS00284; SERPIN; 1.
ProDom	[Domain structure / List of seq. sharing at least 1 domain].
BLOCKS	P07092.
ProtoNet	P07092.
ProtoMap	P07092.
PRESAGE	P07092.
DIP	P07092.
ModBase	P07092.
SWISS-2DPAGE	GET REGION ON 2D PAGE.

Keywords

Serine protease inhibitor; Serpin; Heparin-binding; Neurone; Glycoprotein; Signal.

Features

Key	From	To	Length	Description
SIGNAL	1	19	19	POTENTIAL.
CHAIN	20	397	378	GLIA DERIVED NEXIN.
CARBOHYD	159	159		N-LINKED (GLCNAC...) (POTENTIAL).
ACT_SITE	364	365		REACTIVE BOND (POTENTIAL).

Feature table viewer

Sequence information

Other databases

- There are databases that connect sequence information with other data like literature references, three-dimensional (protein) structure, genomic localisation, or disease relatedness. Usually, they are indexed with primary database accession numbers. Often, they also have an interface to search with the sequence itself (mostly by BLAST).

- Some examples:

  OMIM (Online Mendelian inheritance in man): Lists genes that are important in human disease (http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=OMIM).

- Examples (continued):

  Locus Link Links to a supposedly unique locus on the genome, some cross links; only available for some organisms. (http://www.ncbi.nlm.nih.gov/LocusLink/)

  PFAM Gives information about domain structure and relations to other proteins containing these domains (http://www.sanger.ac.uk/Software/Pfam/).

  Gene Cards Gives concise information for human genes, including links to other (non-primary) databases (http://bioinformatics.weizmann.ac.il/cards/, mirror in Heidelberg http://www.dkfz-heidelberg.de/GeneCards/).

Sample entry in OMIM

Sample entry in Locus Link

The relation of clone information to genes and proteins

- Microarrays are produced using information on *expressed sequences* as EST clones, cDNAs, partial cDNAs etc.∎

- At the other end, functional information is generated (and available) for *proteins*. Hence, there is a need to map a clone sequence ID to a protein ID. This is non-trivial.∎

- First, there are usually hundreds of ESTs (and several cDNA sequences) that map to the same gene. The Database *Unigene* tries to resolve this clustering by sequence clustering. However, this is still imperfect, thus certain transcripts map to more than one cluster, and certain genes are split across several clusters.∎

- For some reason, Bioconductor does not use Unigene, but Locus Link to link features on an array to a gene or protein.

dkfz

ESTs in a Unigene cluster
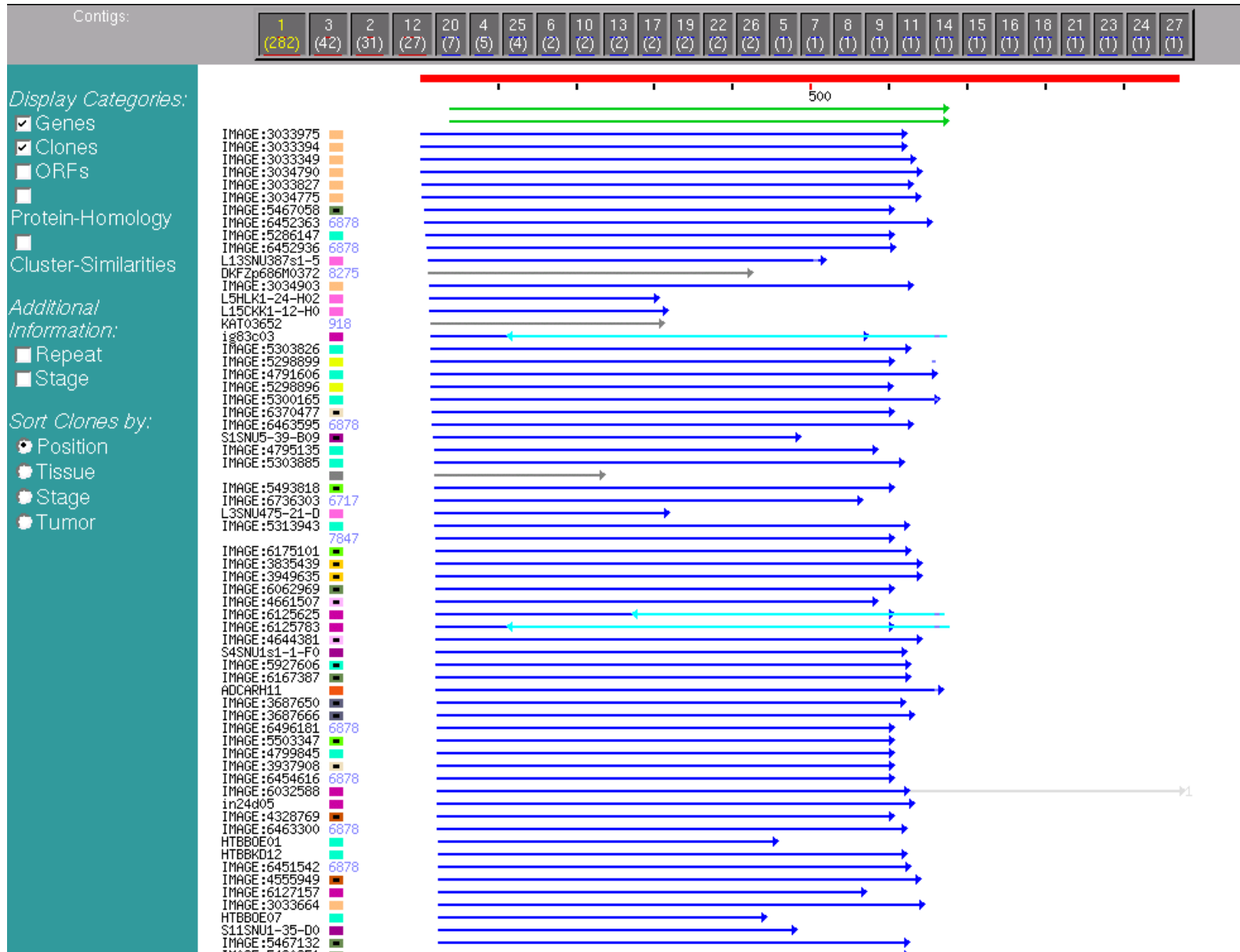
The Human Genome Sequence

- With the completion of the human genome sequence, you'd think that such ambiguities can be resolved. In fact, that is not the case. ▪

- Part of the problem is due to the fact that it is hard to predict gene structure (intron/exon) without knowing the entire mRNA sequence, which happens for about two-thirds of all genes. ▪

- Then, there are errors in the assembly (putting together the sequence snippets). A typical symptom is that a gene appears to map to multiple loci on the same chromosome, with very high sequence similarity. Then, the same sequence was probably introduced several times in the assembly. ▪

- We will later discuss the consequences for microarray annotation.

dkfz

Genomic mapping: ENSEMBL Browser
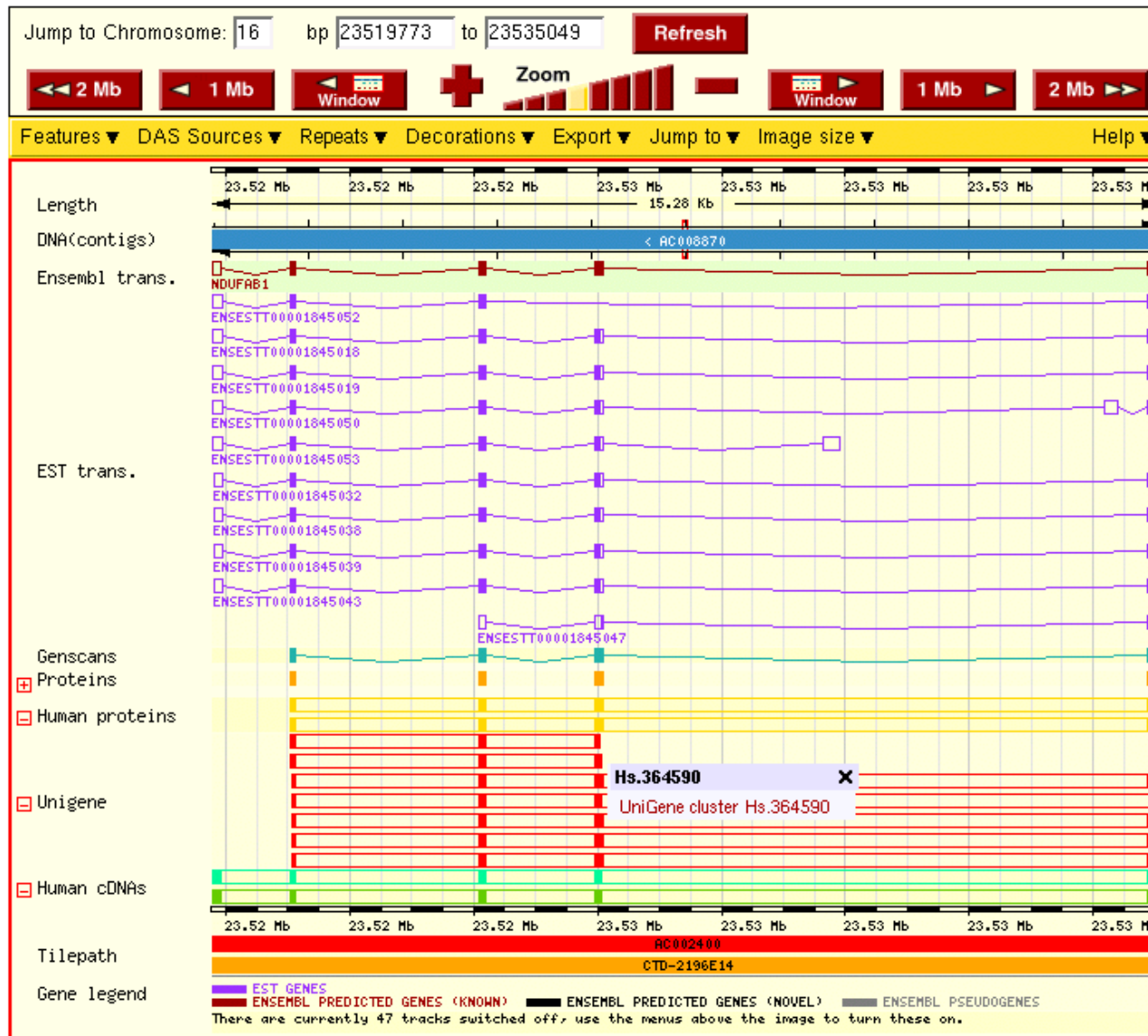
Where do we get all this information?

- Of course, all this can be looked up in a database for the "interesting" genes. For more than some dozens of genes, this is tedious.

- If you're experienced in writing scripts and dealing with biological and relational databases, you can collect the information on your own and build your annotation database. This requires some experience with bioinformatics. You can even prepare your data to access them inside **R**, using the package `AnnBuilder`. It's beyond the scope of this course to explain how this works.

- Fortunately, some data have been precompiled by kind people, and they are distributed via **Bioconductor**.

Data packages in Bioconductor

BioConductor: *open source software for bioinformatics*

Name	Species	Annotation Packages	CDF Packages	Probe Packages
ag	Unknown			Source, Win32
atgenome	Arabidopsis			Source, Win32
ath1121501	Unknown		Source, Win32	Source, Win32
clegans	C. elegans		Source, Win32	Source, Win32
cyp450	CYP 450		Source, Win32	
drosgenome1	Drosphila		Source, Win32	Source, Win32
ecoliantisense	E. coli			Source, Win32
ecoli	E. coli		Source, Win32	Source, Win32
ecolias	E. coli		Source, Win32	
genflex	GenFlex		Source, Win32	
gp53	Unknown		Source, Win32	
hcg110	Human		Source, Win32	Source, Win32
hgfocus	Human		Source, Win32	Source, Win32
hgu133a	Human	Source, Win32	Source, Win32	Source, Win32
hgu133atag	Human		Source, Win32	Source, Win32
hgu133b	Human	Source, Win32	Source, Win32	Source, Win32
hgu95a	Human		Source, Win32	Source, Win32
hgu95av2	Human	Source, Win32	Source, Win32	Source, Win32
hgu95b	Human	Source, Win32	Source, Win32	Source, Win32
hgu95c	Human	Source, Win32	Source, Win32	Source, Win32
hgu95d	Human	Source, Win32	Source, Win32	Source, Win32
hgu95e	Human	Source, Win32	Source, Win32	Source, Win32
hivprtplus2	HIV		Source, Win32	
hu35ksuba	Human		Source, Win32	
hu35ksubb	Human		Source, Win32	
hu35ksubc	Human		Source, Win32	
hu35ksubd	Human		Source, Win32	
hu6800	Human	Source, Win32	Source, Win32	Source, Win32

Bioconductor metadata packages

- These packages contain one-to-one and one-to-many mappings for frequently used chips, especially Affymetrix arrays.

- Information available includes gene names, gene symbol, database accession numbers, Gene Ontology function description, enzmye classification number (EC), relations to PubMed abstracts, and others.

- The data use the framework of the `annotate` package, so I will briefly explain how it works.

Environments in R

- To quickly find information on one subject in a long list, a data structure called *hash table* is frequently used in computer science.

- A hash table is a list of key/value pairs, where the key is used to find the corresponding value. To go the other way round, you have to use pattern matching, which is much slower.

- In R, hash tables are implemented as *environments*. For the moment, we do not care about the philosophy behind it and simply treat it as another word for hash table.

Setting up environments

To set up a new environment:

```
symbol.hash = new.env(hash=TRUE)
```

To create a key/value pair:

```
assign("1234_at", "EphA3", env=symbol.hash)
```

To list all keys of an environment:

```
ls(env=symbol.hash)
```

To get the value for a certain key:

```
get("1234_at", env=symbol.hash)
```

The annotate package

- That's all standard R. The annotate package gives one further function, `multiget`, which retrieves more than one entry at a time, and definitions for special data, e.g. PubMed abstracts, or chromosomal location objects.

- ChromLoc objects are quite useful if you want to associate gene expression with certain positions on a chromosome, e.g. if aberration occurs in your samples.

- You can construct a ChromLoc object on your own ($\rightarrow$ Vignette), or use the function `buildChromLocation`. For chip HGU95a_v2:

```
library(hgu95av2)
cl.95a = buildChromLocation("hgu95av2")
```

Plots for ChromLocation objects

- Plotting methods are available via library `geneplotter`
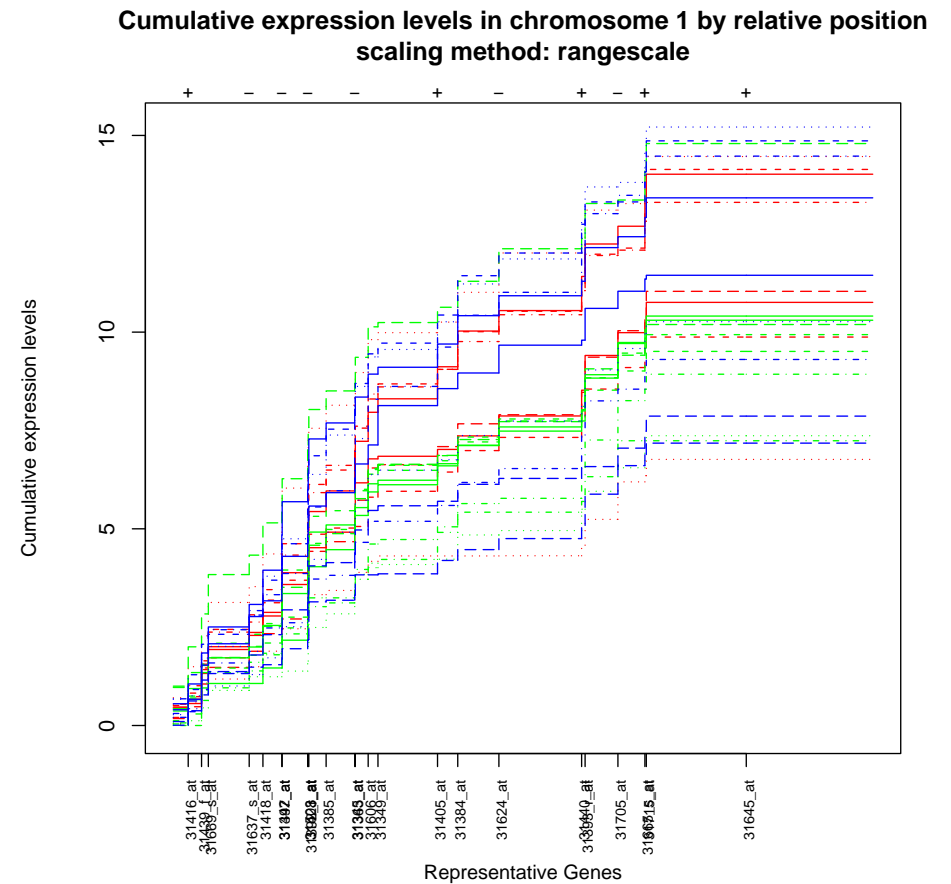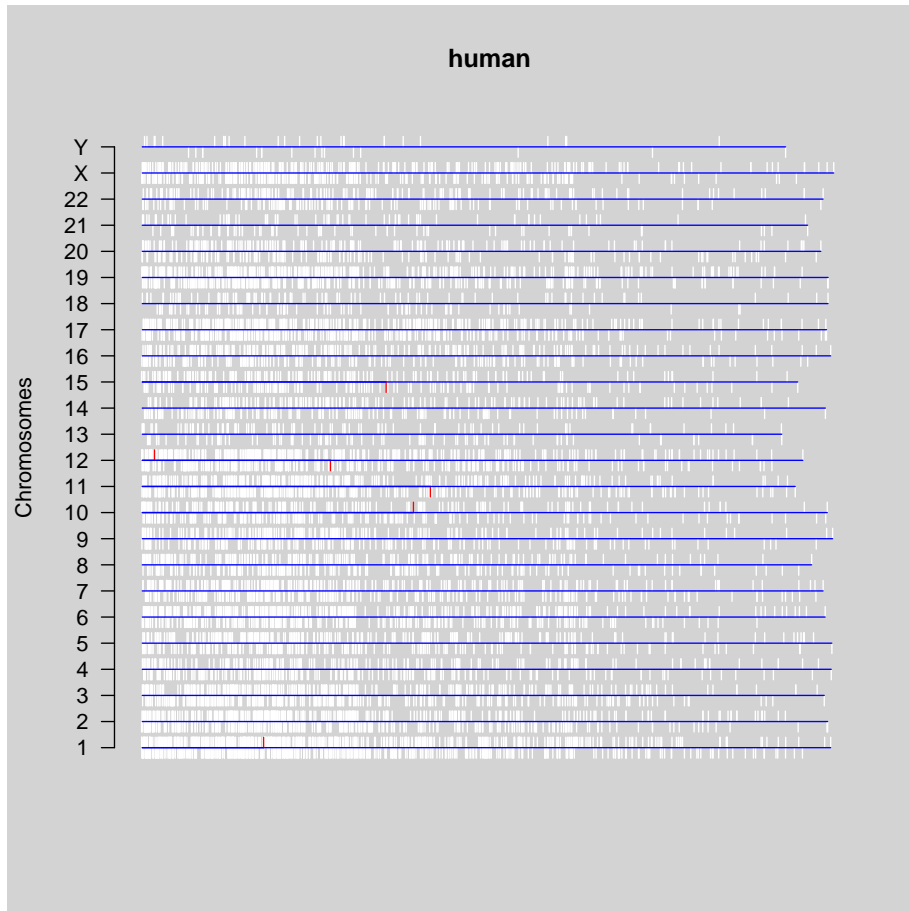
How to get annotation for a set of genes

- Suppose you have found some interesting genes. The index in the matrix is in `index.int`. To get the gene names:

  ```
  gnam.int = geneNames(exprset)[index.int]
  ```

- To find the description:

  ```
  multiget(gnam.int, env=hgu95av2GENENAME)
  ```

- To get EC Numbers (relating to KEGG pathways):

  ```
  multiget(gnam.int, env=hgu95av2ENZYME)
  ```

dkfz

Some caveats

- Because of the non-unique matching of sequences to the genome, array features are sometimes annotated with more than one position:

```
a = ls(env=hgu95av2CHRLOC)
table(sapply(multiget(a, env=hgu95av2CHRLOC),
    length))
```

1	2	3	4	5	6	7	8
11793	647	127	29	13	10	2	1

- For the 800 or so sequences with more than one location, only the first one is used, although there is no warning. It should be desirable to resolve the ambiguities by hand, but nobody has done yet.

- There are even 14 probe sets on HGU95A_v2 that map to 2 chromosomes; however, these are located on some special extrachromosomal segment and annotated with "X" and "Y".

- N.B. There is a special annotation package for Affymetrix arrays, `annaffy`. It does not provide much other functionality than `annotate`, but allows to do the same things differently (and maybe more conveniently).

Pattern matching

- To find something in character vectors or character lists, some pattern matching is required.

- If you have real full names, use `match`, e.g.

  ```
  match("1234_at", rownames(exprs(exprset)) )
  ```

- This will give you the index of ''`1234_at`''. It works also with more than one gene:

  ```
  match(gnam.int, rownames(exprs(exprset)) )
  ```

  will give all indeces for genes in `gnam.int`.

- If you want to use regular expression matching, use `grep`.

dkfz

Export of annotation to HTML

- `annotate` is able to export tables of gene annotations to HTML, which is much nicer to browse than text tables

- Suppose, from a t-test you have for some genes `igenes`: mean of genes in class 1, `igenes.gp1`, mean in class 2, `igenes.gp2`, and P-value `igenes.pval`. To construct pretty HTML output:

```
igenes.ll = multiget(igenes, env=hgu95av2LOCUSID)
igenes.sym = multiget(igenes, env=hgu95av2SYMBOL)
ll.htmlpage(igenes.ll, "HOWTO.igenes", "Some genes",
   list(igenes,sym, igenes, round(igenes.gp1,3),
   round(igenes.gp2,3),round(igenes.pval,3)))
```

dkfz

The result

BioConductor Linkage List

Some genes

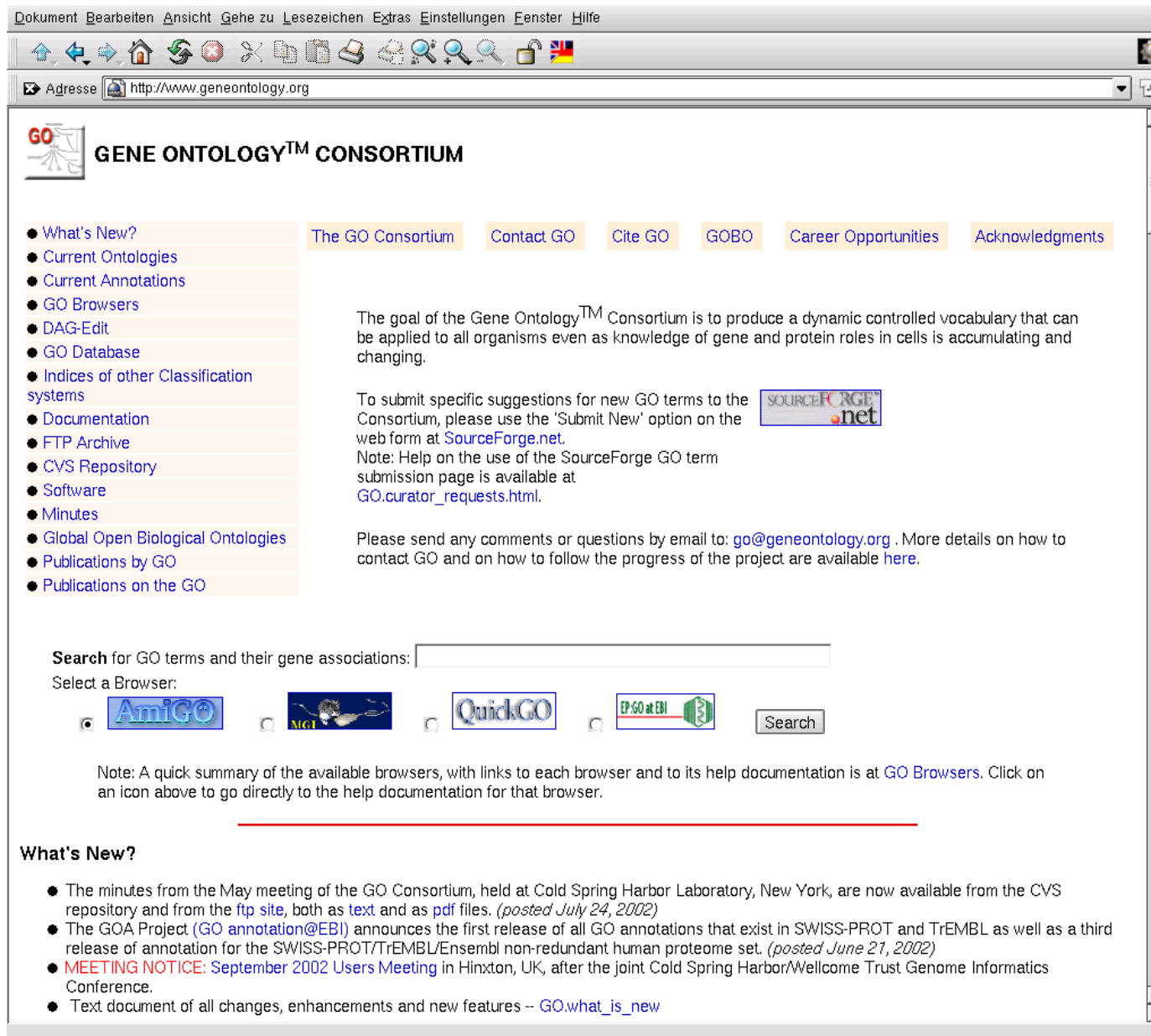|					
23378	KIAA0409	31484_at	145.869	153.948	0.635
221823	LOC221823	31485_at	150.41	153.703	0.892
4330	MN1	31486_s_at	13.057	16.238	0.447
9637	FEZ2	31487_at	82.982	27.448	0.311
27335	eIF3k	31488_s_at	268.605	259.847	0.864
NA	NA	31489_at	0.886	0.479	0.873
6331	SCN5A	31490_at	200.904	194.797	0.767
841	CASP8	31491_s_at	22.029	23.582	0.606
27335	eIF3k	31492_at	293.814	318.384	0.736
1442	CSH1	31493_s_at	29.719	32.583	0.82
NA	NA	31494_at	6.14	5.071	0.773
6846	XCL2	31495_at	118.936	113.031	0.714
6846	XCL2	31496_g_at	49.544	42.06	0.455
2543	GAGE1	31497_at	309.21	363.383	0.354
2578	GAGE6	31498_f_at	104.038	161.529	0.44
2215	FCGR3B	31499_s_at	163.479	132.496	0.448

dkfz

Function annotation

- Probably, the most important thing you want to know is what the genes or their products are concerned with, i.e. their **function**.

- Function annotation is difficult: Different people use different words for the same function, or may mean different things by the same word. The context in which a gene was found (e.g. "TGFβ-induced gene") may not be particularly associated with its function.

- Inference of function from sequence alone is error-prone and sometimes unreliable. The best function annotation systems (GO, SwissProt) use human beings who read the literature before assigning a function to a gene.

The Gene Ontology system

- To overcome some of the problems, an annotation system has been created: Gene Ontology (http://www.geneontology.org). Ontology means here the art (or science) of giving everything its correct name.

- It represents a unified, consistent system, i.e. terms occur only once, and there is a dictionary of allowed words.

- Furthermore, terms are related to each other: the hierarchy goes from very general terms to very detailed ones.

The Gene Ontology site

The Gene Ontology hierarchy

Actual annotation

- Gene Ontology by itself is only a system for annotating genes and proteins. It does not relate database entries to a special annotation value.

- Luckily, research communities for several model organisms have agreed on entering Gene Ontology information into the databases. As this is done 'by hand', GO annotation for most organisms is far from complete.

Available Gene Ontology information

	Biological Process		Molecular Function		Cellular Component		Total Gene Products Associated	Total References Included as Evidence	TAB Delimited File(s) of Gene Associations
	All codes	no IEA code	All codes	no IEA code	All codes	no IEA code			
SGD *Saccharomyces*	6382	3527	6392	3369	3661	3661	6899	2643	download View
FlyBase *Drosophila*	3362	3354	6374	6365	3425	3398	7299	5179	download View
MGI *Mus*	6367	2139	7594	2271	5948	2115	8666	2170	download View
TAIR *Arabidopsis*	5532	151	7597	2081	2490	290	9654	386	download View
PomBase *Schizosaccharomyces*	3466	3466	0	0	1939	1939	3650	3524	download View
WormBase *Caenorhabditis*	4920	1311	5559	18	2822	387	6747	27	download View
RGD *Rattus*	913	0	1179	0	753	0	1303	1	download View
Gramene: *Oryza* (Rice)	2267	55	3110	46	1029	49	3321	1093	download View
TIGR: *Arabidopsis*	1918	1918	4696	4696	1080	1080	4985	472	download View
TIGR: Gene Index README	78488	0	79569	0	69890	0	97809	1	download
TIGR: *Vibrio cholerae*	2923	2923	2721	2721	189	189	2924	10	download View
Compugen README	631750	0	631105	0	640209	0	658168	1	download View
GO Annotations @ EBI: Human README	15754	7784	18055	7349	13190	6511	19912	9618	download
GO Annotations @ EBI: SwissPROT/TrEMBL README	360534	10014	442771	15103	285587	7801	507964	13160	download
Sanger: *G. morsitans* (Tsetse fly) README	1284	0	2397	0	1251	0	2653	1	download

numbers as of September 22, 2002

In the table above gene association counts are provided for all evidence codes and separately for everything except IEA. The IEA code, inferred from electronic annotation, is the lowest quality code. IEA is the only code currently in use that does not require human judgement during the curation process. Also see the GO evidence code documentation.

dkfz

Some software packages for function analysis

- There are some packages (outside R) that allow to map gene expression profiles to biological information, like pathways.

- Unfortunately, pathway information is not well structured, and little systematic information on pathways is available.

- Some systems collect pathway maps from users. However, most pathway information is still on metabolic pathways (see KEGG for this).

- For other pathway information, see `www.biocarta.com` and `www.stke.org` (requires registration). For software, see e.g. GeneMAPP (`www.genemapp.org`) or GoMiner (`http://discover.nci.nih.gov/gominer`).

dkfz

Dealing with GO annotations

- Since the annotation system is hierarchical, i.e. for each term there is a hierarchical list of more general terms, we can compare functions of genes on every level we wish.

- Technically, this amounts to the problem of finding the least common parent node between to genes of interest.

- This can be used to find clusters of functionally related genes in a list that comes out of some other analysis.

dkfz

Comparing GO-annotated genes
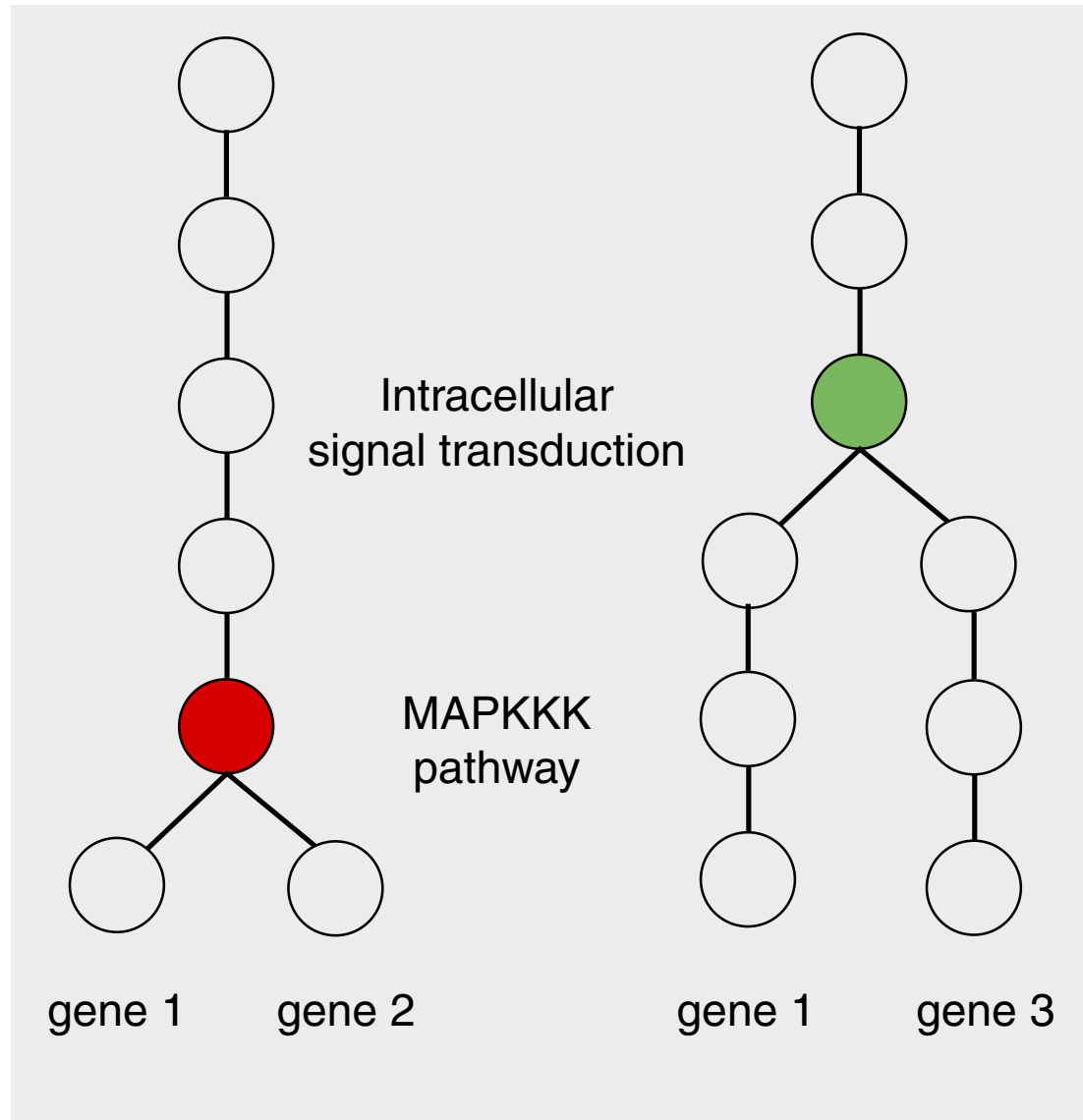
GO functional clusters as a graph
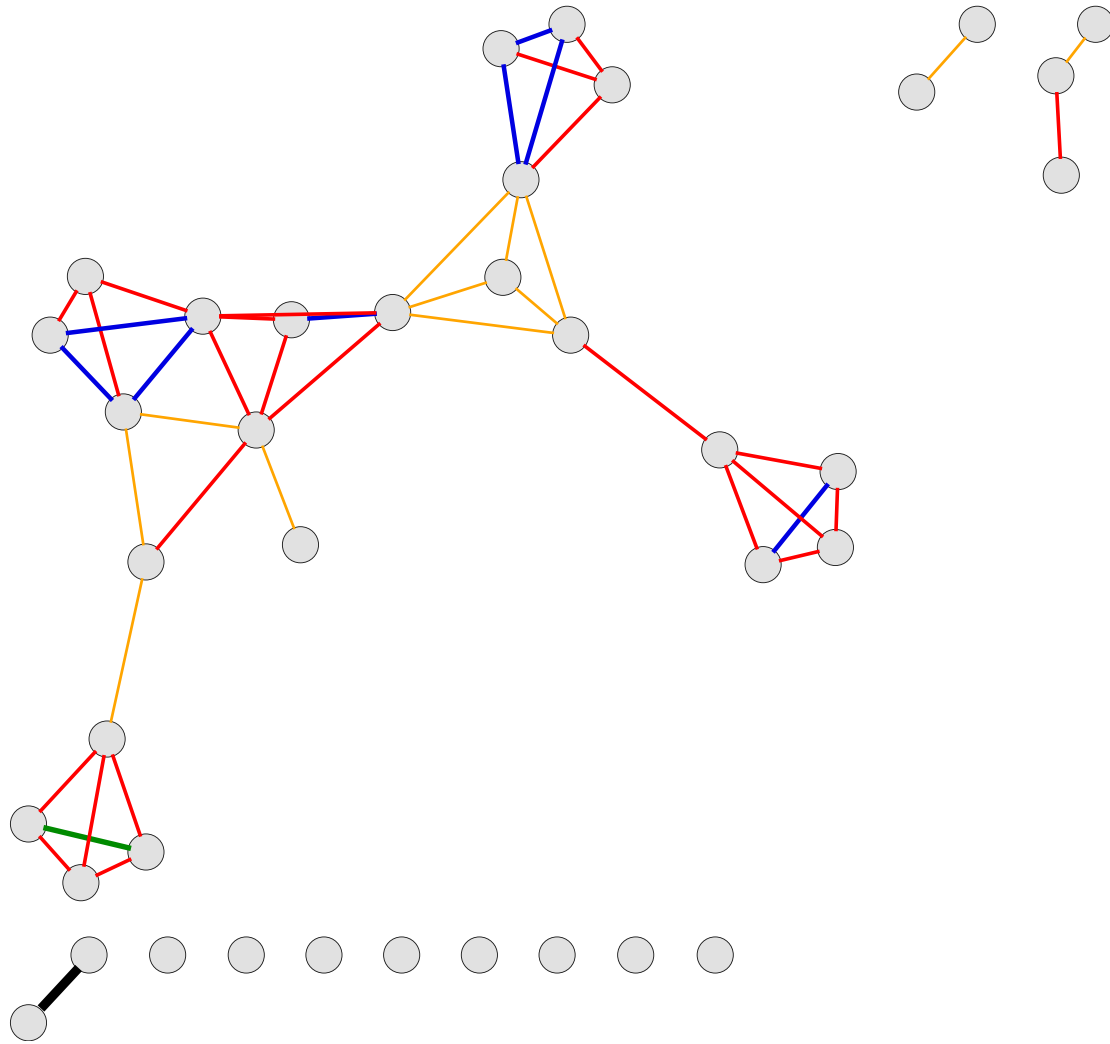
Graphs as analysis tools

- Graphs are quite useful for bioinformatic analysis, and have a long-standing history in sequence analysis.

- Recently, some functionality has been built into R to deal with graphs (`graph`, `Rgraphviz`, `RBGL`). Certainly, the most useful capability is to visualize graphs via `Rgraphviz`. The R package is an interface to the external program `graphviz` (from AT&T). Big graphs should be visualized by means of `ggobi`, however.

- Some other immediate use is to construct PubMed co-citation graphs for genes of interest. Functions for this exist. However, for many other applications the meaning of graphs or graph-theoretic algorithms is not clear, so a lot of work remains to be done.

dkfz