HowTo: Querying online Data

Jeff Gentry and Robert Gentleman

April 20, 2026

1 Overview

This article demonstrates how you can make use of the tools that have been provided for on-line querying of data resources. These tools rely on others (such as the NLM and NCBI) providing and documenting appropriate web interfaces. The tools described here allow you to either retrieve the data (usually in XML) or have it rendered in a browser on the local machine. To do this you will need the *Biobase*, *XML*, and *annotate* packages. The functionality in this article was first described in (Gentleman and Gentry, 2002), although some enhancements have been made since the writing of that article.

Assembling and using meta-data annotation is a non-trivial task. In the Bioconductor Project we have developed tools to support two different methods of accessing meta-data. One is based on obtaining data from a variety of sources, of curating it and packaging it in a form that is suitable for analysing microarray data. The second method is to make use of on-line resources such as those provided by NLM and NCBI. The functions described in this vignette provide infrastructure for that second type of meta-data usage.

We first describe the functions that allow users to specify queries and open the appropriate web page on their local machine. Then, we investigate the much richer set of tools that are provided by NLM for accessing and working with PubMed data.

2 Using the Browser

There are currently four functions that provide functionality for accessing online data resources. They are:

genbank Users specify GenBank identifiers and can request them to be rendered in the browser or returned in XML.

pubmed Users specify PubMed identifiers and can request them to be rendered in the browser or returned in XML. More details on parsing and manipulating the XML are given below.

entrezGeneByID Users specify Entrez Gene identifiers and the appropriate links are opened in the browser. Entrez Gene does not provide XML so there is no download option, currently. The user can request that the URL be rendered or returned.

entrezGeneQuery Users specify a string that will be used as the Entrez Gene query and the species of interest (there can be several). The user can request either that the URL be rendered or returned.

Both `genbank` and `pubmed` can return XML versions of the data. These returned values can subsequently be processed using functionality provided by the *XML* package (Temple Lang, 2000). Specific details and examples for PubMed are given in Section 3.

The function `entrezGeneByID` takes a set of known Entrez Gene identifiers and constructs a URL that will have these rendered. The user can either save the URL (perhaps to send to someone else or to embed in an HTML page, see the vignette on creating HTML output for more details).

The function `entrezGeneQuery` takes a character string to be used for querying PubMed. For example, this function call,

```
entrezGeneQuery("leukemia", "Homo sapiens")
```

will find all Human genes that have the word leukemia associated with them in their Entrez Gene records. Note that the R code is merely an interface to the services provided by NLM and NCBI and users are referred to those sites for complete descriptions of the algorithms they use for searching etc.

3 Accessing PubMed information

In this section we demonstrate how to query PubMed and how to operate on the data that are returned. As noted above, these queries generate XML, which must then be parsed to provide the specific data items of interest. Our example is based on the `sample.ExpressionSet` data from the package *Biobase*. Users should be able to easily replace these data with their own.

```
> library("annotate")
> data(sample.ExpressionSet)
> affys <- featureNames(sample.ExpressionSet)[490:500]
> affys
```

```
 [1] "31729_at" "31730_at" "31731_at" "31732_at" "31733_at" "31734_at"
 [7] "31735_at" "31736_at" "31737_at" "31738_at" "31739_at"
```

Here we have selected an arbitrary set of 11 genes to be interested in from our sample data. However, `sample.ExpressionSet` provided us with Affymetrix identifiers, and for the `pubmed` function, we need to use PubMed ID values. To obtain these, we can use the annotation tools within *annotate*.

```
> library("hgu95av2.db")
> ids <- getPMID(affys,"hgu95av2")
> ids <- unlist(ids,use.names=FALSE)
> ids <- unique(ids[!is.na(as.numeric(ids))])
> length(ids)
```

```
[1] 960
```

```
> ids[1:10]
```

```
 [1] "1939271" "2449431" "7729427" "7835343" "7836461" "7933101"
 [7] "8121496" "8680883" "8764009" "8764062"
```

We use `getPMID` to obtain the PubMed identifiers that are related to our probes of interest. Then we process these to leave out any that have no PMIDs and we remove duplicates as well. The mapping to PMIDs are actually based on Entrez Gene identifiers and since the mapping from Affymetrix IDs to Entrz Gene is many to one there is some chance of duplication. From our initial 11 Affymetrix identifiers we see that there are 960 unique PubMed identifiers (i.e. papers).

For each of these papers we can obtain information, such as the title, the authors, the abstract, the Entrez Gene identifiers for genes that are referred to in the paper and many other pieces of information. Again, for a complete listing and description the reader is referred to the NLM website.

We next generate the query and store the results in a variable named `x`. This object is of class `XMLDocument` and to manipulate it we will use functions provided by the XML package.

```
> x <- pubmed(ids[1:10])
> a <- xmlRoot(x)
> numAbst <- length(xmlChildren(a))
> numAbst
```

```
[1] 10
```

Our search of the 960 PubMed IDs (from the 11 Affymetrix IDs) has resulted in 10 abstracts from PubMed (stored in R using XML format). The *annotate* package also provides a `pubMedAbst` class, which will take the raw XML format from a call to `pubmed` and extract the interesting sections for easy review.

```
> arts <- vector("list", length=numAbst)
> absts <- rep(NA, numAbst)
> for (i in 1:numAbst) {
+     ## Generate the PubMedAbst object for this abstract
+     arts[[i]] <- buildPubMedAbst(a[[i]])
+     ## Retrieve the abstract text for this abstract
+     absts[i] <- abstText(arts[[i]])
+ }
> arts[[7]]
```

```
An object of class 'pubMedAbst':
Title: Direct interaction of human TFIID with the HIV-1
     transactivator tat.
PMID: 8121496
```

```
Authors: F Kashanchi, G Piras, MF Radonovich, JF Duvall, A
     Fattaey, CM Chiang, RG Roeder, JN Brady
Journal: Nature
Date: Jan 1994
```

In the S language we say that the `pubMedAbst` class has a number of different slots. They are:

authors The vector of authors.

pmid The PubMed record number.

abstText The actual abstract (in text).

articleTitle The title of the article.

journal The journal it is published in.

pubDate The publication date.

These can all be individually extracted utilizing the provided methods, such as `abstText` in the above example. As you can see, the `pubMedAbst` class provides several key pieces of information: authors, abstract text, article title, journal, and the publication date of the journal.

Once the abstracts have been assembled you can search them using any of the standard search techniques. Suppose for example we wanted to know which abstracts have the term `cDNA` in them, then the following code chunk shows how to identify these abstracts.

```
> found <- grep("cDNA",absts)
> goodAbsts <- arts[found]
> length(goodAbsts)
```

```
[1] 2
```

So 2 of the articles relating to our genes of interest mention the term `cDNA` in their abstracts.

Lastly, as a demonstration for how one can use the `query` toolset to cross reference several databases, we can use the same set of PubMed IDs with another function. In this example, the `genbank` function is used with the argument `type="uid"`. By default, the `genbank` function assumes that the id values passed in are Genbank accession numbers, but we currently have PubMed ID values that we want to use. The `type="uid"` argument specifies that we are using PubMed IDs (aka NCBI UID numbers).

```
> y <- genbank(ids[1:10], type="uid")
> b <- xmlRoot(y)
```

At this point the object `b` can be manipulated in a manner similar to `a` from the PubMed example.

Also, note that both `pubmed` and `genbank` have an option to display the data directly in the browser instead of XML, by specifying `disp="browser"` in the parameter listing.

4 Generating HTML output for your abstracts

Many users find it useful to have a web page created with links for all of their abstracts, leading to the actual PubMed page online. These pages can then be distributed to other people who have an interest in the abstracts that you have found. There are two formats for this, the first provides for a simple HTML page which has a link for every abstract, and the other provides for a framed HTML page with the links on the left and the resulting PubMed page in the main frame. For these examples, we will be using temporary files:

```
> fname <- tempfile()
> pmAbst2HTML(goodAbsts, filename=fname)
> fnameBase <- tempfile()
> pmAbst2HTML(goodAbsts, filename=fnameBase, frames=TRUE)
```

5 Session Information

The version number of R and packages loaded for generating the vignette were:

```
R version 4.6.0 RC (2026-04-17 r89917)
Platform: x86_64-pc-linux-gnu
Running under: Ubuntu 24.04.4 LTS

Matrix products: default
BLAS:   /home/biocbuild/bbs-3.24-bioc/R/lib/libRblas.so
LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.12.0  LAPACK version 3.1

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_GB              LC_COLLATE=C
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

time zone: America/New_York
tzcode source: system (glibc)

attached base packages:
[1] grid      stats4    stats     graphics  grDevices utils
[7] datasets  methods   base

other attached packages:
 [1] Rgraphviz_2.55.0     graph_1.89.1         xtable_1.8-8
 [4] GO.db_3.23.1         hgu95av2.db_3.13.0   org.Hs.eg.db_3.23.1
```

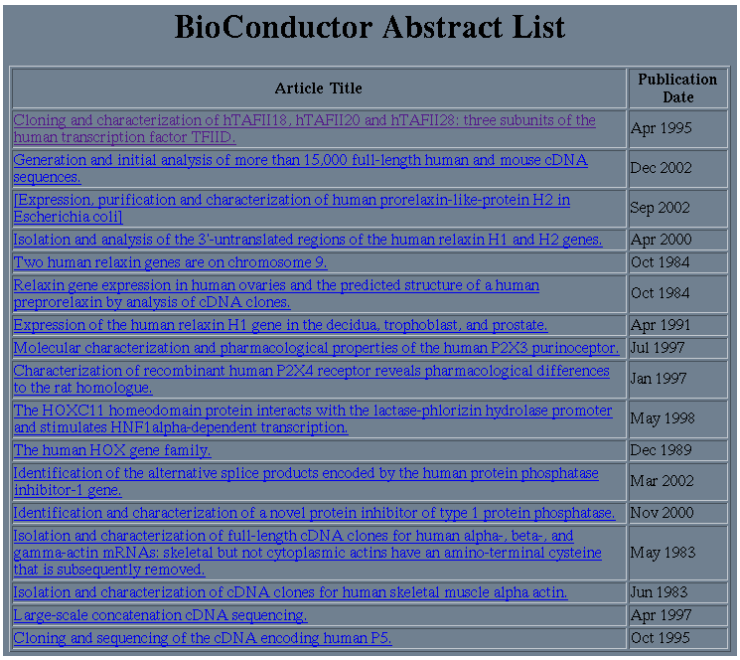
```
 [7] annotate_1.89.0      XML_3.99-0.23        AnnotationDbi_1.73.1
[10] IRanges_2.45.0       S4Vectors_0.49.2     Biobase_2.71.0
[13] BiocGenerics_0.57.1  generics_0.1.4       BiocStyle_2.39.0

loaded via a namespace (and not attached):
 [1] bit_4.6.0           jsonlite_2.0.0      compiler_4.6.0
 [4] BiocManager_1.30.27 crayon_1.5.3        blob_1.3.0
 [7] Biostrings_2.79.5   jquerylib_0.1.4     Seqinfo_1.1.0
[10] png_0.1-9           yaml_2.3.12         fastmap_1.2.0
[13] R6_2.6.1            XVector_0.51.0      knitr_1.51
[16] bookdown_0.46       DBI_1.3.0           bslib_0.10.0
[19] rlang_1.2.0         KEGGREST_1.51.1     cachem_1.1.0
[22] xfun_0.57           sass_0.4.10         bit64_4.6.0-1
[25] otel_0.2.0          RSQLite_2.4.6       memoise_2.0.1
[28] cli_3.6.6           digest_0.6.39       lifecycle_1.0.5
[31] vctrs_0.7.3         evaluate_1.0.5      rmarkdown_2.31
[34] httr_1.4.8          pkgconfig_2.0.3     tools_4.6.0
[37] htmltools_0.5.9
```

References

R. Gentleman and J. Gentry. Querying pubmed. *R News*, 2(2):28–31, June 2002. URL http://CRAN.R-project.org/doc/Rnews/.

Duncan Temple Lang. Tools for parsing and generating xml within r and s-plus. *CRAN*, 2000. URL http://www.omegahat.org/RSXML.



BioConductor Abstract List

Article Title	Publication Date
Cloning and characterization of hTAFII18, hTAFII20 and hTAFII28: three subunits of the human transcription factor TFIID.	Apr 1995
Generation and initial analysis of more than 15,000 full-length human and mouse cDNA sequences.	Dec 2002
[Expression, purification and characterization of human prorelaxin-like-protein H2 in Escherichia coli]	Sep 2002
Isolation and analysis of the 3'-untranslated regions of the human relaxin H1 and H2 genes.	Apr 2000
Two human relaxin genes are on chromosome 9.	Oct 1984
Relaxin gene expression in human ovaries and the predicted structure of a human preprorelaxin by analysis of cDNA clones.	Oct 1984
Expression of the human relaxin H1 gene in the decidua, trophoblast, and prostate.	Apr 1991
Molecular characterization and pharmacological properties of the human P2X3 purinoceptor.	Jul 1997
Characterization of recombinant human P2X4 receptor reveals pharmacological differences to the rat homologue.	Jan 1997
The HOXC11 homeodomain protein interacts with the lactase-phlorizin hydrolase promoter and stimulates HNF1alpha-dependent transcription.	May 1998
The human HOX gene family.	Dec 1989
Identification of the alternative splice products encoded by the human protein phosphatase inhibitor-1 gene.	Mar 2002
Identification and characterization of a novel protein inhibitor of type 1 protein phosphatase.	Nov 2000
Isolation and characterization of full-length cDNA clones for human alpha-, beta-, and gamma-actin mRNAs: skeletal but not cytoplasmic actins have an amino-terminal cysteine that is subsequently removed.	May 1983
Isolation and characterization of cDNA clones for human skeletal muscle alpha actin.	Jun 1983
Large-scale concatenation cDNA sequencing.	Apr 1997
Cloning and sequencing of the cDNA encoding human P5.	Oct 1995

Figure 1: pmAbst2HTML without frames

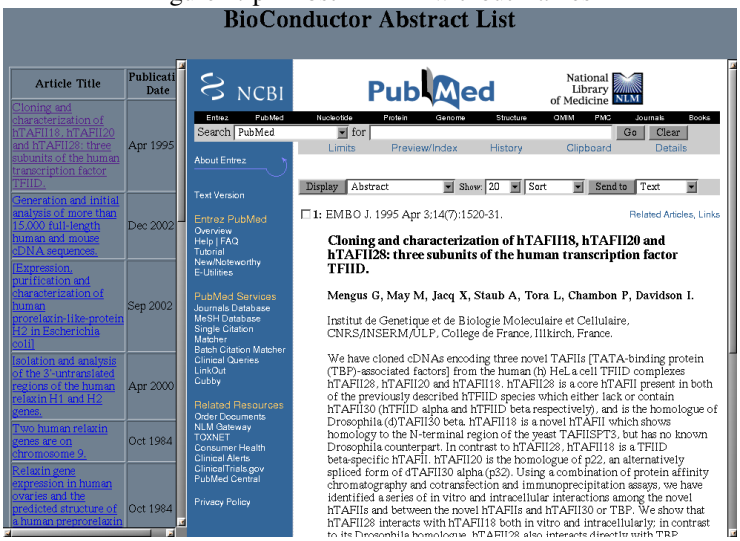


Figure 2: pmAbst2HTML with frames