Using the PAnnBuilder Package

Hong Li[‡] [*]

May 3, 2016

[‡]Key Lab of Systems Biology
Shanghai Institutes for Biological Sciences
Chinese Academy of Sciences, P. R. China

Contents

---

[*] sysptm@gmail.com

1 Overview

In genomic era, genome-scale experiments and data analyses require genes to be annotated from different sources, an R R Development Core Team (2008) package AnnBuilder was developped for this purpose Zhang et al. (2003). In post genomic era, advances in proteomics highlight the urgence of understanding protein language Jensen (2006). However, relative to genes, AnnBuilder is limited in protein annotation due to the complexity of proteins caused by 3-D strucutre, alternative splicing, modification, dynamic location and other features. The package PAnnBuilder focuses on assembling proteomic annotation data, which should be very useful for proteomic data interpretation. It not only inherits the good features of AnnBuilder such as automatically parsing protein annotation data and building R packages from selected sources, but also emphasizes specific features of proteins. PAnnBuilder currently supports 16 databases involving diverse aspects of proteomics, such as protein primary data, protein domain/family, subcellular location, protein interaction, post-translational modifications, body fluid proteomics, homolog/ortholog groups and so on. Additionally, PAnnBuilder allows annotation to unknown proteins based on sequence similarities to other well-annotated proteins. To extend the use of PAnnBuilder, 54 standard R annotation packages are produced from main protein databases, which are freely available for download via biocLite.

2 Getting Started

2.1 Requirements

PAnnBuilder requires the support from the following items:

1. For PAnnBuilder >= 1.3.0, R >=2.7.0 is needed for building SQLite-based package. Dependent R packages are needed to be installed: *methods*, *utils*, *RSQLite*, *Biobase* (>= 1.17.0), *AnnotationDbi (>= 1.3.12)*. If you use the installation script "PAnnBuilder.R" to install PAnnBuilder (>=1.3.0), it will automatically check these dependent packages and install missing packages from CRAN or Bioconductor.

2. Rtools is needed for Windows user. The matched version of Rtools with R can be downloaded via `http://www.murdoch-sutherland.com/Rtools/`.

3. Perl is required to parse the rather large annotation source data files. It can be downloaded from `http://www.perl.com/download.csp`.

4. Program formatdb and BLASTP is needed for function `crossBuilder`, `crossBuilder_DB`, `pSeqBuilder`, `pSeqBuilder_DB`.

   BLAST can be downloaded from http://www.ncbi.nlm.nih.gov/BLAST/download.shtml.

Note: It is better to have enough space for the temporary directory. The path of the per-session temporary directory can be acquired by:

```
> tempdir()
```

```
[1] "/tmp/RtmpzoLKEC"
```

2.2 Installation

The biocLite script is used to install PAnnBuilder from within R:

```
> source("http://bioconductor.org/biocLite.R")
> biocLite("PAnnBuilder")
```

```
> library(PAnnBuilder)
```

Note: Web Connection is needed to install PAnnBuilder and its depended packages.

2.3 Public Data Sources

PAnnBuilder parses proteomics annotation data from public sources and build R annotation packages. It also provides convenient functions to access these sources. For example, you can get all supported databases for ”Homo sapiens” by:

```
> library(PAnnBuilder)          ## Load package
> getALLUrl("Homo sapiens")     ## Get urls
> getALLBuilt("Homo sapiens")  ## Get version/release
```

Detail description of all supported public data sources in PAnnBuilder are as follows:

UniProt The data `ftp://ftp.ebi.ac.uk/pub/databases/uniprot/knowledgebase` will be used to map protein UniProt identifiers to diverse annotation available in UniProt database.

IPI The data `ftp://ftp.ebi.ac.uk/pub/databases/IPI/current` will be used to map protein IPI identifiers to diverse annotation available in International Protein Index database.

RefSeq The data `ftp://ftp.ncbi.nih.gov/refseq` will be used to map protein RefSeq identifiers to diverse annotation available in NCBI RefSeq database.

Entrez Gene The data `ftp://ftp.ncbi.nih.gov/gene/DATA` will be used to annotate genes after the Entrez Gene identifiers have been obtained.

Gene Ontology The data `ftp://ftp.ebi.ac.uk/pub/databases/GO/goa/proteomes` and `http://archive.geneontology.org/latest-termdb` will be used to obtain gene ontology information.

KEGG Some data at `ftp://ftp.genome.jp/pub/kegg` will be used to obtain pathway information.

HomoloGene A data file provided by `ftp://ftp.ncbi.nlm.nih.gov/pub/HomoloGene/current/` will be used to extract mappings between GeneID/ProteinGI and HomoloGene ids.

InParanoid The data `http://inparanoid.sbc.su.se` will be used to obtain ortholog protein groups between two organisms.

Gene Interaction The data `ftp://ftp.ncbi.nih.gov/gene/GeneRIF` will be used to extract protein-protein interactions between Entrez GeneID or Protein RefSeq ids.

IntAct The data `ftp://ftp.ebi.ac.uk/pub/databases/intact/current/psimitab` will be used to extract protein-protein interactions between UniProt protein accession numbers.

MPPI The data `http://mips.gsf.de/proj/ppi/data/mppi.gz` will be used to extract protein-protein interactions between UniProt protein accession numbers.

3DID The data `http://gatealoy.pcb.ub.es/3did/download` will be used to extract domain-domain interactions between Pfam domain identifiers.

DOMINE The data `http://domine.utdallas.edu` will be used to extract domain-domain interactions between Pfam domain identifiers.

DBSubLoc The data `http://www.bioinfo.tsinghua.edu.cn/~guotao` will be used to obtain subcellular localization for protein from SWISS-PROT and PIR database.

BaCelLo The data `http://gpcr2.biocomp.unibo.it/bacello` will be used to map SwissProt eukaryotes protein identifiers to subcellular localization.

SCOP The data `http://scop.mrc-lmb.cam.ac.uk/scop` will be used to map PDB structure identifiers to SCOP domain identifiers.

PeptideAtlas The data `http://www.peptideatlas.org/builds` will be used to obtain experimentally identified peptides and their coordinates on chromosomes.

SysPTM The data `http://www.biosino.org/papers/SysPTM` will be used to obtain protein post-translational modifications information.

Sys-BodyFluid The data `http://www.biosino.org/papers/Sys-BodyFluid` will be used to map IPI protein identifiers to body fluids.

2.4 Annotation Packages Produced by PAnnBuilder

2.4.1 Packages produced by PAnnBuilder

PAnnBuilder has powerful ability to build R package for assembling proteome annotation. However, the process of building new package may be time-consuming because of the downloading and parsing of large data files. To make PAnnBuilder useful for any users, we have built many frequently used annotation packages in advance. These pre-built package can be downloaded via biocLite.

These packages are divided into two classes: enviroment-based packages built by "*Builder" functions (see Table ??); SQLite based packages built by "*Builder_DB" functions (see Table 1). They are widely used methods for building Bioconductor meta-data packages. Each SQLite-based annotation package (identified by a ".db" suffix in the package name) contains a number of AnnDbBimap objects in place of the environment objects found in the old-style environment-based annotation packages. In future, SQLite-based annotation package will replace environment-based packages.

The pre-built packages provide a quick start for R beginners. If one wants to analyze protein set in Human IPI database, the quickest way is to download and use SQLite based package "org.Hs.ipi.db". However, if the package one wants has not been built or a new-version database is released, new package should be built using functions in PAnnBuilder (See Section 4 for methods of building annotation packages).

2.4.2 Using annotation data package

SQLite-based *.db packages are capable of flexible data queries, reverse mapping, and data filtering. Vignette of "AnnotationDbi" detailedly described how to use SQLite based packages (`http://www.bioconductor.org/packages/release/bioc/vignettes/AnnotationDbi/inst/doc/AnnotationDbi.pdf`). Here package "org.Hs.ipi.db" produced by PAnnBuilder is illustrated as an example in the following code chuck.

1. Install and load annotation package. Package "org.Hs.ipi" is used as an example, other packages can be derived accordingly.

   ```
   > biocLite("org.Hs.ipi.db")
   ```

   ```
   > library(org.Hs.ipi.db) # Load annotation package
   ```

2. Browse data in the package.

   ```
   > ?org.Hs.ipiGENEID
   ```

3. Convert the environment object into a "list" object, and get values by index or name.

   ```
   > xx <- as.list(org.Hs.ipiGENEID)
   > xx[!is.na(xx)][1:3]

   $IPI00000001.2
   [1] "6780"

   $IPI00000005.1
   [1] "4893"
   ```

Table 1: SQLite-based Annotation Packges Produced by PAnnBuilder.

Description	**R Function**	**Source**	**Organism**	**Package**
complete and canonical annotaion for all proteins of a specific organism, including protein description, Entrez gene identifier, KEGG pathway, gene ontology, domain, and so on.	pBaseBuilder_DB	IPI	Homo sapiens Mus musculus Rattus norvegicus	org.Hs.ipi.db org.Mm.ipi.db org.Rn.ipi.db
		Swiss-Prot	Homo sapiens Mus musculus Rattus norvegicus	org.Hs.sp.db org.Mm.sp.db org.Rn.sp.db
		RefSeq	Homo sapiens Mus musculus Rattus norvegicus	org.Hs.ref.db org.Mm.ref.db org.Rn.ref.db
protein indentifier mapping	crossBuilder_DB	Swiss-Prot, IPI, RefSeq	Homo sapiens Mus musculus Rattus norvegicus	org.Hs.cross.db org.Mm.cross.db org.Rn.cross.db
protein-protein or domain-domain interaction data	intBuilder_DB	IntAct NCBI Gene MPPI 3did Domine		int.intact.db int.geneint.db int.mppi.db int.did.db int.domine.db
protein subcell location	subcellBuilder_DB	BaCelLo DBSubLoc		sc.bacello.db sc.dbsubloc.db
protein structure classification	scopBuilder_DB	SCOP		scop.db
protein post-translational modification	ptmBuilder_DB	SysPTM		sysptm.db
body fluid protein	bfBuilder_DB	SysBodyFluid	Homo sapiens	org.Hs.bf.db
homolog protein group	HomoloGeneBuilder_DB	HomoloGene		homolog.db
ortholog protein group	InParanoidBuilder_DB	InParanoid	Homo sapiens, Mus musculus	org.HsMm.ortholog.db
peptides identified by mass spectrometry	PeptideAtlasBuilder_DB	PeptideAtlas	Homo sapiens	org.Hs.pep.db
gene ontology	GOABuilder_DB	GOA	Homo sapiens	org.Hs.goa.db
identifier and name	dNameBuilder_DB			dName.db

```
$IPI00000006.1
[1] "3265"

> xx[["IPI00792103.1"]]

[1] "28957"
```

4. Specific utilities for SQLite-based *.db packages.

```
> ## Access the data in table (data.frame) format via function "toTable".
> toTable(org.Hs.ipiPATH[1:3])

          ipi_id  path_id
1  IPI00000005.1 hsa04010
2  IPI00000005.1 hsa04012
3  IPI00000005.1 hsa04062
4  IPI00000005.1 hsa04360
5  IPI00000005.1 hsa04370
6  IPI00000005.1 hsa04530
7  IPI00000005.1 hsa04540
8  IPI00000005.1 hsa04650
9  IPI00000005.1 hsa04660
10 IPI00000005.1 hsa04662
11 IPI00000005.1 hsa04664
12 IPI00000005.1 hsa04720
13 IPI00000005.1 hsa04722
14 IPI00000005.1 hsa04730
15 IPI00000005.1 hsa04810
16 IPI00000005.1 hsa04910
17 IPI00000005.1 hsa04912
18 IPI00000005.1 hsa04916
19 IPI00000005.1 hsa05200
20 IPI00000005.1 hsa05211
21 IPI00000005.1 hsa05213
22 IPI00000005.1 hsa05214
23 IPI00000005.1 hsa05215
24 IPI00000005.1 hsa05216
25 IPI00000005.1 hsa05218
26 IPI00000005.1 hsa05219
27 IPI00000005.1 hsa05220
28 IPI00000005.1 hsa05221
29 IPI00000005.1 hsa05223
30 IPI00000006.1 hsa04010
31 IPI00000006.1 hsa04012
```

```
32 IPI00000006.1 hsa04062
33 IPI00000006.1 hsa04144
34 IPI00000006.1 hsa04360
35 IPI00000006.1 hsa04370
36 IPI00000006.1 hsa04510
37 IPI00000006.1 hsa04530
38 IPI00000006.1 hsa04540
39 IPI00000006.1 hsa04650
40 IPI00000006.1 hsa04660
41 IPI00000006.1 hsa04662
42 IPI00000006.1 hsa04664
43 IPI00000006.1 hsa04720
44 IPI00000006.1 hsa04722
45 IPI00000006.1 hsa04730
46 IPI00000006.1 hsa04810
47 IPI00000006.1 hsa04910
48 IPI00000006.1 hsa04912
49 IPI00000006.1 hsa04916
50 IPI00000006.1 hsa05200
51 IPI00000006.1 hsa05211
52 IPI00000006.1 hsa05213
53 IPI00000006.1 hsa05214
54 IPI00000006.1 hsa05215
55 IPI00000006.1 hsa05216
56 IPI00000006.1 hsa05218
57 IPI00000006.1 hsa05219
58 IPI00000006.1 hsa05220
59 IPI00000006.1 hsa05221
60 IPI00000006.1 hsa05223
61 IPI00000013.1 hsa04142
```

```
> ## reverse the role of protein and pathway via
> ## function "revmap" or "reverseSplit".
> tmp1 <- revmap(org.Hs.ipiPATH)  ## return a AnnDbBimap Object
> class(tmp1)
```

```
[1] "AnnDbBimap"
attr(,"package")
[1] "AnnotationDbi"
```

```
> as.list(tmp1)[1]
```

```
$hsa04010
   [1] "IPI00000005.1"  "IPI00000006.1"  "IPI00000044.1"  "IPI00000073.2"
```

```
  [5] "IPI00000075.3"  "IPI00000076.2"  "IPI00000240.2"  "IPI00000867.5"
  [9] "IPI00000977.1"  "IPI00001671.1"  "IPI00001962.1"  "IPI00002089.5"
 [13] "IPI00002232.2"  "IPI00002411.3"  "IPI00002503.2"  "IPI00002685.1"
 [17] "IPI00002849.3"  "IPI00002857.2"  "IPI00003096.1"  "IPI00003145.1"
 [21] "IPI00003148.1"  "IPI00003366.1"  "IPI00003431.1"  "IPI00003478.2"
 [25] "IPI00003479.3"  "IPI00003783.1"  "IPI00003814.1"  "IPI00003865.1"
 [29] "IPI00003928.1"  "IPI00004272.2"  "IPI00004390.3"  "IPI00004838.2"
 [33] "IPI00004839.1"  "IPI00005104.1"  "IPI00005142.1"  "IPI00005553.2"
 [37] "IPI00005567.1"  "IPI00005729.1"  "IPI00005733.1"  "IPI00005741.1"
 [41] "IPI00005777.1"  "IPI00006192.2"  "IPI00006283.3"  "IPI00006752.2"
 [45] "IPI00007123.1"  "IPI00007128.2"  "IPI00007189.1"  "IPI00007307.1"
 [49] "IPI00007525.1"  "IPI00007577.1"  "IPI00007702.1"  "IPI00007792.1"
 [53] "IPI00007879.1"  "IPI00008115.3"  "IPI00008830.1"  "IPI00008837.3"
 [57] "IPI00008965.1"  "IPI00009008.2"  "IPI00010156.2"  "IPI00010184.4"
 [61] "IPI00010270.1"  "IPI00010271.3"  "IPI00010403.1"  "IPI00010466.3"
 [65] "IPI00010680.3"  "IPI00011071.2"  "IPI00011072.1"  "IPI00011172.1"
 [69] "IPI00011258.8"  "IPI00011488.4"  "IPI00011860.3"  "IPI00011866.1"
 [73] "IPI00011868.2"  "IPI00011879.2"  "IPI00011975.1"  "IPI00012036.1"
 [77] "IPI00012039.1"  "IPI00012042.1"  "IPI00012058.1"  "IPI00012136.1"
 [81] "IPI00012270.2"  "IPI00012512.2"  "IPI00012811.4"  "IPI00012866.2"
 [85] "IPI00012870.1"  "IPI00013031.1"  "IPI00014608.1"  "IPI00015148.3"
 [89] "IPI00015777.2"  "IPI00015811.1"  "IPI00015902.3"  "IPI00016099.2"
 [93] "IPI00016305.1"  "IPI00016709.2"  "IPI00016729.1"  "IPI00016784.1"
 [97] "IPI00016786.1"  "IPI00016958.1"  "IPI00017095.1"  "IPI00017149.1"
[101] "IPI00017305.2"  "IPI00017801.3"  "IPI00018195.3"  "IPI00018214.1"
[105] "IPI00018259.4"  "IPI00018274.1"  "IPI00018290.1"  "IPI00018671.1"
[109] "IPI00018880.3"  "IPI00019345.1"  "IPI00019459.1"  "IPI00019473.2"
[113] "IPI00019812.1"  "IPI00019956.2"  "IPI00020131.1"  "IPI00020134.2"
[117] "IPI00020258.3"  "IPI00020418.1"  "IPI00020431.2"  "IPI00020898.1"
[121] "IPI00020904.1"  "IPI00020950.3"  "IPI00021327.3"  "IPI00021382.1"
[125] "IPI00021780.1"  "IPI00021786.1"  "IPI00021792.1"  "IPI00021833.1"
[129] "IPI00022396.1"  "IPI00022536.1"  "IPI00023133.1"  "IPI00023138.1"
[133] "IPI00023547.1"  "IPI00023636.3"  "IPI00024166.3"  "IPI00024184.2"
[137] "IPI00024249.1"  "IPI00024251.1"  "IPI00024252.1"  "IPI00024253.1"
[141] "IPI00024578.1"  "IPI00024672.1"  "IPI00024673.2"  "IPI00024674.2"
[145] "IPI00024709.1"  "IPI00024733.4"  "IPI00025076.2"  "IPI00025087.3"
[149] "IPI00025474.1"  "IPI00025477.5"  "IPI00025499.3"  "IPI00025512.2"
[153] "IPI00026054.1"  "IPI00026108.1"  "IPI00026262.3"  "IPI00026407.1"
[157] "IPI00026612.1"  "IPI00026664.1"  "IPI00026836.3"  "IPI00026900.1"
[161] "IPI00026962.3"  "IPI00026987.1"  "IPI00027174.1"  "IPI00027454.2"
[165] "IPI00027464.8"  "IPI00027508.1"  "IPI00027721.1"  "IPI00027809.2"
[169] "IPI00027914.4"  "IPI00028232.1"  "IPI00028428.2"  "IPI00028483.1"
```

```
[173] "IPI00028915.3"  "IPI00029074.1"  "IPI00029104.1"  "IPI00029251.4" 
[177] "IPI00029260.2"  "IPI00029458.1"  "IPI00029502.3"  "IPI00029505.3" 
[181] "IPI00029643.3"  "IPI00029734.1"  "IPI00029760.1"  "IPI00030097.1" 
[185] "IPI00030278.4"  "IPI00030814.1"  "IPI00030919.3"  "IPI00030935.1" 
[189] "IPI00031476.1"  "IPI00031747.3"  "IPI00031777.2"  "IPI00031790.2" 
[193] "IPI00032908.1"  "IPI00033008.1"  "IPI00033016.2"  "IPI00033560.2" 
[197] "IPI00036489.1"  "IPI00037070.3"  "IPI00043401.2"  "IPI00045065.1" 
[201] "IPI00060723.2"  "IPI00063188.1"  "IPI00064296.1"  "IPI00064357.1" 
[205] "IPI00065432.7"  "IPI00065537.2"  "IPI00066574.1"  "IPI00068871.2" 
[209] "IPI00081833.1"  "IPI00103604.2"  "IPI00106506.2"  "IPI00107759.2" 
[213] "IPI00148567.6"  "IPI00149048.2"  "IPI00149094.4"  "IPI00152899.1" 
[217] "IPI00154603.4"  "IPI00155892.2"  "IPI00157561.1"  "IPI00157737.2" 
[221] "IPI00157813.6"  "IPI00157910.2"  "IPI00158128.1"  "IPI00158248.3" 
[225] "IPI00158733.4"  "IPI00160672.1"  "IPI00163320.3"  "IPI00163641.2" 
[229] "IPI00164934.3"  "IPI00165045.4"  "IPI00165947.3"  "IPI00168325.4" 
[233] "IPI00170811.1"  "IPI00170867.3"  "IPI00171312.1"  "IPI00171874.3" 
[237] "IPI00178352.6"  "IPI00179189.7"  "IPI00179415.4"  "IPI00179433.3" 
[241] "IPI00180375.6"  "IPI00181703.1"  "IPI00181738.3"  "IPI00182383.7" 
[245] "IPI00183471.2"  "IPI00183965.4"  "IPI00185860.6"  "IPI00186536.9" 
[249] "IPI00187091.6"  "IPI00215666.1"  "IPI00215763.1"  "IPI00216095.1" 
[253] "IPI00216196.1"  "IPI00216354.4"  "IPI00216357.1"  "IPI00216358.1" 
[257] "IPI00216359.1"  "IPI00216471.2"  "IPI00216507.4"  "IPI00216511.1" 
[261] "IPI00216602.3"  "IPI00216603.1"  "IPI00216604.1"  "IPI00216605.1" 
[265] "IPI00216859.3"  "IPI00217021.2"  "IPI00217024.1"  "IPI00217131.1" 
[269] "IPI00217484.1"  "IPI00217497.1"  "IPI00217498.17" "IPI00217499.8" 
[273] "IPI00217960.1"  "IPI00217976.4"  "IPI00218065.9"  "IPI00218070.1" 
[277] "IPI00218332.1"  "IPI00218333.1"  "IPI00218338.5"  "IPI00218399.2" 
[281] "IPI00218400.2"  "IPI00218401.2"  "IPI00218402.2"  "IPI00218404.2" 
[285] "IPI00218405.2"  "IPI00218416.1"  "IPI00218417.1"  "IPI00218418.1" 
[289] "IPI00218419.1"  "IPI00218420.1"  "IPI00218421.2"  "IPI00218422.1" 
[293] "IPI00218423.3"  "IPI00218424.3"  "IPI00218425.4"  "IPI00218566.1" 
[297] "IPI00218567.1"  "IPI00218687.3"  "IPI00218688.3"  "IPI00218689.3" 
[301] "IPI00218690.1"  "IPI00218857.1"  "IPI00218858.1"  "IPI00218862.1" 
[305] "IPI00218924.5"  "IPI00219084.2"  "IPI00219390.1"  "IPI00219391.1" 
[309] "IPI00219393.1"  "IPI00219394.3"  "IPI00219395.2"  "IPI00219396.1" 
[313] "IPI00219397.1"  "IPI00219398.1"  "IPI00219411.3"  "IPI00219509.1" 
[317] "IPI00219510.1"  "IPI00219537.3"  "IPI00219569.1"  "IPI00219584.3" 
[321] "IPI00219592.3"  "IPI00219601.2"  "IPI00219604.3"  "IPI00219607.2" 
[325] "IPI00219628.3"  "IPI00219675.1"  "IPI00219799.2"  "IPI00219800.1" 
[329] "IPI00219802.1"  "IPI00219817.1"  "IPI00219929.1"  "IPI00219933.2" 
[333] "IPI00219983.1"  "IPI00219984.1"  "IPI00220075.1"  "IPI00220156.1" 
[337] "IPI00220171.5"  "IPI00220173.2"  "IPI00220174.4"  "IPI00220175.5" 
```

```
[341] "IPI00220207.1"  "IPI00220209.3"  "IPI00220253.1"  "IPI00220254.1"
[345] "IPI00220305.1"  "IPI00220306.1"  "IPI00220318.1"  "IPI00220356.1"
[349] "IPI00220382.2"  "IPI00220383.2"  "IPI00220431.1"  "IPI00220438.1"
[353] "IPI00220454.1"  "IPI00220513.2"  "IPI00220514.1"  "IPI00220870.1"
[357] "IPI00220885.1"  "IPI00220983.1"  "IPI00221034.9"  "IPI00221141.3"
[361] "IPI00221142.3"  "IPI00221143.3"  "IPI00221227.1"  "IPI00221228.1"
[365] "IPI00221229.3"  "IPI00221232.9"  "IPI00221346.1"  "IPI00221347.1"
[369] "IPI00221348.1"  "IPI00234446.2"  "IPI00235003.10" "IPI00235354.1"
[373] "IPI00241935.2"  "IPI00247309.3"  "IPI00288892.2"  "IPI00289092.1"
[377] "IPI00289334.2"  "IPI00289547.4"  "IPI00289746.2"  "IPI00291314.1"
[381] "IPI00291347.5"  "IPI00291367.4"  "IPI00291990.1"  "IPI00291992.1"
[385] "IPI00292140.4"  "IPI00292181.4"  "IPI00292537.3"  "IPI00292635.1"
[389] "IPI00293683.4"  "IPI00293857.1"  "IPI00294486.1"  "IPI00295390.4"
[393] "IPI00295401.4"  "IPI00295738.1"  "IPI00296283.3"  "IPI00297845.4"
[397] "IPI00298726.2"  "IPI00299037.3"  "IPI00299240.6"  "IPI00299512.4"
[401] "IPI00300321.5"  "IPI00301527.4"  "IPI00302112.1"  "IPI00302181.4"
[405] "IPI00302592.2"  "IPI00303550.2"  "IPI00303797.3"  "IPI00304048.6"
[409] "IPI00304111.2"  "IPI00304235.1"  "IPI00304578.1"  "IPI00304925.5"
[413] "IPI00305469.4"  "IPI00305491.5"  "IPI00307223.6"  "IPI00307805.5"
[417] "IPI00328245.3"  "IPI00328744.6"  "IPI00329411.1"  "IPI00329638.11"
[421] "IPI00329757.7"  "IPI00332347.5"  "IPI00332838.3"  "IPI00333541.6"
[425] "IPI00334195.1"  "IPI00334512.5"  "IPI00335101.1"  "IPI00335248.1"
[429] "IPI00335706.4"  "IPI00336003.2"  "IPI00336017.5"  "IPI00336178.1"
[433] "IPI00339269.1"  "IPI00375287.4"  "IPI00375319.3"  "IPI00375340.4"
[437] "IPI00375455.2"  "IPI00376119.1"  "IPI00376687.2"  "IPI00376860.4"
[441] "IPI00377126.2"  "IPI00377178.2"  "IPI00377232.3"  "IPI00382394.9"
[445] "IPI00382665.4"  "IPI00382697.2"  "IPI00382698.2"  "IPI00382699.2"
[449] "IPI00382818.2"  "IPI00383065.1"  "IPI00383173.1"  "IPI00383208.1"
[453] "IPI00383509.1"  "IPI00383903.1"  "IPI00383961.1"  "IPI00384224.4"
[457] "IPI00384621.4"  "IPI00384672.3"  "IPI00384713.5"  "IPI00384771.1"
[461] "IPI00384906.1"  "IPI00384910.1"  "IPI00384922.6"  "IPI00385215.3"
[465] "IPI00385449.4"  "IPI00385645.1"  "IPI00385784.1"  "IPI00386089.3"
[469] "IPI00386260.6"  "IPI00386448.4"  "IPI00386650.3"  "IPI00394738.4"
[473] "IPI00394739.5"  "IPI00394837.2"  "IPI00395607.2"  "IPI00395625.1"
[477] "IPI00395654.7"  "IPI00396114.1"  "IPI00396630.3"  "IPI00396912.3"
[481] "IPI00397553.1"  "IPI00397554.6"  "IPI00398504.5"  "IPI00399054.2"
[485] "IPI00402119.4"  "IPI00409678.4"  "IPI00410124.1"  "IPI00410125.1"
[489] "IPI00410208.1"  "IPI00410216.1"  "IPI00410217.1"  "IPI00410218.1"
[493] "IPI00410219.1"  "IPI00410220.1"  "IPI00410223.1"  "IPI00410485.2"
[497] "IPI00410600.3"  "IPI00411343.6"  "IPI00411715.2"  "IPI00411984.5"
[501] "IPI00412316.1"  "IPI00412417.1"  "IPI00412433.1"  "IPI00412525.1"
[505] "IPI00413731.3"  "IPI00413939.3"  "IPI00413958.5"  "IPI00414365.1"
```

```
[509] "IPI00418221.4"  "IPI00419504.2"  "IPI00419782.3"  "IPI00419979.3"
[513] "IPI00420109.3"  "IPI00423568.1"  "IPI00423570.3"  "IPI00425663.1"
[517] "IPI00425919.2"  "IPI00426283.3"  "IPI00432402.1"  "IPI00435928.1"
[521] "IPI00440705.1"  "IPI00443029.1"  "IPI00446866.4"  "IPI00448764.1"
[525] "IPI00455036.4"  "IPI00455176.1"  "IPI00465135.3"  "IPI00465168.4"
[529] "IPI00465333.2"  "IPI00470535.1"  "IPI00472283.3"  "IPI00477536.3"
[533] "IPI00477982.1"  "IPI00478317.3"  "IPI00478527.2"  "IPI00478653.2"
[537] "IPI00478699.1"  "IPI00478791.2"  "IPI00479216.1"  "IPI00479444.2"
[541] "IPI00479470.2"  "IPI00479514.2"  "IPI00479608.6"  "IPI00479687.2"
[545] "IPI00479758.2"  "IPI00479798.2"  "IPI00479852.1"  "IPI00479926.3"
[549] "IPI00479997.4"  "IPI00480038.7"  "IPI00513741.2"  "IPI00513748.1"
[553] "IPI00513803.3"  "IPI00513863.2"  "IPI00513934.2"  "IPI00514108.3"
[557] "IPI00514131.6"  "IPI00514270.3"  "IPI00514420.3"  "IPI00514519.3"
[561] "IPI00514746.1"  "IPI00515114.2"  "IPI00552068.4"  "IPI00552416.4"
[565] "IPI00552653.2"  "IPI00552858.2"  "IPI00552941.2"  "IPI00553112.4"
[569] "IPI00553169.5"  "IPI00553172.1"  "IPI00555566.3"  "IPI00555621.3"
[573] "IPI00555640.3"  "IPI00555727.1"  "IPI00555871.1"  "IPI00556086.8"
[577] "IPI00556200.1"  "IPI00556393.1"  "IPI00556455.1"  "IPI00556516.1"
[581] "IPI00556557.1"  "IPI00556575.1"  "IPI00556585.1"  "IPI00604575.2"
[585] "IPI00604647.1"  "IPI00604745.3"  "IPI00607583.2"  "IPI00607609.1"
[589] "IPI00639798.1"  "IPI00639865.1"  "IPI00639883.3"  "IPI00640510.1"
[593] "IPI00640515.2"  "IPI00640563.1"  "IPI00641051.3"  "IPI00641071.3"
[597] "IPI00641117.2"  "IPI00641409.1"  "IPI00641592.2"  "IPI00641808.1"
[601] "IPI00641838.1"  "IPI00641984.2"  "IPI00642012.1"  "IPI00642411.2"
[605] "IPI00642537.3"  "IPI00642590.1"  "IPI00642606.2"  "IPI00642646.1"
[609] "IPI00642750.1"  "IPI00642867.1"  "IPI00642897.3"  "IPI00643115.1"
[613] "IPI00643152.2"  "IPI00643790.1"  "IPI00643986.1"  "IPI00644248.1"
[617] "IPI00644377.1"  "IPI00644413.2"  "IPI00644576.1"  "IPI00644803.1"
[621] "IPI00644841.1"  "IPI00645282.1"  "IPI00645543.1"  "IPI00645585.2"
[625] "IPI00645750.1"  "IPI00645788.2"  "IPI00645837.1"  "IPI00645961.1"
[629] "IPI00645990.1"  "IPI00646169.2"  "IPI00646929.1"  "IPI00647432.2"
[633] "IPI00647460.1"  "IPI00647477.1"  "IPI00647767.3"  "IPI00647907.2"
[637] "IPI00651725.1"  "IPI00651767.1"  "IPI00654673.2"  "IPI00654847.2"
[641] "IPI00655954.1"  "IPI00656051.1"  "IPI00656138.1"  "IPI00657767.3"
[645] "IPI00657795.1"  "IPI00657850.2"  "IPI00658101.1"  "IPI00658102.1"
[649] "IPI00658172.1"  "IPI00737155.1"  "IPI00738231.1"  "IPI00739094.1"
[653] "IPI00739595.1"  "IPI00741763.2"  "IPI00742705.3"  "IPI00742900.2"
[657] "IPI00743108.1"  "IPI00743309.2"  "IPI00743663.3"  "IPI00743802.1"
[661] "IPI00744363.4"  "IPI00744508.2"  "IPI00744618.3"  "IPI00744662.1"
[665] "IPI00744784.3"  "IPI00744894.1"  "IPI00745806.3"  "IPI00745931.1"
[669] "IPI00747283.4"  "IPI00747653.1"  "IPI00747748.1"  "IPI00747794.3"
[673] "IPI00747921.1"  "IPI00748556.2"  "IPI00748557.1"  "IPI00748581.3"
```

```
[677] "IPI00748604.2"  "IPI00749135.1"  "IPI00759758.1"  "IPI00759772.1"
[681] "IPI00759837.2"  "IPI00760722.1"  "IPI00783406.3"  "IPI00783893.1"
[685] "IPI00783975.4"  "IPI00784270.1"  "IPI00784816.1"  "IPI00786904.1"
[689] "IPI00787244.3"  "IPI00788843.1"  "IPI00788857.1"  "IPI00788893.1"
[693] "IPI00788928.1"  "IPI00788987.1"  "IPI00789111.2"  "IPI00789143.1"
[697] "IPI00789315.2"  "IPI00789349.1"  "IPI00789446.1"  "IPI00789533.1"
[701] "IPI00789592.1"  "IPI00789633.1"  "IPI00789708.2"  "IPI00789776.1"
[705] "IPI00789786.1"  "IPI00789844.2"  "IPI00789904.1"  "IPI00789918.1"
[709] "IPI00790124.1"  "IPI00790152.2"  "IPI00790187.1"  "IPI00790293.1"
[713] "IPI00790539.1"  "IPI00790757.1"  "IPI00790775.1"  "IPI00790776.4"
[717] "IPI00790874.1"  "IPI00790935.1"  "IPI00790968.1"  "IPI00791074.1"
[721] "IPI00791155.1"  "IPI00791238.1"  "IPI00791356.1"  "IPI00791404.1"
[725] "IPI00792086.1"  "IPI00792275.1"  "IPI00792571.1"  "IPI00792639.1"
[729] "IPI00792701.1"  "IPI00792746.1"  "IPI00792754.2"  "IPI00792812.1"
[733] "IPI00792891.1"  "IPI00793099.2"  "IPI00793141.1"  "IPI00793356.1"
[737] "IPI00793436.1"  "IPI00793439.1"  "IPI00793475.1"  "IPI00793507.2"
[741] "IPI00793679.1"  "IPI00793709.1"  "IPI00793781.1"  "IPI00793792.2"
[745] "IPI00793794.1"  "IPI00793942.3"  "IPI00794125.1"  "IPI00794137.2"
[749] "IPI00794157.1"  "IPI00794227.1"  "IPI00794282.1"  "IPI00794300.2"
[753] "IPI00794313.1"  "IPI00794357.2"  "IPI00794394.1"  "IPI00794433.1"
[757] "IPI00794469.1"  "IPI00794495.2"  "IPI00794507.1"  "IPI00794695.1"
[761] "IPI00794799.3"  "IPI00794872.2"  "IPI00794923.1"  "IPI00794968.1"
[765] "IPI00794995.2"  "IPI00795061.1"  "IPI00795187.1"  "IPI00795296.1"
[769] "IPI00795330.1"  "IPI00795397.1"  "IPI00795415.1"  "IPI00795591.1"
[773] "IPI00795667.1"  "IPI00795723.1"  "IPI00795867.1"  "IPI00795875.2"
[777] "IPI00795882.2"  "IPI00795934.1"  "IPI00795936.3"  "IPI00796046.1"
[781] "IPI00796052.3"  "IPI00796056.1"  "IPI00796068.2"  "IPI00796081.1"
[785] "IPI00796085.1"  "IPI00796154.2"  "IPI00796162.1"  "IPI00796188.3"
[789] "IPI00796237.1"  "IPI00796238.1"  "IPI00796240.1"  "IPI00796308.1"
[793] "IPI00796329.1"  "IPI00796349.1"  "IPI00796385.1"  "IPI00796408.1"
[797] "IPI00796432.1"  "IPI00796532.1"  "IPI00796549.1"  "IPI00796714.1"
[801] "IPI00796728.1"  "IPI00796796.1"  "IPI00796918.1"  "IPI00797131.1"
[805] "IPI00797216.1"  "IPI00797240.1"  "IPI00797277.1"  "IPI00797302.1"
[809] "IPI00797354.1"  "IPI00797598.1"  "IPI00797606.3"  "IPI00797720.1"
[813] "IPI00797722.1"  "IPI00797728.1"  "IPI00797850.1"  "IPI00797883.1"
[817] "IPI00798021.1"  "IPI00798101.1"  "IPI00798140.1"  "IPI00798166.1"
[821] "IPI00798172.1"  "IPI00798186.1"  "IPI00798192.1"  "IPI00798213.1"
[825] "IPI00798235.2"  "IPI00807463.2"  "IPI00815649.1"  "IPI00815689.1"
[829] "IPI00815762.1"  "IPI00816049.1"  "IPI00816059.1"  "IPI00816109.1"
[833] "IPI00816182.1"  "IPI00816223.1"  "IPI00816804.1"  "IPI00827513.2"
[837] "IPI00827543.1"  "IPI00827553.1"  "IPI00827578.1"  "IPI00827633.1"
[841] "IPI00827681.1"  "IPI00827766.1"  "IPI00827775.1"  "IPI00827851.2"
```

```
 [845] "IPI00827855.1"  "IPI00827887.2"  "IPI00827942.2"  "IPI00827956.1"
 [849] "IPI00827998.1"  "IPI00828000.1"  "IPI00828035.1"  "IPI00828116.2"
 [853] "IPI00828128.2"  "IPI00828174.2"  "IPI00828206.2"  "IPI00828210.1"
 [857] "IPI00829683.1"  "IPI00844074.1"  "IPI00844141.2"  "IPI00844572.2"
 [861] "IPI00845339.1"  "IPI00845373.1"  "IPI00847283.2"  "IPI00847452.1"
 [865] "IPI00852600.1"  "IPI00852607.2"  "IPI00852631.1"  "IPI00852680.1"
 [869] "IPI00852720.1"  "IPI00852753.1"  "IPI00852757.1"  "IPI00852775.1"
 [873] "IPI00852832.1"  "IPI00852879.1"  "IPI00852947.1"  "IPI00852959.1"
 [877] "IPI00853075.1"  "IPI00853109.1"  "IPI00853117.1"  "IPI00853140.1"
 [881] "IPI00853219.1"  "IPI00853317.1"  "IPI00853405.3"  "IPI00853572.1"
 [885] "IPI00853583.1"  "IPI00854815.2"  "IPI00854859.3"  "IPI00855724.2"
 [889] "IPI00855801.2"  "IPI00855890.2"  "IPI00855923.1"  "IPI00855985.2"
 [893] "IPI00856004.1"  "IPI00856031.2"  "IPI00856054.2"  "IPI00868677.2"
 [897] "IPI00871334.2"  "IPI00871339.1"  "IPI00871770.1"  "IPI00872126.2"
 [901] "IPI00872489.2"  "IPI00872930.1"  "IPI00873202.1"  "IPI00873275.2"
 [905] "IPI00873362.1"  "IPI00873480.1"  "IPI00873613.2"  "IPI00873863.2"
 [909] "IPI00874012.1"  "IPI00877088.1"  "IPI00877120.1"  "IPI00877126.2"
 [913] "IPI00877617.1"  "IPI00877619.2"  "IPI00877663.1"  "IPI00877816.1"
 [917] "IPI00877974.1"  "IPI00878066.1"  "IPI00879948.1"  "IPI00883897.1"
 [921] "IPI00884950.1"  "IPI00885089.1"  "IPI00885115.1"  "IPI00885132.1"
 [925] "IPI00885149.1"  "IPI00886879.1"  "IPI00887297.1"  "IPI00889565.1"
 [929] "IPI00889577.1"  "IPI00889582.2"  "IPI00889638.3"  "IPI00889699.1"
 [933] "IPI00889707.3"  "IPI00889718.1"  "IPI00892587.1"  "IPI00892591.1"
 [937] "IPI00892717.1"  "IPI00892738.1"  "IPI00892862.1"  "IPI00892963.1"
 [941] "IPI00893043.1"  "IPI00893099.1"  "IPI00893150.1"  "IPI00893213.1"
 [945] "IPI00893214.1"  "IPI00893281.1"  "IPI00893322.1"  "IPI00893667.1"
 [949] "IPI00893682.1"  "IPI00893685.1"  "IPI00893816.1"  "IPI00893964.1"
 [953] "IPI00894032.1"  "IPI00894151.1"  "IPI00894162.1"  "IPI00894243.1"
 [957] "IPI00894294.1"  "IPI00894303.1"  "IPI00894309.1"  "IPI00894391.1"
 [961] "IPI00894402.1"  "IPI00894418.1"  "IPI00894497.1"  "IPI00894510.1"
 [965] "IPI00895905.3"  "IPI00900275.2"  "IPI00900291.1"  "IPI00900293.1"
 [969] "IPI00900312.1"  "IPI00902720.1"  "IPI00902894.1"  "IPI00903123.2"
 [973] "IPI00908307.1"  "IPI00908435.2"  "IPI00908504.1"  "IPI00908754.1"
 [977] "IPI00908920.1"  "IPI00909103.2"  "IPI00909160.1"  "IPI00909190.1"
 [981] "IPI00909329.1"  "IPI00909404.1"  "IPI00909453.2"  "IPI00909460.1"
 [985] "IPI00909648.1"  "IPI00909675.1"  "IPI00909967.1"  "IPI00910047.1"
 [989] "IPI00910244.1"  "IPI00910457.1"  "IPI00910473.1"  "IPI00910494.1"
 [993] "IPI00910521.3"  "IPI00910530.1"  "IPI00910669.2"  "IPI00910746.1"
 [997] "IPI00910776.1"  "IPI00910794.2"  "IPI00910825.1"  "IPI00910863.1"
[1001] "IPI00910869.1"  "IPI00910870.1"  "IPI00910905.1"  "IPI00910923.1"
[1005] "IPI00911039.1"  "IPI00913887.1"  "IPI00913982.1"  "IPI00914604.1"
[1009] "IPI00915279.1"  "IPI00915358.1"  "IPI00915447.1"  "IPI00915452.2"
```

```
[1013] "IPI00915456.1"  "IPI00915882.1"  "IPI00915903.1"  "IPI00915990.1"
[1017] "IPI00916037.1"  "IPI00916174.1"  "IPI00916257.1"  "IPI00916263.1"
[1021] "IPI00916330.1"  "IPI00916405.1"  "IPI00916430.1"  "IPI00916492.1"
[1025] "IPI00916598.1"  "IPI00916611.1"  "IPI00916618.1"  "IPI00916767.1"
[1029] "IPI00916951.1"  "IPI00916960.1"  "IPI00916981.1"  "IPI00917017.1"
[1033] "IPI00917067.1"  "IPI00917101.1"  "IPI00917124.1"  "IPI00917203.1"
[1037] "IPI00917204.1"  "IPI00917241.1"  "IPI00917338.2"  "IPI00917423.1"
[1041] "IPI00917487.1"  "IPI00917525.1"  "IPI00917566.1"  "IPI00917851.1"
[1045] "IPI00917872.2"  "IPI00917915.1"  "IPI00917979.1"  "IPI00917998.1"
[1049] "IPI00918007.1"  "IPI00921149.1"  "IPI00921153.1"  "IPI00921346.4"
[1053] "IPI00921598.1"  "IPI00921996.1"  "IPI00922004.1"  "IPI00922192.1"
[1057] "IPI00922363.1"  "IPI00922436.1"  "IPI00922624.1"  "IPI00922901.1"
[1061] "IPI00923386.1"  "IPI00923387.1"  "IPI00923404.1"  "IPI00923405.1"
[1065] "IPI00923412.1"  "IPI00923417.1"  "IPI00923426.1"  "IPI00923428.1"
[1069] "IPI00923439.1"  "IPI00923440.1"  "IPI00923465.1"  "IPI00923469.1"
[1073] "IPI00923488.1"  "IPI00923509.1"  "IPI00923559.1"  "IPI00923568.1"
[1077] "IPI00923580.1"  "IPI00923589.1"  "IPI00923591.1"  "IPI00923608.1"
[1081] "IPI00923620.1"  "IPI00924436.1"  "IPI00924546.1"  "IPI00924548.2"
[1085] "IPI00924684.1"  "IPI00924851.1"  "IPI00924901.1"  "IPI00924942.1"
[1089] "IPI00924977.1"  "IPI00925308.1"  "IPI00925427.3"  "IPI00925452.1"
[1093] "IPI00925537.1"  "IPI00925565.1"  "IPI00925611.1"  "IPI00925642.1"
[1097] "IPI00925706.1"  "IPI00925820.1"  "IPI00925843.1"  "IPI00926072.1"
[1101] "IPI00926409.1"  "IPI00926683.1"  "IPI00926695.1"  "IPI00927016.1"
[1105] "IPI00927077.1"  "IPI00927161.1"  "IPI00927428.2"  "IPI00927467.1"
[1109] "IPI00927735.1"  "IPI00927825.1"  "IPI00927905.1"  "IPI00929159.1"
[1113] "IPI00929211.1"  "IPI00929227.1"  "IPI00929404.1"  "IPI00929541.1"
[1117] "IPI00929556.1"  "IPI00929637.1"  "IPI00929687.1"  "IPI00929689.1"
[1121] "IPI00930096.1"  "IPI00930156.1"  "IPI00930192.1"  "IPI00930194.1"
[1125] "IPI00930195.1"  "IPI00930294.1"  "IPI00930518.1"  "IPI00930557.1"
[1129] "IPI00930559.1"  "IPI00930650.1"  "IPI00930693.1"  "IPI00935431.1"
[1133] "IPI00935603.1"  "IPI00936281.2"  "IPI00936735.1"  "IPI00937463.1"
[1137] "IPI00937757.1"  "IPI00937867.1"  "IPI00938135.1"  "IPI00939442.1"
[1141] "IPI00939526.1"  "IPI00939549.1"  "IPI00939595.1"  "IPI00939729.1"
[1145] "IPI00939755.1"  "IPI00939788.1"  "IPI00939862.1"  "IPI00939921.2"
[1149] "IPI00940093.2"  "IPI00940146.1"  "IPI00940504.1"  "IPI00940553.1"
[1153] "IPI00940780.1"  "IPI00940813.1"  "IPI00940843.1"  "IPI00941127.1"
[1157] "IPI00941128.1"  "IPI00941179.1"  "IPI00941328.1"  "IPI00941430.1"
[1161] "IPI00941452.1"  "IPI00941909.1"  "IPI00941980.1"  "IPI00942094.1"
[1165] "IPI00942110.1"  "IPI00942120.1"  "IPI00942208.1"  "IPI00942214.1"
[1169] "IPI00942261.1"  "IPI00942501.1"  "IPI00942527.1"  "IPI00942576.1"
[1173] "IPI00942606.1"  "IPI00942823.1"  "IPI00942831.1"  "IPI00942942.1"
[1177] "IPI00943104.1"  "IPI00943289.1"  "IPI00943563.1"  "IPI00943898.2"
```

```
[1181] "IPI00943944.1"  "IPI00944898.1"  "IPI00944944.1"  "IPI00944963.1"
[1185] "IPI00945121.1"  "IPI00945123.1"  "IPI00945296.1"  "IPI00945313.1"
[1189] "IPI00945324.1"  "IPI00945340.2"  "IPI00945382.1"  "IPI00945424.1"
[1193] "IPI00945505.1"  "IPI00945610.1"  "IPI00945685.1"  "IPI00945692.1"
[1197] "IPI00945771.1"  "IPI00945776.1"  "IPI00945931.1"  "IPI00945940.1"
[1201] "IPI00946013.1"  "IPI00946109.1"  "IPI00946142.1"  "IPI00946154.1"
[1205] "IPI00946159.1"  "IPI00946292.1"  "IPI00946394.1"  "IPI00946576.1"
[1209] "IPI00946747.1"  "IPI00946753.1"  "IPI00946771.1"  "IPI00947284.1"
[1213] "IPI00947371.1"  "IPI00947394.1"  "IPI00947531.1"  "IPI00947542.1"
[1217] "IPI00947554.1"  "IPI00947556.2"  "IPI00952629.1"  "IPI00952817.1"
[1221] "IPI00952879.1"  "IPI00952975.1"  "IPI00953109.2"  "IPI00953141.1"
[1225] "IPI00953197.1"  "IPI00953206.1"  "IPI00953225.1"  "IPI00953262.1"
[1229] "IPI00953264.1"  "IPI00953285.1"  "IPI00953527.1"  "IPI00953645.1"
[1233] "IPI00953650.1"  "IPI00954560.1"  "IPI00954938.1"  "IPI00955858.1"
[1237] "IPI00956225.1"  "IPI00956260.1"  "IPI00956295.1"  "IPI00956346.1"
[1241] "IPI00956390.1"  "IPI00956452.1"  "IPI00963857.1"  "IPI00963883.1"
[1245] "IPI00964015.1"  "IPI00964039.1"  "IPI00964067.1"  "IPI00964164.1"
[1249] "IPI00964167.1"  "IPI00964213.1"  "IPI00964243.1"  "IPI00964374.1"
[1253] "IPI00964387.1"  "IPI00964478.1"  "IPI00964528.1"  "IPI00964579.1"
[1257] "IPI00964592.1"  "IPI00964611.1"  "IPI00964615.1"  "IPI00964641.1"
[1261] "IPI00964662.1"  "IPI00964667.1"  "IPI00964670.1"  "IPI00964687.1"
[1265] "IPI00964757.1"  "IPI00964816.1"  "IPI00964921.1"  "IPI00964932.1"
[1269] "IPI00965012.1"  "IPI00965055.1"  "IPI00965139.1"  "IPI00965195.1"
[1273] "IPI00965504.1"  "IPI00965511.1"  "IPI00965562.1"  "IPI00965602.1"
[1277] "IPI00965676.1"  "IPI00965760.1"  "IPI00965825.1"  "IPI00965895.1"
[1281] "IPI00966035.1"  "IPI00966051.1"  "IPI00966095.1"  "IPI00966109.1"
[1285] "IPI00966179.1"  "IPI00966195.1"  "IPI00966241.1"  "IPI00966245.1"
[1289] "IPI00966251.1"  "IPI00966310.1"  "IPI00966317.1"  "IPI00966376.1"
[1293] "IPI00966433.1"  "IPI00966491.1"  "IPI00966502.1"  "IPI00966576.1"
[1297] "IPI00966639.1"  "IPI00966790.1"  "IPI00966827.1"  "IPI00966866.1"
[1301] "IPI00966870.1"  "IPI00966920.1"  "IPI00966975.1"  "IPI00966992.1"
[1305] "IPI00966996.1"  "IPI00967077.1"  "IPI00967081.1"  "IPI00967108.1"
[1309] "IPI00967121.1"  "IPI00967123.1"  "IPI00967142.1"  "IPI00967162.1"
[1313] "IPI00967325.1"  "IPI00967368.1"  "IPI00967402.1"  "IPI00967472.1"
[1317] "IPI00967552.1"  "IPI00967630.1"  "IPI00967652.1"  "IPI00967704.1"
[1321] "IPI00967705.1"  "IPI00967763.1"  "IPI00967780.1"  "IPI00967839.1"
[1325] "IPI00967879.1"  "IPI00967948.1"  "IPI00967958.1"  "IPI00967977.1"
[1329] "IPI00967981.1"  "IPI00968000.1"  "IPI00968035.1"  "IPI00968099.1"
[1333] "IPI00968139.1"  "IPI00968165.1"  "IPI00968201.1"  "IPI00968205.1"
[1337] "IPI00968216.1"  "IPI00968284.1"  "IPI00969053.1"  "IPI00969378.1"
[1341] "IPI00969541.1"
```

```
> tmp2 <- reverseSplit(as.list(org.Hs.ipiPATH)) ## return a list Object
```

```
> class(tmp2)

[1] "list"

> tmp2[1]

$hsa00010
  [1] "IPI00003925.6" "IPI00005118.2" "IPI00006663.1" "IPI00009744.3"
  [5] "IPI00009790.1" "IPI00015911.2" "IPI00016768.3" "IPI00018031.1"
  [9] "IPI00018246.5" "IPI00019500.3" "IPI00021338.2" "IPI00022430.1"
 [13] "IPI00024087.3" "IPI00026663.2" "IPI00027165.3" "IPI00027497.5"
 [17] "IPI00028155.2" "IPI00060200.3" "IPI00073772.5" "IPI00102864.3"
 [21] "IPI00103467.4" "IPI00103892.1" "IPI00148061.3" "IPI00166751.3"
 [25] "IPI00169383.3" "IPI00184775.3" "IPI00215979.3" "IPI00216171.3"
 [29] "IPI00216792.2" "IPI00216932.4" "IPI00217872.3" "IPI00217966.9"
 [33] "IPI00218407.6" "IPI00218474.6" "IPI00218570.6" "IPI00218767.1"
 [37] "IPI00218768.1" "IPI00218896.3" "IPI00218899.6" "IPI00219018.7"
 [41] "IPI00219217.3" "IPI00219526.6" "IPI00219568.4" "IPI00219585.4"
 [45] "IPI00220271.3" "IPI00220644.8" "IPI00220663.3" "IPI00220665.6"
 [49] "IPI00220667.3" "IPI00221234.7" "IPI00244083.4" "IPI00248961.1"
 [53] "IPI00292698.4" "IPI00292709.2" "IPI00296183.7" "IPI00298423.3"
 [57] "IPI00299456.4" "IPI00306044.3" "IPI00306301.2" "IPI00329593.3"
 [61] "IPI00332371.9" "IPI00333619.4" "IPI00374975.2" "IPI00382746.1"
 [65] "IPI00384116.3" "IPI00384883.3" "IPI00384967.3" "IPI00385347.1"
 [69] "IPI00386733.2" "IPI00394758.1" "IPI00413730.4" "IPI00418262.5"
 [73] "IPI00451401.3" "IPI00465025.1" "IPI00465028.7" "IPI00465179.4"
 [77] "IPI00465248.5" "IPI00465343.3" "IPI00465439.5" "IPI00473031.6"
 [81] "IPI00479186.7" "IPI00479877.4" "IPI00513830.1" "IPI00549564.6"
 [85] "IPI00549725.6" "IPI00549885.4" "IPI00550060.2" "IPI00550364.8"
 [89] "IPI00550486.1" "IPI00552290.3" "IPI00552617.3" "IPI00553155.1"
 [93] "IPI00554498.3" "IPI00555601.1" "IPI00555722.1" "IPI00555728.1"
 [97] "IPI00555848.5" "IPI00556013.1" "IPI00556284.2" "IPI00556385.1"
[101] "IPI00556477.1" "IPI00604528.3" "IPI00607708.3" "IPI00639981.1"
[105] "IPI00640568.1" "IPI00640862.1" "IPI00642546.1" "IPI00642664.1"
[109] "IPI00642732.3" "IPI00643196.1" "IPI00643575.2" "IPI00644994.1"
[113] "IPI00645848.1" "IPI00646468.1" "IPI00647702.1" "IPI00654709.1"
[117] "IPI00743142.2" "IPI00743713.3" "IPI00746777.3" "IPI00759806.1"
[121] "IPI00784216.3" "IPI00788640.1" "IPI00788737.1" "IPI00788836.1"
[125] "IPI00788938.1" "IPI00789081.1" "IPI00789134.4" "IPI00789171.1"
[129] "IPI00789173.1" "IPI00789301.1" "IPI00790892.1" "IPI00791170.1"
[133] "IPI00791428.2" "IPI00791564.2" "IPI00791666.1" "IPI00792207.2"
[137] "IPI00792375.1" "IPI00792448.1" "IPI00792655.1" "IPI00792715.1"
[141] "IPI00793665.1" "IPI00793922.1" "IPI00794508.1" "IPI00794605.1"
```

```
[145] "IPI00794991.1" "IPI00795075.1" "IPI00795257.3" "IPI00795549.1"
[149] "IPI00795622.2" "IPI00795914.1" "IPI00796111.1" "IPI00796116.1"
[153] "IPI00796333.1" "IPI00796633.1" "IPI00796735.1" "IPI00796823.1"
[157] "IPI00796852.1" "IPI00797038.1" "IPI00797221.7" "IPI00797270.4"
[161] "IPI00797580.1" "IPI00798351.1" "IPI00815786.1" "IPI00815793.1"
[165] "IPI00815950.1" "IPI00830064.1" "IPI00844133.1" "IPI00847989.3"
[169] "IPI00871353.1" "IPI00872487.1" "IPI00872991.2" "IPI00873455.1"
[173] "IPI00902542.1" "IPI00903226.1" "IPI00903303.1" "IPI00908386.1"
[177] "IPI00908791.2" "IPI00908881.2" "IPI00908927.1" "IPI00909143.1"
[181] "IPI00909158.1" "IPI00909256.1" "IPI00909325.1" "IPI00909560.1"
[185] "IPI00909595.1" "IPI00909694.1" "IPI00909829.1" "IPI00909949.1"
[189] "IPI00910420.1" "IPI00910642.1" "IPI00910682.2" "IPI00910754.1"
[193] "IPI00910781.1" "IPI00910974.1" "IPI00910979.1" "IPI00913991.1"
[197] "IPI00915933.1" "IPI00916206.1" "IPI00916818.1" "IPI00916990.1"
[201] "IPI00916994.1" "IPI00917139.1" "IPI00917193.1" "IPI00917237.1"
[205] "IPI00917473.1" "IPI00917841.1" "IPI00922697.2" "IPI00925520.1"
[209] "IPI00926110.1" "IPI00926319.1" "IPI00926810.1" "IPI00927039.1"
[213] "IPI00927177.1" "IPI00927398.1" "IPI00927598.1" "IPI00927949.1"
[217] "IPI00930416.1" "IPI00936002.1" "IPI00939286.1" "IPI00939339.1"
[221] "IPI00939637.1" "IPI00940003.1" "IPI00940201.1" "IPI00940629.1"
[225] "IPI00941093.1" "IPI00941338.1" "IPI00941899.1" "IPI00942494.1"
[229] "IPI00942961.1" "IPI00945309.1" "IPI00945466.1" "IPI00945625.1"
[233] "IPI00945694.1" "IPI00945766.2" "IPI00945873.1" "IPI00946018.1"
[237] "IPI00946160.1" "IPI00946173.1" "IPI00946252.1" "IPI00946400.1"
[241] "IPI00946404.1" "IPI00946812.1" "IPI00947127.1" "IPI00947129.1"
[245] "IPI00947319.1" "IPI00952697.1" "IPI00952747.1" "IPI00952964.1"
[249] "IPI00953501.1" "IPI00955788.1" "IPI00955815.2" "IPI00955977.1"
[253] "IPI00964325.1" "IPI00964589.1" "IPI00964823.1" "IPI00964982.1"
[257] "IPI00965164.1" "IPI00965186.1" "IPI00965370.1" "IPI00965480.1"
[261] "IPI00965781.1" "IPI00966014.1" "IPI00966461.1" "IPI00966505.1"
[265] "IPI00966616.1" "IPI00966735.1" "IPI00966846.1" "IPI00967028.1"
[269] "IPI00967098.1" "IPI00967275.1" "IPI00969124.1"
```

```
> ## The left and right keys of the Bimap can be extracted
> ## using "Lkeys" and "Rkeys".
> Lkeys(org.Hs.ipiPATH)[1:3]
```

```
[1] "IPI00000005.1" "IPI00000006.1" "IPI00000013.1"
```

```
> Rkeys(org.Hs.ipiPATH)[1:3]
```

```
[1] "hsa04010" "hsa04012" "hsa04062"
```

```
> ## Get the create table statements.
> org.Hs.ipi_dbschema()

--
-- IPI_DB schema
-- ===============
--

CREATE TABLE basic (
  ipi_id CHAR(13) NOT NULL,               -- IPI Protein Identifier
  ipi_ac VARCHAR(20) NOT NULL,               -- IPI Acession Number in current vers
  len INTEGER NOT NULL,
  mw INTEGER NOT NULL,
  de VARCHAR(255) NOT NULL,                 -- Protein Description
  symbol VARCHAR(80) NOT NULL,
  sp_ac VARCHAR(20) NOT NULL,               -- Swiss-Prot Primary Acession Number
  sp_id VARCHAR(20) NOT NULL,               -- Swiss-Prot Protein Identifier
  ref_id VARCHAR(20) NOT NULL,               -- RefSeq accession number
  gi VARCHAR(10) NOT NULL,               -- NCBI Protein GI
  gene_id VARCHAR(10) NOT NULL       -- Entrez Gene ID
  unigene_id VARCHAR(10) NOT NULL,              -- UniGene ID
  kegg_id VARCHAR(20) NOT NULL,             -- KEGG gene ID
  FOREIGN KEY (ipi_id)
);

CREATE TABLE seq (
  ipi_id CHAR(13) NOT NULL,               -- IPI Protein Identifier
  seq text NOT NULL,               -- Protein Sequence
  FOREIGN KEY (ipi_id)
);

CREATE TABLE ipiac (
  ipi_id CHAR(13) NOT NULL,               -- IPI Protein Identifier
  ipi_acs CHAR(11) NOT NULL,               -- IPI Protein Acession Number
  FOREIGN KEY (ipi_id)
);
CREATE TABLE go (
  ipi_id CHAR(13) NOT NULL,               -- IPI Protein Identifier
  go_id CHAR(10) NOT NULL,                      -- GO ID
  evidence CHAR(3) NOT NULL,                    -- GO evidence code
  ontology CHAR(2) NOT NULL,                    -- GO ontology
  FOREIGN KEY (ipi_id)
);
```

```
CREATE TABLE path (
  ipi_id CHAR(13) NOT NULL,              -- IPI Protein Identifier
  path_id CHAR(5) NOT NULL,                    -- KEGG pathway short ID
  FOREIGN KEY (ipi_id)
};
CREATE TABLE pfam (
  ipi_id CHAR(13) NOT NULL,              -- IPI Protein Identifier
  pfam_id CHAR(7) NULL,                         -- Pfam ID
  FOREIGN KEY (ipi_id)
);
CREATE TABLE interpro (
  ipi_id CHAR(13) NOT NULL,              -- IPI Protein Identifier
  interpro_id CHAR(9) NOT NULL,                  -- InterPro ID
  FOREIGN KEY (ipi_id)
);
CREATE TABLE prosite (
  ipi_id CHAR(13) NOT NULL,              -- IPI Protein Identifier
  prosite_id CHAR(7) NULL,                       -- PROSITE ID
  FOREIGN KEY (ipi_id)
);
```

```
> ## Use "SELECT" SQL query.
> selectSQL<-paste("SELECT ipi_id, de",
+            "FROM basic",
+            "WHERE de like '%histone%'")
> tmp3 <- dbGetQuery(org.Hs.ipi_dbconn(), selectSQL)
> tmp3[1:3,]
```

```
         ipi_id                                                      de
1 IPI00001830.1 HETEROCHROMATIN-SPECIFIC NONHISTONE PROTEIN (FRAGMENT).
2 IPI00002220.4   HISTONE DEACETYLASE COMPLEX SUBUNIT SAP130 ISOFORM A.
3 IPI00002831.4             HISTONE DEACETYLASE COMPLEX SUBUNIT SAP30L.
```

3 Function Description

3.1 Getting URL and Version

To download data file from public database, the first step is getting its URL and release/version information. URLs of supported databases are stored in ”data/sourceURLs.txt”. Following functions are used to get the url:

- `getSrcUrl` - return url according given database.
- `getALLUrl` - return urls for all databases used in PAnnBuilder packages.

- `getSrcBuilt` - return release/version according given database.
- `getALLBuilt` - return release/version information for all databases used in PAnnBuilder packages.

3.2 Parsing and Writing Data

Parsing is a key step to convert original data file to R object. Sometimes R is directlly used to parse and write data. But for large data file or complicated data format, perl is firstly employed to quickly process data, and then R function reads the result file into R objects.

3.2.1 Employing perl program to parse data

Segment of perl program is written into file in ”inst/scripts”. Name and function of these parser files are as follows:

- spParser - parse protein data from SwissProt or TrEMBL
- ipiParser - parse protein data from IPI
- refseqParser - parse protein data from NCBI RefSeq
- equalParser - find protein ID mapping with equal sequences
- mergeParser - merge different ID mapping files
- mppiParser - parse protein protein interaction data from MIPS
- paParser - parse data from PeptideAtlas
- dbsublocParser - parse data from DBSubLoc
- pfamNameParser - parse domain id and name from Pfam
- blastParser - filter the results of blast

Function `fileMuncher` and `fileMuncher_DB` perl file based on given parser file and additional input data file, then perform this perl program via R.

3.2.2 Writing data using R

Besides using perl program, R functions also parse data from simple data file and store them as R environment objects or Bimap objects.

- `createEmptyDPkg` - create an empty R packge at given directory.
- `writeSQ` - write sequence data into R package.

- `writeName` - parse mulitple data file, and write the mapping of id and name into R package. It employs `writeGOName`, `writeKEGGName`, `writePFAMName`, `writeINTERPROName` and `writeTAXName` to respectively write data from GO, KEGG, Pfam, InterPro and TAX.

- `writeSCOPData` - parse structural classification of proteins from SCOP database.

- `writeSubCellData` - parse data from protein subcellular location databases, and write into R package. It employs `writeBACELLOData` and `writeDBSUBLOCData` to respectively write data from BaCelLo and DBSubLoc.

- `writeIntData` - parse data from protein-protein/domain-domain interaction databases, and write into R package. It employs `writeGENEINTData`, `writeINTACTData`, `writeMPPIData`, `write3DIDData` and `writeDOMINEData` to respectively write data from NCBI gene interaction data file, EBI intact, MIPS interaction data, 3DID database and DOMINE database.

- `writePtmData` - parse database involving protein post-translational modifications, and write into R package. It employs `writeSYSPTMData` to write data from SysPTM database.

- `writeBfData` - write data involving body fulids proteomics into R package. It employs `writeSYSBODYFLUIDData` to write data from Sys-BodyFluid database.

- `writeGOAData` - write gene ontology terms from GOA database into R package.

- `writeHomoloGeneData` - write homolog groups from NCBI HomoloGene into R package.

- `writeInParanoidData` - write paralog groups from InParanoid into R package.

- `writePeptideAtlasData` - write peptides identified by Mass Spectrometry from PeptideAtlas database into R package.

- `writeMeta_DB` - write meta information about the annotation package into SQLite-based package.

- `writeData_DB` - parse data from databases, and write data as tables into SQLite-based R package. It employs `writeSPData_DB`, `writeIPIData_DB`, `writeREFSEQData_DB`, `writeGENEINTData_DB`, `writeINTACTData_DB`, `writeMPPIData_DB`, `write3DIDData_DB`, `writeDOMINEData_DB`, `writeSYSBODYFLUIDData_DB`, `writeSYSPTMData_DB`, `writeSCOPData_DB`, `writeBACELLOData_DB`, `writeDBSUBLOCData_DB`, `writeGOAData_DB`, `writeHomoloGeneData_DB`, `writeInParanoidData_DB`, and `writePeptideAtlasData_DB` to respectively write data from Swiss-Prot, IPI, NCBI RefSeq database, and so on.

- `writeName_DB` - parse mulitple data file, and write the mapping of id and name into SQLite-based R package. It employs `writeGOName_DB`, `writeKEGGName_DB`, `writePFAMName_DB`, `writeINTERPROName_DB` and `writeTAXName_DB` to respectively write data from GO, KEGG, Pfam, InterPro and TAX.
- `createSeeds` - define a list of AnnDbBimap objects which indicates key and value of .
- `createAnnObjs` - produce AnnDbBimap objects based on the definiation in `createSeeds`.

3.3 Writing Help Documents

Help documents is an important part for new package. Diverse templates of help documents are stored in the "inst/templates" directory. When building new package, R functions use these templates to create "*.rd" help file in the "man" directory:

- `getRepList` - return a list which will replace the symbols in template file.
- `copyTemplates_DB` - implement similar function with `copyTemplates`, and is specially developped for SQLite-based annotation package.
- `writeDescription_DB` - implement similar function with `writeDescription`, and is specially developped for SQLite-based annotation package.

3.4 Building Data Packages

Basic functions described above make it possible to build proteomic annotation data packages. Based on these, PAnnBuilder develops multiple sophisticated functions to assemble proteomic annotaion data. Each function is implemented by the "*Builder_DB" R function.

- `pBaseBuilder_DB` - build annotation data packages for primary protein database such as SwissProt, TREMBL, IPI or NCBI RefSeq protein data.
- `pSeqBuilder_DB` - build annotation data packages for query protein sequences based on sequence similarity.
- `crossBuilder_DB` - build annotation data packages for protein id mapping in SwissProt, Trembl, IPI and NCBI Refseq databases.
- `subcellBuilder_DB` - build annotation data packages for protein subcellular location from BaCelLo or DBSubLoc database.
- `HomoloGeneBuilder_DB` - build annotation data packages for homolog protein group from NCBI HomoloGene database.
- `InParanoidBuilder_DB` - build annotation data packages for ortholog protein group between two given organisms from InParanoid database.

- `GOABuilder_DB` - build annotation data packages for mapping proteins of UniProt to Gene Ontolgy from GOA database.
- `scopBuilder_DB` - build annotation data packages for Structural Classification of Proteins.
- `intBuilder_DB` - build annotation data packages for protein-protein or domain-domain interaction from IntAct, MPPI, 3DID, DOMINE or NCBI Gene interaction database.
- `PeptideAtlasBuilder_DB` - build annotation data packages for experimentally identified peptides from PeptideAtlas database.
- `ptmBuilder_DB` - build annotation data packages for post-translational modifications from SysPTM database.
- `bfBuilder_DB` - build annotation data packages for proteins in body fluids from SysBodyFluid database.
- `dNameBuilder_DB` - build annotation data packages for mapping between entry ID and name from GO, KEGG, Pfam, InterPro and NCBI Taxonomy databases.

4 Building Annotation Data Packages

1. The first thing you need to do is setting basic parameters such as ”pkgpath”, ”version”, and ”author”.

```
> # Set path, version and author for the package.
> library(PAnnBuilder)
> pkgPath <- tempdir()
> version <- "1.0.0"
> author <- list()
> author[["authors"]] <- "Hong Li"
> author[["maintainer"]] <- "Hong Li <sysptm@gmail.com>"
```

2. Then you can run diverse ”*Builder_DB” functions to build packages by yourselves. `pBaseBuilder_DB`, `subcellBuilder_DB`, and `pSeqBuilder_DB` are taken as examples to build annotation packages.

pBaseBuilder_DB builds annotation data packages for proteins in three primary protein databases (SwissProt, IPI, RefSeq). It is a convenient way to obtain complete and canonical annotaion, including protein description, Entrez gene identifier, KEGG pathway, gene ontology, domain, coordinates on chromosomes and so on. For example, if you want to bulid annotation package for Mouse IPI database, you can use codes as follows:

```
>  ## Build SQLite based annotation package "org.Mm.ipi.db"
>  ## for Mouse IPI database.
>  ## Note: Perl is needed for parsing data file.
>  ##       Rtools is needed for Windows user.
>  pBaseBuilder_DB(baseMapType = "ipi", organism = "Mus musculus",
+                 prefix = "org.Mm.ipi", pkgPath = pkgPath, version = versio
+                 author = author)
```

After running, a subdirectory called "org.Mm.ipi" will be produced in the path given by "pkgPath". This directory contains all data and files, which can be used to build R package by "R CMD build" command.

subcellBuilder_DB builds annotation data Package which provides protein subcellular location information.

```
>  ## Build subcellular location annotation package "sc.bacello.db"
>  ## from BaCelLo database.
>  subcellBuilder_DB(src="BaCelLo", prefix="sc.bacello",
+                   pkgPath, version, author)
>   ## List all files in created directory "sc.bacello.db".
>   dir(file.path(pkgPath,"sc.bacello.db"))

[1] "DESCRIPTION" "NAMESPACE"   "R"           "inst"        "man"
```

pSeqBuilder_DB uses blast to calculate sequence similarity between query proteins and subject proteins, then assign annotation for query protein according to existing annotation of its similar proteins. pSeqBuilder_DB is useful for proteins which have not well annotated. Following code chunk gives an example for annotation query proteins by pSeqBuilder_DB. Needed R packages *org.Hs.sp.db*, *org.Hs.ipi.db*, can be downloaded via biocLite.

```
> ## Read query sequence.
> tmp = system.file("extdata", "query.example", package="PAnnBuilder")
> tmp = readLines(tmp)
> tag = grep("^>",tmp)
> query <- sapply(1:(length(tag)-1), function(x){
+      paste(tmp[(tag[x]+1):(tag[x+1]-1)], collapse="") })
> query <- c(query, paste(tmp[(tag[length(tag)]+1):length(tmp)], collapse="")
> names(query) = sub(">","",tmp[tag])
> ## Set parameters for sequence similarity.
> blast <- c("blastp", "10.0", "BLOSUM62", "0", "-1", "-1", "T", "F")
> names(blast) <- c("p","e","M","W","G","E","U","F")
> match <- c(0.00001, 0.9, 0.9)
> names(match) <- c("e","c","i")
> ## Install ackages "org.Hs.sp", "org.Hs.ipi".
> if( !require("org.Hs.sp.db") ){
+   biocLite("org.Hs.sp.db")
```

```
+ }
> if( !require("org.Hs.ipi.db") ){
+   biocLite("org.Hs.ipi.db")
+ }
> ## Use packages "org.Hs.sp.db", "org.Hs.ipi.db" to produce annotation
> ## R package for query sequence.
>   annPkgs = c("org.Hs.sp.db","org.Hs.ipi.db")
>   seqName = c("org.Hs.spSEQ","org.Hs.ipiSEQ")
>   pSeqBuilder_DB(query, annPkgs, seqName, blast, match,
+   prefix="test1", pkgPath, version, author)
```

3. After the running of ”*Builder_DB” function has been finished, a subdirectory named ”pkgName” will be produced in given ”pkgPath”. Then the command ”R CMD build” can be used to build source R package, and ”R CMD –binary build” can be used to build binary R package for Windows.

4. Note:

    - Web connection is needed to download files from public databases, and Perl is needed to parse data files. Additionally, Rtools is needed for Windows user.
    - Users should be aware that downloading, parsing, and saving data may take a long time, in addition to requiring enough disk space to store temporary data files.
    - ”R CMD build” and ”R CMD –binary build” should be used in command line, not in R. Detailed document about how to create your own packages can be found in the book ”Writing R Extensions” (`http://cran.r-project.org/doc/manuals/R-exts.pdf`). For Windows users, ”R CMD build” needs you to have installed the files for building source packages (which is the default), as well as the Windows toolset (see the ”R Installation and Administration” manual at `http://cran.r-project.org/doc/manuals/R-admin.pdf`).

5 Session Information

This vignette was generated using the following package versions:

```
R version 3.3.0 (2016-05-03)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 14.04.4 LTS

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        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

attached base packages:
[1] stats4    parallel  stats     graphics  grDevices utils     datasets
[8] methods   base

other attached packages:
[1] org.Hs.ipi.db_1.3.0  PAnnBuilder_1.36.0   AnnotationDbi_1.34.0
[4] IRanges_2.6.0        S4Vectors_0.10.0     Biobase_2.32.0
[7] BiocGenerics_0.18.0  RSQLite_1.0.0        DBI_0.4

loaded via a namespace (and not attached):
[1] tools_3.3.0
```

References

O. N. Jensen. Interpreting the protein language using proteomics. *Nat Rev Mol Cell Biol*, 7 (6):391–403, 2006.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. URL `http://www.R-project.org`. ISBN 3-900051-07-0.

J. Zhang, V. Carey, and R. Gentleman. An extensible application for assembling annotation for genomic data. *Bioinformatics*, 19(1):155–6, 2003.