

RSNPper: utilities for SNP data

VJ Carey `stvjc@channing.harvard.edu`

November 1, 2004

Contents

1	Introduction	1
2	How it works	1
3	Overview of the functions	3
4	Demonstrations	3
4.1	Obtaining information on genes	3
4.2	Obtaining information on SNPs	5
5	Application: SNP density on chr 1	8

1 Introduction

This document describes `RSNPper` version 1.0, added to Bioconductor in October of 2003. This first version focuses on SNP metadata, with functions that retrieve SNP-related data from the Boston Children's Hospital Informatics Program `SNPper` web service ?.

Earlier non-released versions of this package included considerable code for working with `prettybase` format and for conducting other tasks in SNP discovery projects. That material has been moved to `inst/OLD` and may be re-introduced later. Users seeking legacy support should contact the author.

2 How it works

Loading required package: `XML`

The core of this package is the XML-RPC service at `CHIP` accessible through the following URL stub:

```
> print(.SNPperBaseURL)
```

```
[1] "http://snpper.chip.org/bio/rpcserv/dummy?cmd="
```

The `useSNPper` function allows you to work directly with the XML-RPC server by packing up appropriate command and argument strings.

```
> dput(useSNPper)
```

```
function (cmd, parmstring)
{
  targ <- url(paste(.SNPperBaseURL, cmd, parmstring, sep = ""))
  open(targ)
  on.exit(close(targ))
  readLines(targ)
}
function (cmd, parmstring)
{
  targ <- url(paste(.SNPperBaseURL, cmd, parmstring, sep = ""))
  open(targ)
  on.exit(close(targ))
  readLines(targ)
}
```

```
> print(useSNPper("geneinfo", "&name=CRP")[1:7])
```

```
[1] " <SNPPER-RPC SOURCE=\"SNPper - IIPGA - http://snpper.chip.org/\" VERSION=\" $Revision: 1.1 \">"
[2] " <GENEINFO>"
[3] " <GENE ID=\"799\">"
[4] " <GENEID>799</GENEID>"
[5] " <NAME>CRP</NAME>"
[6] " <CHROM>chr1</CHROM>"
[7] " <STRAND>--</STRAND>"
```

The main functions of *RSNPper* attend to simplifying specification of parameters and parsing and packaging the XML results.

Note on auditability. All functions return textual information coupled with auditing information as a 'toolInfo' attribute, detailing the SNPper supplied information on the human genome sequence build, the dbSNP version, and the SNPper version from which the results are obtained. At present, there is one exception: when `itemsInRange` is invoked with `item='countsnps'`, no toolInfo data is obtained. This will be corrected once the `countsnps` command at SNPper returns valid XML element tags.

3 Overview of the functions

The current set of functions intended for investigative use is:

- `geneInfo` – general information about location and nomenclature
- `geneLayout` – information about exon locations
- `geneSNPs` – all SNPs associated with a given gene
- `SNPinfo` – detailed information on a SNP
- `itemsInRange` – supports chromosome scanning for genes, SNPs, or counts of SNPs

An omission: for SNP information, I have not collected information on submitter.

4 Demonstrations

4.1 Obtaining information on genes

The `geneInfo` function will collect some basic information on a gene. The gene may be specified by HUGO name, mRNA accession number, or SNPper id.

```
> print(geneInfo("CRP"))
```

snpper.ID	NAME
"799"	"CRP"
CHROM	STRAND
"chr1"	"-"
PRODUCT	LOCUSLINK
"C-reactive protein, pentraxin-related"	"1401"
OMIM	UNIGENE
"123260"	"Hs.76452"
SWISSPROT	NSNPS
"P02741"	"85"
REFSEQACC	MRNAACC
"NT_079484.1"	"NM_000567"
TRANSCRIPT.START	CODINGSEQ.START
"156899244"	"156900107"
TRANSCRIPT.END	CODINGSEQ.END
"156901156"	"156901067"

```
attr(,"toolInfo")
```

SOURCE
"SNPper - IIPGA - http://snpper.chip.org/ "
VERSION

```

"$Revision: 1.35 $"
GENOME
"hg16"
DBSNP
"118"

```

The `geneLayout` function provides information on exon locations.

```
> print(geneLayout("546"))
```

```

          ID          NAME          CHROM TRANSCRIPT.START
" " "CGI-100" "chr1" "93089211"
CODINGSEQ.START TRANSCRIPT.END CODINGSEQ.END exon1.start
"93092028" "93117712" "93117600" "93089211"
exon1.end exon2.start exon2.end exon3.start
"93092247" "93093658" "93093842" "93097487"
exon3.end exon4.start exon4.end
"93097585" "93117412" "93117713"
attr(,"toolInfo")
SOURCE
"SNPper - IIPGA - http://snpper.chip.org/"
VERSION
"$Revision: 1.35 $"
GENOME
"hg16"
DBSNP
"118"

```

Information on all the genes catalogued in a certain chromosomal region can be obtained using `itemsInRange`.

```
> print(itemsInRange("genes", "chr1", "156400000", "156500000"))
```

```
[[1]]
```

```

NAME
"FCER1A"
CHROM
"chr1"
PRODUCT
"Fc fragment of IgE, high affinity I, receptor for; alpha polypeptide precursor"
NSNPS
"50"

```

```
$CHR
```

```

[1] "chr1"

$START
[1] "156400000"

$END
[1] "156500000"

$COUNT
[1] "1"

attr("toolInfo")
SOURCE
"SNPper - IIPGA - http://snpper.chip.org/"
VERSION
"$Revision: 1.35 $"
GENOME
"hg16"
DBSNP
"118"

```

4.2 Obtaining information on SNPs

Suppose you want information on the SNP with dbSNP id rs25.

```
> print(SNPinfo("25"))
```

DBSNPID	TSCID	CHROMOSOME	POSITION	ALLELES	ROLE	RELPOS
"rs25"	" "	"chr7"	"11328479"	"A/G"	" "	" "
AMINO	AMINOPOS					
" "	" "					

```

attr("toolInfo")
SOURCE
"SNPper - IIPGA - http://snpper.chip.org/"
VERSION
"$Revision: 1.35 $"
GENOME
"hg16"
DBSNP
"118"

```

Suppose instead you want information on all the SNPs cataloged in a certain chromosomal region.

```
> ird <- itemsInRange("snps", "chr1", "156400000", "156500000")
> print(length(ird))
```

```
[1] 160
```

```
> print(ird[1:3])
```

```
[[1]]
```

DBSNPID	TSCID	CHROMOSOME	POSITION
"rs3027048"	"TSC0112402"	"chr1"	"156400505"
ALLELES	ROLE	RELPOS	AMINO
"C/T"	"UTR"	"8489"	" "
AMINOPOS	HUGO	LOCUSLINK	NAME
" "	"FY"	"2532"	"Duffy blood group"
MRNA			
"NM_002036"			

```
[[2]]
```

DBSNPID	TSCID	CHROMOSOME	POSITION
"rs3027049"	" "	"chr1"	"156401020"
ALLELES	ROLE	RELPOS	AMINO
"A/C"	"UTR"	"9004"	" "
AMINOPOS	HUGO	LOCUSLINK	NAME
" "	"FY"	"2532"	"Duffy blood group"
MRNA			
"NM_002036"			

```
[[3]]
```

DBSNPID	TSCID	CHROMOSOME	POSITION
"rs3027050"	" "	"chr1"	"156401029"
ALLELES	ROLE	RELPOS	AMINO
"C/G"	"UTR"	"9013"	" "
AMINOPOS	HUGO	LOCUSLINK	NAME
" "	"FY"	"2532"	"Duffy blood group"
MRNA			
"NM_002036"			

Note that the start and end locations are supplied as strings. This is to avoid coercion to textual scientific notation.

Additional detail on the count of SNPs can be obtained more briefly:

```
> print(itemsInRange("countsnps", "chr1", "156400000", "156500000"))
```

```
total exonic nonsyn
156      3      2
```

To see all the SNPs associated with a given gene, use the `geneSNPs` function. This requires knowledge of the SNPper gene id, which can be obtained using `geneInfo`.

```
> gs <- geneSNPs("546")
> print(length(gs))
```

```
[1] 64
```

```
> print(gs[1:3])
```

```
[[1]]
```

DBSNPID	TSCID	CHROMOSOME	POSITION
"rs6696026"	" "	"chr1"	"93079821"
ALLELES	ROLE	RELPOS	AMINO
"A/C"	"UTR"	"37779"	" "
AMINOPOS	HUGO	LOCUSLINK	NAME
" "	"CGI-100"	"50999"	"CGI-100 protein"
MRNA			
"NM_016040"			

```
[[2]]
```

DBSNPID	TSCID	CHROMOSOME	POSITION
"rs6670960"	" "	"chr1"	"93079828"
ALLELES	ROLE	RELPOS	AMINO
"A/G"	"UTR"	"37772"	" "
AMINOPOS	HUGO	LOCUSLINK	NAME
" "	"CGI-100"	"50999"	"CGI-100 protein"
MRNA			
"NM_016040"			

```
[[3]]
```

DBSNPID	TSCID	CHROMOSOME	POSITION
"rs1932313"	"TSC0995777"	"chr1"	"93080113"
ALLELES	ROLE	RELPOS	AMINO
"C/T"	"UTR"	"37487"	" "
AMINOPOS	HUGO	LOCUSLINK	NAME
" "	"CGI-100"	"50999"	"CGI-100 protein"
MRNA			
"NM_016040"			

5 Application: SNP density on chr 1

Human chromosome 1 is approximately 300Mb, and 142,629 SNPs have been recorded as of dbSNP build 106, according to NCBI SNP/maplists/maplist-newmap.html on 13 Sep 03. Let's see if these facilities can recover this sort of information. Counting the number of SNPs on a long chromosomal region seems to take a long time for SNPper, so we will break up the task.

```
> print(itemsInRange("countsnp", "chr1", "1", "100000"))

total exonic nonsyn
  171      0      0

> system("sleep 2")
> print(itemsInRange("countsnp", "chr1", "100001", "200000"))

total exonic nonsyn
   0      0      0

> system("sleep 2")
> print(itemsInRange("countsnp", "chr1", "200001", "300000"))

total exonic nonsyn
  13      0      0

> system("sleep 2")
```

These runs complete in a reasonable amount of time. Here we will just look at the first 2Mb in intervals of .1Mb.

```
> starts <- as.character(as.integer(seq(1, 2000001, 1e+05)))
> ends <- as.character(as.integer(as.integer(starts) + 99999))
> out <- matrix(NA, nr = 20, nc = 3)
> for (i in 1:20) {
+   cat(i)
+   out[i, ] <- itemsInRange("countsnp", "chr1", starts[i],
+     ends[i])
+   system("sleep 2")
+ }

1234567891011121314151617181920

> print(out)
```


	[,1]	[,2]	[,3]
[1,]	171	0	0
[2,]	0	0	0
[3,]	13	0	0
[4,]	2	0	0
[5,]	2	0	0
[6,]	50	0	0
[7,]	152	0	0
[8,]	241	0	0
[9,]	183	8	1
[10,]	125	4	0
[11,]	196	27	9
[12,]	185	31	11
[13,]	87	7	3
[14,]	262	27	7
[15,]	154	11	2
[16,]	69	17	8
[17,]	169	50	1
[18,]	78	0	0
[19,]	42	0	0
[20,]	236	11	3