Using the FunciSNP package 'FunciSNP: An R/Bioconductor Tool Integrating Functional Non-coding Datasets with Genetic Association Studies to Identify Candidate Regulatory SNPs'

Simon G. Coetzee[°‡*], Suhn K. Rhie[‡], Benjamin P. Berman[‡], Gerhard A. Coetzee[‡] and Houtan Noushmehr[°‡†]

April 30, 2018

[°]Faculdade de Medicina de Ribeirão Preto
Departmento de Genética
Universidade de São Paulo
Ribeirão Preto, São Paulo, BRASIL

–

[‡]Norris Cancer Center
Keck School of Medicine
University of Southern California
Los Angeles, CA, USA

Contents

[*]scoetzee NEAR gmail POINT com
[†]houtan NEAR usp POINT br

1 Introduction

FunciSNP assist in identifying putative functional SNP in LD to previously annotated GWAS SNPs (tagSNP). Extracting information from the 1000 genomes database (1kg) by relative genomic position of GWAS tagSNP currated for a particular trait or disease, FunciSNP aims to integrate the two information with sequence information provided by peaks identified from high-throughput sequencing. FunciSNP assumes user will provide peaks identified using any available ChIP peak algorithm, such as FindPeaks, HOMER, or SICER. *FunciSNP* will currate all 1kg SNPs which are in linkage disequilibrium (LD) to a known disease associated tagSNP and more importantly determine if the 1kg SNP in LD to the tagSNP overlaps a genomic biological feature.

Correlated SNPs are directly imported from the current public release of the 1000 genomes database. 1000 genomes ftp servers available for the 1000 genomes public data:

- National Center for Biotechnology Information (NCBI)[1]
- European Bioinformatics Institute (EBI)[2]

[1] `ftp://ftp-trace.ncbi.nih.gov/1000genomes/`
[2] `ftp://ftp.1000genomes.ebi.ac.uk/vol1/`

Correlated SNPs in LD to a tagSNP and overlapping genomic biological features are known as putative functional SNPs.

This vignette provides a 'HOW-TO' guide in setting up and running *FunciSNP* on your machine. FunciSNP was developed with the idea that a user will have uninterupted high-speed internet access as well as a desktop machine with more than 4 multiple cores. If user is using a windows machine, multiple cores options will not work and thus total time to complete initial FunciSNP analysis will take longer than expected. Be sure you have uninterupted computing power when using a windows machine. If using a linux machine, please use 'screen' (see 'man screen' for more information).

1.1 Benchmark

Using a 64bit Linux machine running 11.04 Ubuntu OS with 24G RAM and 8 cores connected to a academic high-speed internet port, the amount of time to complete 99 tagSNP across 20 different biofeatures took less than 30 min to complete. We anticipate about 2 hours to complete the same analysis using one core.

1.2 Genome-Wide Association Studies SNP (GWAS SNP)

Genome-wide association studies (GWASs) have yielded numerous single nucleotide polymorphisms (SNPs) associated with many phenotypes. In some cases tens of SNPs, called tagSNPs, mark many loci of single complex diseases such as prostate (> 50 loci), breast (> 20 loci), ovarian (>10 loci), colorectal (>20 loci) and brain cancer (>5 loci) for which functionality remains unknown. Since most of the tagSNPs (>80%) are found in non-protein coding regions, finding direct information on the functional and/or causal variant has been an important limitation of GWAS data interpretation.

1.3 1000 genomes project (1kg)

The 1000 genomes project recently released a catalog of most human genomic variants (minor allele frequency of >0.1%) across many different ethnic populations. Initially, the 1000 genomes project goal was to sequence up to 1000 individuals, but has since sequenced more than 2000 individuals, thereby increasing our current knowledge of known genomic variations which currently sits at just over 50 million SNPs genome wide (approx. 2% of the entire genome and on average 1 SNP every 60 base pairs)

1.4 Genomic features (Biofeatures)

With the advent of advanced sequencing technologies (next-generation sequencing, NGS), genomic regulatory areas in non-coding regions have been well characterized and annotated. Coupled with chromatin immuno-precipitation for a protein (e.g. transcription factor of histone) of interest, also known as ChIPseq, the technology have provided us with a unique view of the genomic landscape, thereby providing a wealth of new knowledge for genomics

research. Work by large consortia groups such as the Encyclopedia of DNA Elements (ENCODE), the Roadmap Epigenomics Mapping Consortium and The Cancer Genome Atlas (TCGA), have made publicly available a growing catalog of many different histone marks, transcription factors and genome-wide sequencing datasets for a variety of different diseases and cell lines, including well characterized cancer cell lines such as MCF7 (breast cancer), HCT116 (colon cancer), U87 (brain cancer) and LNCaP (prostate cancer).

2 Installing and Loading FunciSNP

To obtain a copy of *FunciSNP*, you will need to install BiocInstaller via Bioconductor:
library(BiocInstaller); biocLite("FunciSNP");

If you download the source code from the bioconductor page which lists FunciSNP, you can install *FunciSNP* by following the instructions described in R CRAN. By installing *FunciSNP* from source, the package assumes you have all the required libraries installed.

- Rsamtools (>= 1.6.1)
- rtracklayer(>= 1.14.1)
- GGtools (>= 4.0.0)
- methods
- ChIPpeakAnno (>= 2.2.0)
- GenomicRanges
- TxDb.Hsapiens.UCSC.hg19.knownGene
- VariantAnnotation
- plyr
- org.Hs.eg.db
- snpStats

The following loads the *FunciSNP* library in R.

```
> options(width=80);
> library(FunciSNP);
> package.version("FunciSNP");

[1] "1.24.0"
```

3 Running getFSNPs to identify putative functional SNPs

Before running *getFSNPs*, two input files are required. A list of tagSNPs and a folder with all available biological features (peak files in BED format).

3.1 Create a GWAS SNP file

GWAS SNPs (tagSNP) should be listed in a tab or whitespace separated file. Three columns are required for each tagSNP:

- Position (chrom:position)
- rsID (rsXXXXXXXX)
- population (EUR, AFR, AMR, ASN, or ALL)

'Positon' should be the exact postion for each rsID as reported by human genome build hg19 (chrom:postion). 'rsID' should contain a unique rsID as determined by the 1000 genomes database (1kg)[3] for each identified 'tagSNP'. Population should be a three letter code to determine original ethnic population for which the associated 'tagSNP' was identified. The three letter code should be either European (EUR), Asian (ASN), African (AFR), American (AMR), or All (ALL). List each tagSNP per ethnic population. If similar rsID was identified in multiple ethnic population, list each duplicate tagSNP separately with the appropriate ethnic pouplation.

Several GWAS SNPs significantly associated with Glioblastoma multiforme (GBM)[4] were collected for this example. GBM is a brain cancer with median survival at less than 12 months, making this form of cancer one of the most aggressive of all cancer types. Currently, there is no known function of any of these associated tagSNPs. In this example, GBM includes lower grade glioma, therefore we use the 'glioma' to label all objects.

```
> ## Full path to the example GWAS SNP regions file for Glioblastoma
> #  (collected from SNPedia on Jan 2012)
> glioma.snp <- file.path(system.file('extdata', package='FunciSNP'),
+ dir(system.file('extdata',package='FunciSNP'), pattern='.snp$'));
> gsnp <- read.delim(file=glioma.snp,sep=" ",header=FALSE);
> gsnp;
```

```
           V1        V2  V3
1 11:118477367  rs498872 EUR
2    5:1286516 rs2736100 ASN
3   9:22068652 rs4977756 EUR
4  20:62309839 rs6010620 EUR
```

[3]Be sure the rsID is located in this browser: `http://browser.1000genomes.org/`

[4]See `http://www.snpedia.com/index.php/Glioma`

Now, `glioma.snp` contains the full path to the GWAS tagSNP.

3.2 Biofeatures in BED format

Each biofeature used to identify correlated SNP should be in standard BED format[5]. Each biofeature should be stored in one folder and should have file extension '*.bed'.

Here is an example of three different biofeatures used for this glioma example. NRSF and PolII (both transcription factors) where extracted from a recent release of ENCODE, as well as promoters of approximately 38,000 gene transcription start sites (TSS). Promoters are identified as +1000 to -100 base pair of each annotated TSS. In addition, we include all known DNAseI sites as supplied by ENCODE as well as FAIRE data. In additoin, we used known CTCF sites to differentiate the DNAseI. The DNAseI and FAIRE data were extracted in April of 2012 and they represent the best known regions across several different cell lines. In addition, for the FAIRE data, we selected peaks with p-values less than 0.01.

```
> ## Full path to the example biological features BED files
> #  derived from the ENCODE project for Glioblastoma U-87 cell lines.
> glioma.bio <- system.file('extdata',package='FunciSNP');
> #user supplied biofeatures
> as.matrix(list.files(glioma.bio, pattern='.bed$'));

     [,1]
[1,] "TFBS_Nrsf_U87.bed"
[2,] "TFBS_Pol2_U87.bed"

> #FunciSNP builtin biofeatures
> as.matrix(list.files(paste(glioma.bio, "/builtInFeatures", sep=""),
+             pattern='.bed$'));

     [,1]

> nrsf.filename <- list.files(glioma.bio, pattern='.bed$')[1];
> pol2.filename <- list.files(glioma.bio, pattern='.bed$')[2];
> Ctcf <- ctcf_only
> Dnase1 <- encode_dnase1_only
> Dnase1Ctcf <- encode_dnase1_with_ctcf
> Faire <- encode_faire
> Promoters <- known_gene_promoters
> Nrsf <- read.delim(file=paste(glioma.bio, nrsf.filename,sep="/"), sep="\t",
+ header=FALSE);
> PolII <- read.delim(file=paste(glioma.bio, pol2.filename,sep="/"), sep="\t",
+ header=FALSE);
> dim(Nrsf);
```

[5]See UCSC FAQ: `http://genome.ucsc.edu/FAQ/FAQformat`

```
[1] 1264    6

> dim(PolII);

[1] 10918     6

> dim(Ctcf);

NULL

> dim(Dnase1);

NULL

> dim(Dnase1Ctcf);

NULL

> dim(Faire);

NULL

> dim(Promoters);

NULL

> ## Example of what the BED format looks like:
> head(Nrsf);

    V1        V2        V3                       V4 V5 V6
1 chr5 178601706 178602140 Merged-chr5-178601923-1  0  +
2 chr5 178850156 178850592 Merged-chr5-178850374-1  0  +
3 chr5 179015119 179015553 Merged-chr5-179015336-1  0  +
4 chr7     23844     24636     Merged-chr7-24240-1  0  +
5 chr7     65601     66065     Merged-chr7-65833-1  0  +
6 chr7    128907    129421    Merged-chr7-129164-1  0  +
```

As an example, `Nrsf` was created to illustrate the format needed for each biofeatures. To run getFSNPs, only the path to the folder to each biofeature is required (`glioma.bio`).

3.3 getFSNPs analysis using two inputs

To run the example data could take more than 5 minutes, thus the R code is commented out for this tutorial. If you are interested in running the glioma example from scratch, please uncomment the following and rerun in your R session. NOTE: The main method to run FunciSNP is *getFSNPs*.

```
> ## FunciSNP analysis, extracts correlated SNPs from the
> ## 1000 genomes db ("ncbi" or "ebi") and finds overlaps between
> ## correlated SNP and biological features and then
> ## calculates LD (Rsquare, Dprime, distance, p-value).
> ## Depending on number of CPUs and internet connection, this step may take
> ## some time. Please consider using a unix machine to access multiple cores.
>
> # glioma <- getFSNPs(snp.regions.file=glioma.snp, bio.features.loc = glioma.bio,
> # bio.features.TSS=FALSE);
```

As an alternative, `glioma` was pre-run and stored in the package as an *R* object. To call this data object, simily run the following commands.

```
> data(glioma);
> class(glioma);

[1] "TSList"
attr(,"package")
[1] "FunciSNP"
```

Now, `glioma` contains the R data structure that holds all the results for this particular analysis. Each tagSNP is stored as a slot which contains associated correlated SNP and overlapping biofeature. It also contains a number of different annotations (see below for more details). To see a brief summary of the results (*summary*), type the following commands:

```
> glioma;

TagSNP List with  4  Tag SNPs and
 1855 nearby,  potentially correlated SNPs, that overlap at least one biofeature
$`R squared: 0.1`
            Total R.sq>=0.1 Percent
tagSNPs         4         4  100.00
1K SNPs      1855       131    7.06
Biofeatures     6         6  100.00

$`R squared: 0.5`
            Total R.sq>=0.5 Percent
```

```
tagSNPs          4          3   75.00
1K SNPs       1855         64    3.45
Biofeatures      6          4   66.67

$`R squared: 0.9`
             Total R.sq>=0.9 Percent
tagSNPs          4          1    25.0
1K SNPs       1855         13     0.7
Biofeatures      6          3    50.0
```

As you can quickly observe from the above analysis, using 4 tagSNPs position and 7 different biological features (ChIPseq for 'NRSF', 'PolII', promoters of approx. 38,000 genes, DNAseI sites, DNAseI sites with CTCFs, FAIRE, CTCFs) as two types of input, FunciSNP identified 1809 1kg SNPs that overlap at least one biofeature. Each 1kg SNP contains an Rsquare value to the associated tagSNP. As a result, the first output (`glioma`), summarizes the analysis subsetted in three different Rsquare values (0.1, 0.5 and 0.9). If we consider Rsquare cutoff at 0.9 (Rsquare ≥ 0.9), 14 1kg SNPs overlapping at least one biofeature. This value represents 0.77% of the total (1809). In addition, at this Rsquare cutoff, 3 biological features are represented among the 14 1kg SNPs.

```
> summary(glioma);

TagSNP List with  4  Tag SNPs and
 1855 nearby,  potentially correlated SNPs, that overlap at least one biofeature
Number of potentially correlated SNPs
overlapping at least x biofeatures, per Tag SNP at a specified R squared
$`R squared: 0.1 in 4 Tag SNPs with a total of `
               bio.1 bio.2 bio.3 bio.4
rs2736100          2     0     0     0
rs4977756         17     4     0     0
rs498872          29     9     0     0
rs6010620         83    20     9     4
TOTAL # 1kgSNPs  131    33     9     4

$`R squared: 0.5 in 3 Tag SNPs with a total of `
               bio.1 bio.2 bio.3 bio.4
rs4977756          7     3     0     0
rs498872           6     2     0     0
rs6010620         51    10     5     3
TOTAL # 1kgSNPs   64    15     5     3

$`R squared: 0.9 in 1 Tag SNPs with a total of `
               bio.1 bio.2
```

```
rs6010620          13     2
TOTAL # 1kgSNPs    13     2
```

Running *summary* however will output a slightly different report yet just as informative. At three different Rsquare cutoffs (0.1, 0.5, 0.9), the summary output illustrates the tagSNP with the total number of 1kg SNPs overlapping a total number of biofeatures. For example, at Rsquare ≥ 0.5, tagSNP 'rs6010620' is assocated with 53 different 1kg SNPs which overlap at least one biofeature, and 11 of them overlap at least two biofeatures.

Each newly identified 1kg SNP is now defined as putative functional SNP since they are in LD to an associated tagSNP and they overlap at least one interesting biological feature. Thus, each 1kg SNP can now be defined as '**YAFSNP**' or 'putative functional SNP.'

4 Annotating newly identified putative functional SNPs

All known genomic features (exon, intron, 5'UTR, 3'UTR, promoter, lincRNA or in gene desert (intergentic)) are used to annotate each newly identified YAFSNP as described above. Information stored in this `glioma.anno` is used for all summary plots, table, and to output results in BED format (see following sections for more details). The following step will output the data.frame.

```
> glioma.anno <- FunciSNPAnnotateSummary(glioma);
> class(glioma.anno);

[1] "data.frame"

> gl.anno <- glioma.anno;
> ## remove rownames for this example section.
> rownames(gl.anno) <- c(1:length(rownames(gl.anno)))
> dim(gl.anno);

[1] 2470   28

> names(gl.anno);

 [1] "chromosome"                        "bio.feature.start"
 [3] "bio.feature.end"                   "bio.feature"
 [5] "corr.snp.id"                       "corr.snp.position"
 [7] "tag.snp.id"                        "tag.snp.position"
 [9] "D.prime"                           "R.squared"
[11] "p.value"                           "distance.from.tag"
[13] "population.count"                  "population"
[15] "nearest.lincRNA.ID"                "nearest.lincRNA.distancetoFeature"
[17] "nearest.lincRNA.coverage"          "nearest.TSS.refseq"
```

```
[19] "nearest.TSS.GeneSymbol"            "nearest.TSS.ensembl"
[21] "nearest.TSS.coverage"              "nearest.TSS.distancetoFeature"
[23] "Promoter"                          "utr5"
[25] "Exon"                              "Intron"
[27] "utr3"                              "Intergenic"
```

```
> head(gl.anno[, c(1:18,20:28)]);
```

```
  chromosome bio.feature.start bio.feature.end              bio.feature
1          5           1200710         1201809 knownGene.Promoters.known
2          5           1211569         1212494         EncodeFaire.known
3          5           1211569         1212494         EncodeFaire.known
4          5           1211569         1212494         EncodeFaire.known
5          5           1211569         1212494         EncodeFaire.known
6          5           1221979         1222994         EncodeFaire.known
   corr.snp.id corr.snp.position tag.snp.id tag.snp.position D.prime
1 chr5:1201778           1201778  rs2736100          1286516      NA
2 chr5:1212406           1212406  rs2736100          1286516       1
3 chr5:1212431           1212431  rs2736100          1286516      NA
4 chr5:1212445           1212445  rs2736100          1286516       1
5 chr5:1212490           1212490  rs2736100          1286516      NA
6 chr5:1221997           1221997  rs2736100          1286516       1
    R.squared p.value distance.from.tag population.count population
1          NA       1            -84738              286        ASN
2 0.002275140       1            -74110              286        ASN
3          NA       1            -74085              286        ASN
4 0.002275140       1            -74071              286        ASN
5          NA       1            -74026              286        ASN
6 0.002700915       1            -64519              286        ASN
  nearest.lincRNA.ID nearest.lincRNA.distancetoFeature nearest.lincRNA.coverage
1     TCONS_00010241                            -40360                upstream
2     TCONS_00010241                            -50988                upstream
3     TCONS_00010241                            -51013                upstream
4     TCONS_00010241                            -51027                upstream
5     TCONS_00010241                            -51072                upstream
6     TCONS_00010241                            -60579                upstream
  nearest.TSS.refseq nearest.TSS.ensembl nearest.TSS.coverage
1       NM_001003841     ENST00000304460               inside
2       NM_001003841     ENST00000304460               inside
3       NM_001003841     ENST00000304460               inside
4       NM_001003841     ENST00000304460               inside
5       NM_001003841     ENST00000304460               inside
6          NM_182632     ENST00000324642             upstream
```

```
  nearest.TSS.distancetoFeature Promoter utr5 Exon Intron utr3 Intergenic
1                            69      YES   NO  YES     NO   NO         NO
2                         10697       NO   NO   NO    YES   NO         NO
3                         10722       NO   NO  YES     NO   NO         NO
4                         10736       NO   NO  YES     NO   NO         NO
5                         10781       NO   NO  YES     NO   NO         NO
6                         -3472       NO   NO  YES     NO   NO         NO
```

```
> summary(gl.anno[, c(1:18,20:28)]);
```

```
  chromosome        bio.feature.start   bio.feature.end
 Length:2470        Min.   :  1190775   Min.   :  1191397
 Class :character   1st Qu.: 22102754   1st Qu.: 22103702
 Mode  :character   Median : 62315624   Median : 62324005
                    Mean   : 56309305   Mean   : 56311568
                    3rd Qu.: 62365185   3rd Qu.: 62366592
                    Max.   :118574381   Max.   :118574590

                      bio.feature        corr.snp.id   corr.snp.position
 EncodeDnaseI_only.known    : 463   rs1291208  :   4   Min.   :  1190800
 EncodeDnaseI_withCTCF.known:  10   rs1291209  :   4   1st Qu.: 22102896
 EncodeFaire.known          :1016   rs1295810  :   4   Median : 62317710
 knownGene.Promoters.known  : 420   rs143566670:   4   Mean   : 56310408
 TFBS_Nrsf_U87              :  13   rs183316902:   4   3rd Qu.: 62365808
 TFBS_Pol2_U87              : 548   rs186971726:   4   Max.   :118574557
                                    (Other)    :2446
    tag.snp.id   tag.snp.position      D.prime         R.squared
 rs2736100: 365   Min.   :  1286516   Min.   :0.0037   Min.   :0.0000
 rs4977756: 418   1st Qu.: 22068652   1st Qu.:0.8934   1st Qu.:0.0010
 rs498872 : 432   Median : 62309839   Median :1.0000   Median :0.0045
 rs6010620:1255   Mean   : 56305808   Mean   :0.8594   Mean   :0.0933
                  3rd Qu.: 62309839   3rd Qu.:1.0000   3rd Qu.:0.0259
                  Max.   :118477367   Max.   :1.0000   Max.   :0.9776
                                      NA's   :1427     NA's   :1427
   p.value       distance.from.tag population.count population
 Min.   :0.0000   Min.   :-100000   Min.   :286.0    ASN: 365
 1st Qu.:1.0000   1st Qu.: -31474   1st Qu.:379.0    EUR:2105
 Median :1.0000   Median :  13384   Median :379.0
 Mean   :0.8198   Mean   :   4600   Mean   :365.3
 3rd Qu.:1.0000   3rd Qu.:  30547   3rd Qu.:379.0
 Max.   :1.0000   Max.   :  99950   Max.   :379.0

     nearest.lincRNA.ID nearest.lincRNA.distancetoFeature
```

```
TCONS_00010241: 365     Min.   :-266862
TCONS_00015797: 418     1st Qu.: -85894
TCONS_00020001: 432     Median :  57458
TCONS_00027984:  88     Mean   :   9749
TCONS_00028269:1167     3rd Qu.:  79664
                        Max.   : 246019

nearest.lincRNA.coverage    nearest.TSS.refseq      nearest.TSS.ensembl
downstream:1564          NM_003823   :600      ENST00000480273:600
inside    :  21          NM_004936   :366      ENST00000276925:366
upstream  : 885          NM_001144758:232      ENST00000361465:232
                         NM_020062   :129      ENST00000266077:129
                         NM_017806   :112      ENST00000486025:112
                         NM_007180   :106      ENST00000264029:106
                         (Other)     :925      (Other)        :925
nearest.TSS.coverage nearest.TSS.distancetoFeature Promoter    utr5
downstream: 195      Min.   :-156104.0            NO :2205   NO :2296
inside    : 865      1st Qu.:  -9952.0            YES: 265   YES: 174
upstream  :1410      Median :   -709.5
                     Mean   : -10244.2
                     3rd Qu.:   4161.0
                     Max.   :  30391.0

 Exon      Intron      utr3      Intergenic
NO :2219   NO : 842   NO :2189   NO :2128
YES: 251   YES:1628   YES: 281   YES: 342
```

```
> rm(gl.anno);
```

As you can see, each tagSNP ('tag.snp.id') is associated with an identifiable YAFSNP ('corr.snp.id') and each are associated with a biological feature ('bio.feature'). Additional columns are included which assist in summarizing the final results.

Now, if you prefer, you can use several functions to help summarize and plot the final analysis or you can use your own set of scripts to further summarize the results. Either case, the final results are stored in `glioma.anno`.

5 Summarize FunciSNP results

The following sections describe methods to summarize and plot the newly identified YAFSNPs.

5.1 Summary table used to describe newly identified YAFSNPs

Using a specified Rsquare value (0-1) to subset the data, a table is generated which summarizes the total number of YAFSNPs, associated tagSNPs, and number of overlapping biofeatures. This will provide user a first look at the total number of available YAFSNP at a particular Rsquare cutoff.

The output is very similar to the output generated by calling `glioma`. But instead of getting a summary report three distinct Rsquare cutoffs, you can now specify the Rsquare cutoffs. In this case, we used rsq = 0.44 (to get a more objective rsq value, see figure 1 on page 17.

```
> FunciSNPtable(glioma.anno, rsq=0.44);
```

```
             Total R.sq>=0.44 Percent
tagSNPs          4          4  100.00
1K SNPs       1855         72    3.88
Biofeatures      6          5   83.33
```

If 'geneSum' argument is set to 'TRUE', a list of gene names is reported instead which informs on the nearest gene symbols to the set of YAFSNPs. Only unique gene symbols are reported since multiple distinct YAFSNP can be near the same gene.

```
> FunciSNPtable(glioma.anno, rsq=0.44, geneSum=TRUE);
```

```
   Gene_Names
1       ARFRP1
2       CDKN2B
3        LIME1
4       PHLDB1
5        RTEL1
6     SLC2A4RG
7        STMN3
8         TERT
9     TNFRSF6B
10        TREH
```

5.2 Summary of correlated SNPs overlapping biofeatures

FunciSNPsummaryOverlaps function helps to determine the total number of YAFSNPs overlapping a number of different biofeatures. This is similar to running *summary* on `glioma` above, except now you can specifically call the function and set a pre-determined 'rsq' value to subset the data and thereby obtain a more objective and informative result.

```
> FunciSNPsummaryOverlaps(glioma.anno)
```

```
                bio.1 bio.2 bio.3 bio.4
rs2736100         128    10     1     0
rs4977756         112    43     1     0
rs498872          124    34     4     0
rs6010620         431   115    31     9
TOTAL # 1kgSNPs   795   202    37     9
```

Using a 'rsq' value, the output is subsetted to summarize the results with Rsquare values $\geq$ 'rsq'.

```
> FunciSNPsummaryOverlaps(glioma.anno, rsq=0.44)
```

```
                bio.1 bio.2 bio.3 bio.4
rs2736100           1     0     0     0
rs4977756           9     3     0     0
rs498872            7     3     0     0
rs6010620          55    13     7     3
TOTAL # 1kgSNPs    72    19     7     3
```

5.3 Summary of correlated SNPs for a number of different tagSNPs

After running *FunciSNPsummaryOverlaps*, the next question one would like to know is which correlated SNPs overlapping a number of different biofeatures for a number of associated tagSNP. Thus, in the example above, we have determined that we are interested in learning more about the YAFSNPs associated with 'rs6010620' and which overlap at least 3 different biofeatures.

```
> rs6010620 <- FunciSNPidsFromSummary(glioma.anno, tagsnpid="rs6010620",
+ num.features=2, rsq=0.44);
> #summary(rs6010620);
> dim(rs6010620);

[1] 36 28

> class(rs6010620);

[1] "data.frame"

> ## See FunciSNPbed to visualize this data in a genome browser.
```

6 Plot FunciSNP results

6.1 Default plot

FunciSNPplot is a function developed to plot various types of plots to summarize and assist end-user in making informed discoveries of FunciSNP results. Plots can be stored in a folder for future reference. Most plots were created in with the idea that they can be directly outputted in presentations or publication formats.

The following example plots the distribution of the Rsquare values for each YAFSNP (Figure 1, page 17). We recommend attempting this plot before subsetting any data by a specified rsq value. The distribution helps to identify a specific Rsquare value that will provide the most informative discovery.

```
> pdf("glioma_dist.pdf")
> FunciSNPplot(glioma.anno)
> dev.off()
```

```
null device
          1
```

Figure 1 (page 17) illustrates the total number of YAFSNPsbinned at different Rsquare cutoffs. As you can see in this figure (1, page 17), there are a total of 13 YAFSNP with an Rsquare ≥ 0.9. Since this plot does not take into consideration unique YAFSNP the number may represent duplicate YAFSNP since they may overlap more than one biological feature.

6.2 Split by tagSNP

Using 'splitbysnp' argument, the same type of plot as above (Figure 1, page 17) is generated, however the total number of YAFSNPs are now divided by the associated tagSNP (Figure 2, page 18). It should be clear from this plot that 3 of the 4 tagSNP have a number of YAFSNP with Rsquares ≥ 0.5. And one tagSNP contains many more YAFSNP ('rs6010620').

```
> FunciSNPplot(glioma.anno, splitbysnp=TRUE)
> ggsave("glioma_dist_bysnp.pdf")
```

6.3 Heatmap of 1kg SNPs by tagSNP vs Biofeature

Now, if you are interested in knowing which biofeature and associated tagSNP contains the most number of 1kg SNPs, run the following.

```
> pdf("glioma_heatmap.pdf")
> FunciSNPplot(glioma.anno, heatmap=TRUE, rsq = 0.1)
> dev.off()
```

```
pdf
  2
```

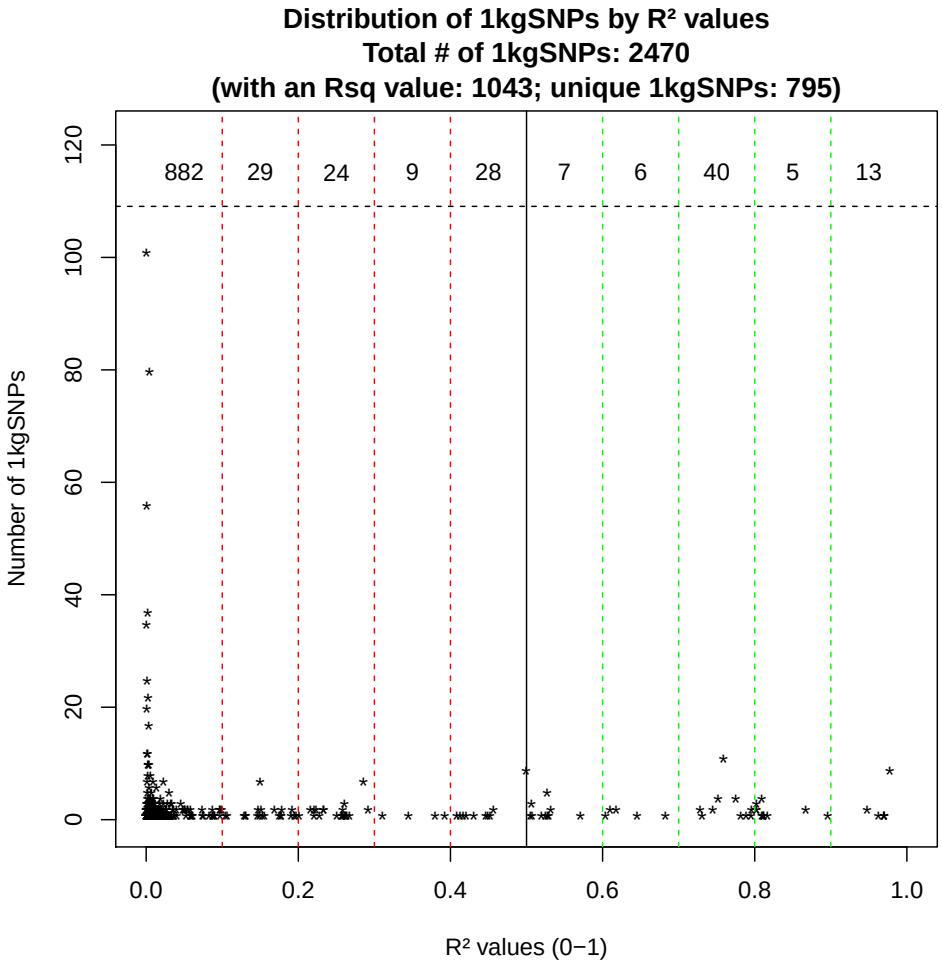

Figure 1: Distribution of Rsquare values of all YAFSNPs. Each marked bin contains the total number of YAFSNPs (correlated SNPs). The sum of all the counts would total the number of correlated SNPs.

6.4 TagSNP and Biofeature Summary

Using 'tagSummary' argument will automatically save all plots in a specific folder. This is done because this function will generate a summary plot for each biofeature. The first plot (Figure 4, page 20) is a scatter plot showing the relationship between Rsquare and Distance to tagSNP for each YAFSNP.

```
> ## Following will output a series of plots for each biofeature at rsq=0.5
> FunciSNPplot(glioma.anno, tagSummary=TRUE, rsq=0.5)

Finished plotting  1 / 6
Finished plotting  2 / 6
```

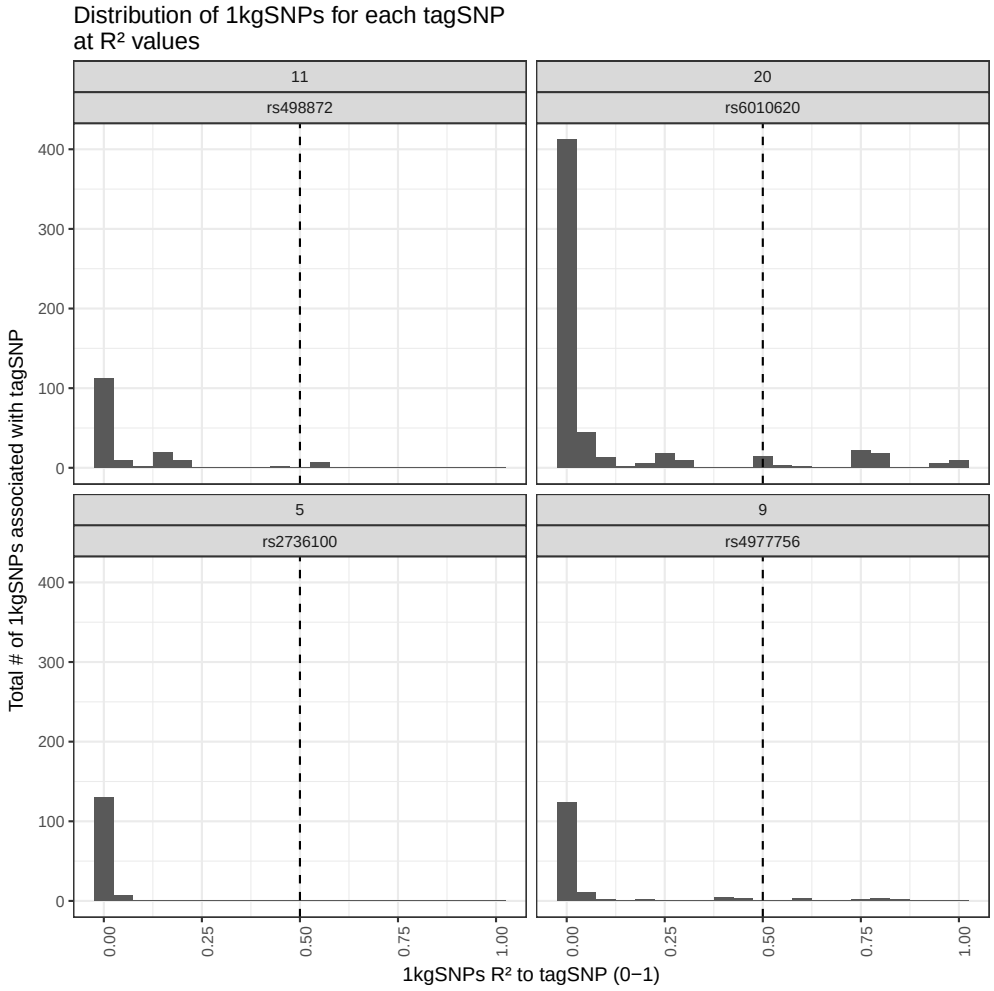

Figure 2: Distribution of Rsquare values of all YAFSNPs divided by the tagSNP and by its genomic location.

```
Finished plotting  3 / 6
Finished plotting  4 / 6
Finished plotting  5 / 6
Finished plotting  6 / 6
```

Figure 4 on page 20 helps identify the relative postion of all newly identified YAFSNP to the associated tagSNP. As highlighted in figure 4, it is clear that tagSNP 'rs6010620' contains many more YAFSNP with Rsquares ≥ 0.5, and the majority of them are within 40,000 base pairs of the tagSNP. There are a few YAFSNP which are more than 50,000 base pairs away while some are within 5,000 base pairs.

The second plot (Figure 5, page 21) is a histogram distribution of total number of YAFSNPs at each Rsquare value. This plot is similar to Figure 2 on page 18, except it is further

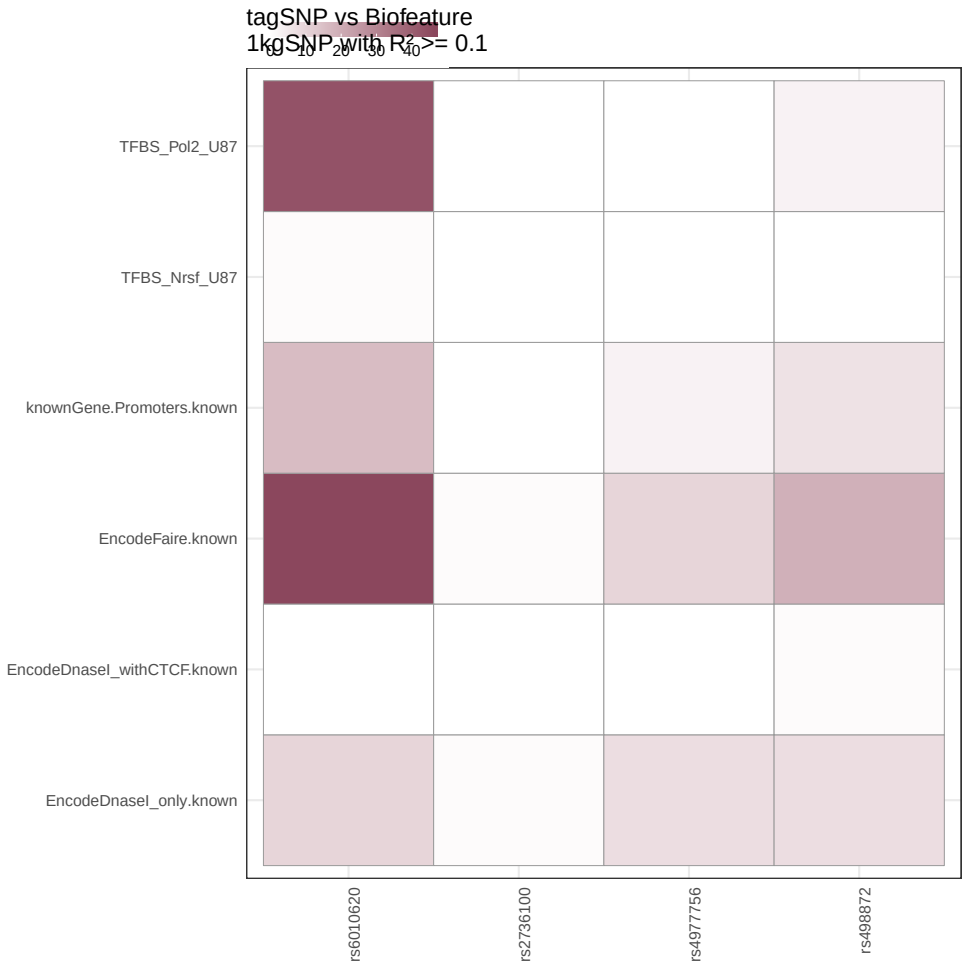

Figure 3: Heatmap of the number of 1kg SNPs by relationship between tagSNP and biofeature.

divided by biofeature. Each set of plot is further divided by tagSNP to help identify locus with the most identifiable YAFSNP. This argument is best used in conjunction with a 'rsq' value.

6.5 Genomic Feature Summary

Using 'genomicSum' argument set to 'TRUE' will output the overall genomic distribution of the newly identified YAFSNPs (Figure 6, page 22). Using 'rsq' value, the plot is divided into all YAFSNPs vs subset. This type of plot informs the relative enrichment for genomic features.

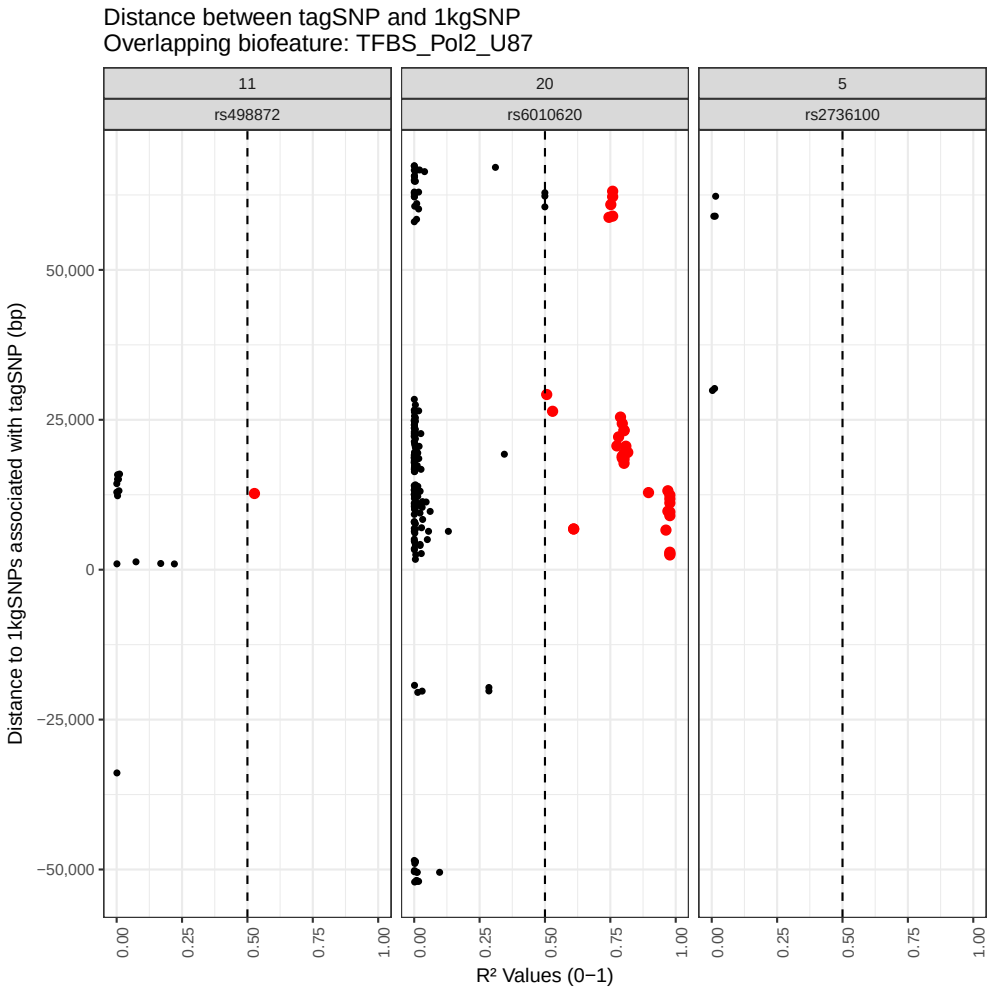

Figure 4: Scatter plot showing the relationship between Rsquare and Distance to tagSNP for each getFSNPs

```
> pdf("glioma_genomic_sum_rcut.pdf")
> FunciSNPplot(glioma.anno, rsq=0.5, genomicSum=TRUE, save=FALSE)
> dev.off()

pdf
  2
```

Figure 6 on page 22 illustrates the distribution of the YAFSNP by genomic features. It is clear by using an Rsquare cutoff of 0.5, there is a slight enrichment of YAFSNP in introns and exonds and a depletion at promoters and other coding regions as well as intergentic regions.

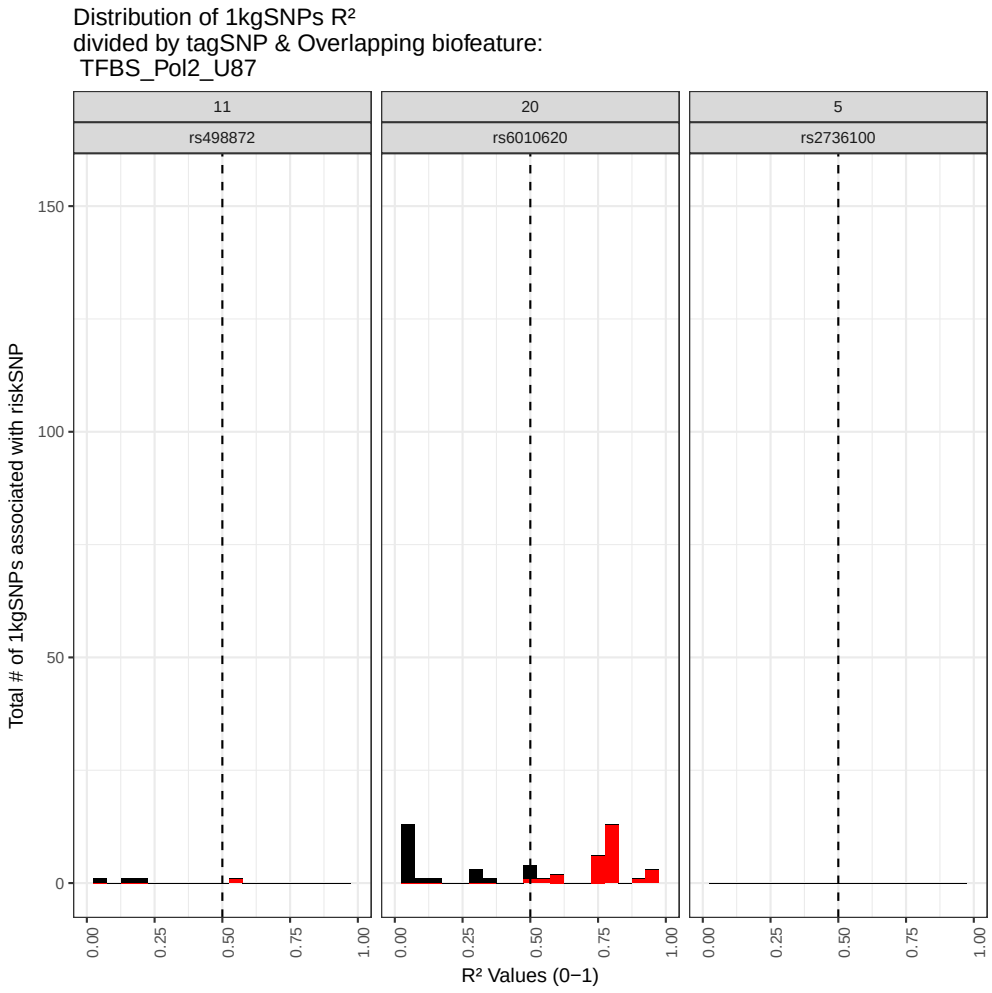

Figure 5: Histogram distribution of number of correlated SNPs at each Rsquare value

7 Visualize FunciSNP results in a genomic browser (outputs BED format)

Finally, after evaluating all results using the above tables and plots functions, a unique pattern emerges that helps identify a unique cluster of tagSNP and biofeature that can identify a set of YAFSNPs. To better visualize and to get a better perspective of the location of each newly identified YAFSNP, the results can be outputted using *FunciSNPbed*.

FunciSNPbed outputs a unique BED file which can be used to view in any genomic browser which supports BED formats. To learn more about BED formats, see UCSC Genome Browser FAQ (`http://genome.ucsc.edu/FAQ/FAQformat`).

```
> ## will output to current working directory.
> FunciSNPbed(glioma.anno, rsq=0.22);
```

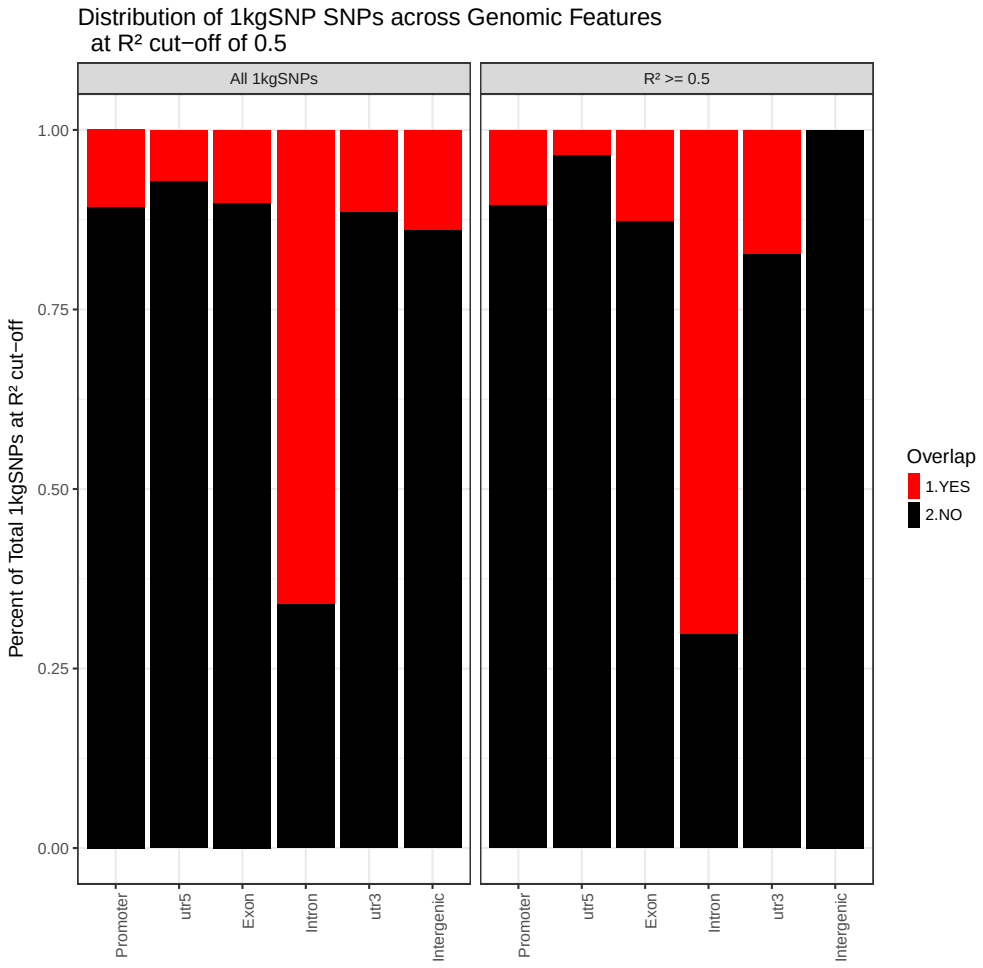

Figure 6: Stacked bar chart summarizing all correlated SNPs for each of the identified genomie features: exon, intron, 5UTR, 3UTR, promoter, lincRNA or in gene desert. Rsquare cutoff at 0.5. This plot is most informative if used with a rsq value.

```
Total corSNP (RED):  103
Total tagSNP (BLK):  4
```

```
> # FunciSNPbed(rs6010620, rsq=0.5);
```

Each tagSNP which is in LD to a corresponding YAFSNP overlapping at least one biofeature is colored black, while the YAFSNP is colored red. The initial position is provided by the first tagSNP and the first linked YAFSNP. We recommend using UCSC genome browser to view your BED files. This is useful so you can view all public and private tracks in relation to FunciSNP results. As an example, see Figure 7 on page 23 or visit this saved UCSC Genome Browser session: `http://goo.gl/xrZPD`.

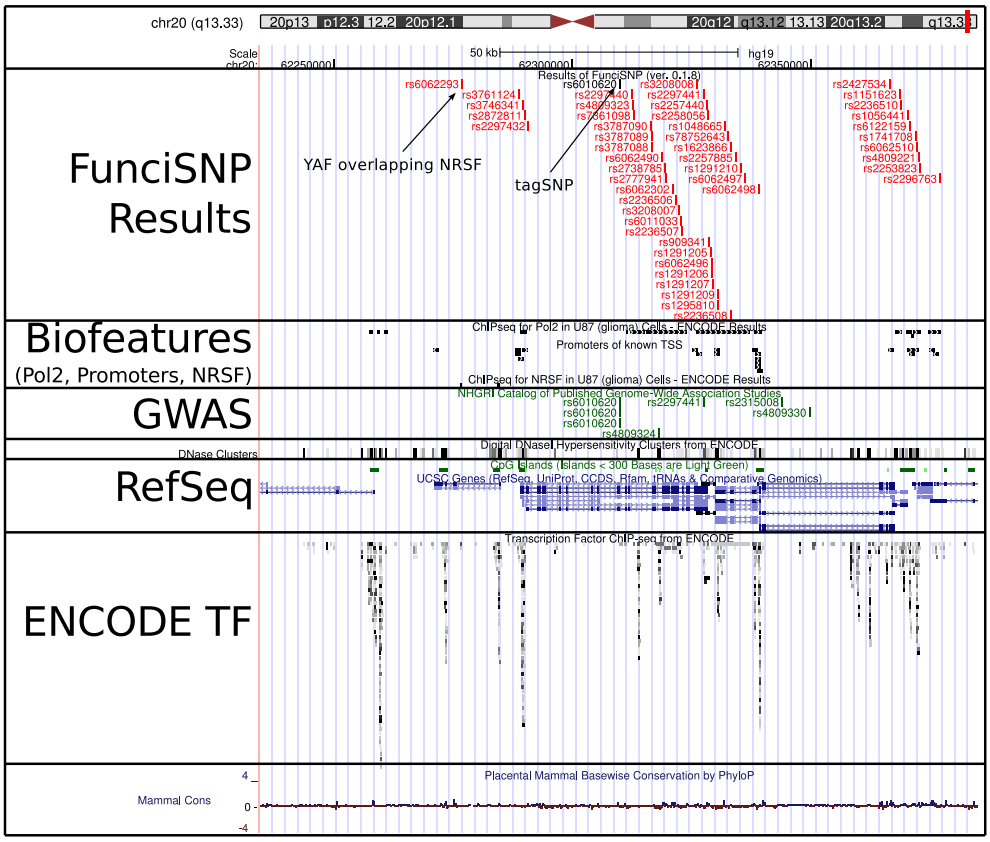

Figure 7: FunciSNP results viewed in UCSC genome browser. Top track represents FunciSNP results, second track is the known GWAS hits.

8 Contact information

Questions or comments, please contact Simon G. Coetzee (scoetzee NEAR gmail POINT com) or Houtan Noushmehr, PhD (houtan NEAR usp POINT br).

9 sessionInfo

- R version 3.5.0 (2018-04-23), `x86_64-pc-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Running under: `Ubuntu 16.04.4 LTS`

- Matrix products: default
- BLAS: `/home/biocbuild/bbs-3.7-bioc/R/lib/libRblas.so`
- LAPACK: `/home/biocbuild/bbs-3.7-bioc/R/lib/libRlapack.so`
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, stats4, utils
- Other packages: AnnotationDbi 1.42.0, Biobase 2.40.0, BiocGenerics 0.26.0, FunciSNP 1.24.0, FunciSNP.data 1.15.0, GenomeInfoDb 1.16.0, GenomicFeatures 1.32.0, GenomicRanges 1.32.0, IRanges 2.14.0, S4Vectors 0.18.0, TxDb.Hsapiens.UCSC.hg19.knownGene 3.2.2, ggplot2 2.2.1
- Loaded via a namespace (and not attached): AnnotationFilter 1.4.0, BSgenome 1.48.0, BiocInstaller 1.30.0, BiocParallel 1.14.0, Biostrings 2.48.0, ChIPpeakAnno 3.14.0, DBI 0.8, DelayedArray 0.6.0, GO.db 3.6.0, GenomeInfoDbData 1.1.0, GenomicAlignments 1.16.0, MASS 7.3-50, Matrix 1.2-14, ProtGenerics 1.12.0, R6 2.2.2, RBGL 1.56.0, RCurl 1.95-4.10, RSQLite 2.1.0, Rcpp 0.12.16, Rsamtools 1.32.0, SummarizedExperiment 1.10.0, VariantAnnotation 1.26.0, VennDiagram 1.6.20, XML 3.98-1.11, XVector 0.20.0, ade4 1.7-11, assertthat 0.2.0, biomaRt 2.36.0, bit 1.1-12, bit64 0.9-7, bitops 1.0-6, blob 1.1.1, colorspace 1.3-2, compiler 3.5.0, curl 3.2, digest 0.6.15, ensembldb 2.4.0, formatR 1.5, futile.logger 1.4.3, futile.options 1.0.1, graph 1.58.0, grid 3.5.0, gtable 0.2.0, httr 1.3.1, idr 1.2, labeling 0.3, lambda.r 1.2.2, lattice 0.20-35, lazyeval 0.2.1, limma 3.36.0, magrittr 1.5, matrixStats 0.53.1, memoise 1.1.0, multtest 2.36.0, munsell 0.4.3, pillar 1.2.2, pkgconfig 2.0.1, plyr 1.8.4, prettyunits 1.0.2, progress 1.1.2, regioneR 1.12.0, reshape 0.8.7, rlang 0.2.0, rtracklayer 1.40.0, scales 0.5.0, seqinr 3.4-5, snpStats 1.30.0, splines 3.5.0, stringi 1.1.7, stringr 1.3.0, survival 2.42-3, tibble 1.4.2, tools 3.5.0, zlibbioc 1.26.0

Our recent paper describing FunciSNP and FunciSNP.data can be found in the Journal Nucleic Acids Research (doi:10.1093/nar/gks542).
This document was proudly made using LaTeXand **Sweave**.