seq2pathway Vignette

Bin Wang, Xinan (Holly) Yang

October 17, 2016

Contents

Abstract

`Seq2pathway` is a novel computational tool to analyze functional gene-sets (including signaling pathways) using variable next-generation sequencing data[1]. Integral to this tool are the "seq2gene" and "gene2pathway" components in series that infer a quantitative pathway-level profile for each sample. The `seq2gene` function assigns phenotype-associated significance of genomic regions to gene-level scores, where the significance could be p-values of SNPs or point mutations, protein-binding affinity, or transcriptional expression level. The `seq2gene` function has the feasibility to assign non-exon regions to a range of neighboring genes besides the nearest one, thus facilitating the study of functional non-coding elements[2]. Then the `gene2pathway` summarizes gene-level measurements to pathway-level scores, comparing the quantity of significance for gene members within a pathway with those outside a pathway. It implements an improved FAIME algorithm together with other three conventional gene-set enrichment analysis methods[3]. The output of `seq2pathway` is a general structured pathway scores, thus allowing one to functionally interpret phenotype-associated significance of genomic regions derived by next generational sequencing experiments.

Package Installation

Currently, seq2pathway works in both Linux and Windows. It has wrapped python scripts to annotate loci to genes, thus requires Python v2.7 running on the system. For Windows, the Python should be installed at "C:/python27/python" (default). Make sure supporting data package seq2pathway.data is installed with seq2pathway package.

If you don't have biocLite() you can get it like this:

```
source("http://bioconductor.org/biocLite.R")
biocLite("seq2pathway.data")
biocLite("seq2pathway")
```

Now install seq2pathway and the data package:

```
> library("seq2pathway.data")
> library("seq2pathway")
```

runseq2pathway

This function provides end-users a straightforward work-flow to implement the seq2pathway algorithms. It facilitates the screening of novel biological functions using just a few code lines, the main function to derive enriched pathways from genomic regions. It uses the Gene Ontology (GO)-defined gene-sets by default and can be run against either the MSigDB-defined[4] or customized gene-sets.

```
> head(runseq2pathway, n=8)
```

```
1 function (inputfile, search_radius = 150000, promoter_radius = 200,
2     promoter_radius2 = 100, genome = c("hg38", "hg19", "mm10",
3         "mm9"), adjacent = FALSE, SNP = FALSE, PromoterStop = FALSE,
4     NearestTwoDirection = TRUE, UTR3 = FALSE, DataBase = c("GOterm"),
5     FAIMETest = FALSE, FisherTest = TRUE, collapsemethod = c("MaxMean",
6         "function", "ME", "maxRowVariance", "MinMean", "absMinMean",
7         "absMaxMean", "Average"), alpha = 5, logCheck = FALSE,
8     B = 100, na.rm = FALSE, min_Intersect_Count = 5)
```

The inputs are almost the same as those introduced below for the the two main functions runseq2gene and gene2pathway_test. We therefore only introduce the new parameters here.

Note that the wrapped function runseq2pathway supports the "FAIME" method only and performs empirical test if the new parameter FAMETest equals to "TRUE".

If setting FAIMETest=TRUE and/or calculating the empirical p-values, an end-user should provide the formatted input file (see following example).

Column 1 the unique IDs (labels) of genomic regions of interest

Column 2 the chromosome IDs (eg. chr5 or 5)

Column 3 the start of genomic regions of interest

Column 4 the end of genomic regions (for SNP and point mutations, the difference of start and end is 1bp)

Column 5 the scores or values of the sample(s) along with the genomic regions

Column ... other custom-defined information

Another new parameter collapsemethod is a character for determining which method to use when call the function collapseRows in package WGCNA[5].

These are the options provided by WGCNA for the parameter collapsemethod(directly from WGCNA Vignette):

"MaxMean" (default) or "MinMean" = choose the row with the highest or lowest mean value, respectively

"maxRowVariance" = choose the row with the highest variance (across the columns of data)

"absMaxMean" or "absMinMean" = choose the row with the highest or lowest mean absolute value

"ME" = choose the eigenrow (first principal component of the rows in each group)

"Average" for each column, take the average value of the rows in each group

"function" use this method for a user-input function (see the description of the argument "methodFunction")

Two main functions

The output of runseq2pathway can be achieved equally by running `runseq2gene` and `gene2pathway_test` functions in series. These two functions facilitate end-users to track details on the gene-level. End-users can also apply the gene2pathway_test function to analyze functional enrichment for customized gene lists independently.

Here we introduce these two main functions separately. For each function, we describe the significance, its features with a flowchart, the inputs and parameters, then the output in details.

"runseq2gene" The first components in series to map genomic regions to coding and non-coding genes[2].

"gene2pathway_test" The second components in series to run pathway enrichment analysis for coding genes. This function provides three alternative pathway estimating methods which are FAIME[3], Kolmogorov-Smirnov test[6], and cumulative rank test[6].

seq2gene

Nearly 99% of human genome are non-coding nucleotides[7]. Identifying and delineating the function of all coding genes and non-coding elements remains a considerable challenge. We developed the computational function `runseq2gene` to link genomic regions of interest to genes in a many-to-many mapping, by considering the possibility that genes within a search radius in both directions from intergenic regions may fall under control of cis-regulation[2]. Using the `seq2gene` strategy with a search radius of 100k-base, our recent study in vivo defined a transcription factor-mediated cis-regulatory element from both ChIP-seq and transcriptomic data[8]. We also identified an intronic locus of one gene regulates the transcript of its neighbor gene instead of its host gene, suggesting the need to associate a functional genomic locus to broader candidate targets[9]. We thus suggest a larger search radius for the `seq2gene` function, such as 100k -150k bases, given that the average enhancer-promoter loop size is 120 kb in mammalian genomes[10] and enhancers act independently of their orientation[11][12].

seq2gene flowchart

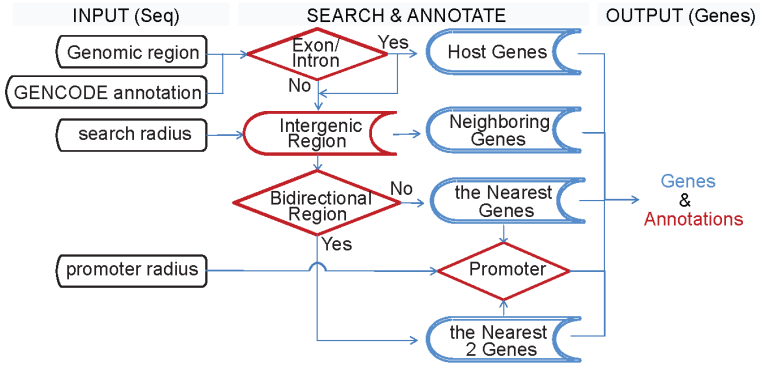


Figure 1: **Seq2gene flowchart**. The inputs are on the left, and the outputs are on the right.

Figure 1 gives the flowchart for the seq2gene process. Built on our previous publication[2], the current seq2gene uses the reference human genome annotation for the ENCODE project (GENCODE) [13] version 19 for human genome and version M4 for mouse genome (Ensembl version 78 in GRCm38). ENCODE is a re-merge between the Ensembl annotation and updates from HAVANA(http://www.gencodegenes.org/releases/). Table 1 lists the statistics of the gene annotations that are used by seq2pathway.

Table 1: Statistics about the seq2pathway-used GENCODE annotation.

Species	GENCODE Release	Corresponding Ensembl assembly	# of coding genes	# of Long non-coding RNAs	# of Small non-coding RNAs	# of Pseudogenes	# of all genes
Human	19(Dec.2013)	GRCh74/hg19	20345	13870	9013	14206	57820
Mouse	M4(Aug.2014)	GRCm38.p3/mm10	22032	6951	5853	7957	43346

The seq2gene algorithm uses a bisection strategy to search among exon and transcript annotations. Figure 2 is the pseudocode for the function[2]. To perform the basic bisect algorithm with respect to exon and transcript separately, we have prepared for end users the internal "exon.table" and "transcript.table" files based on the GENCODE general feature format. Both file use ENSEMBL IDs as the key index.

Algorithm: seq2gene
Input: peaks, exontable, transcripttable, search radius
Output: peak with annotated gene information

```
1.  for i:= 1 to length(peaks) do begin:
2.        m = peakleft
3.        n = peakright
4.        middle = (m+n)/2
5.        locate the nearest exon(J) for peak(i) by the basic bisect algorithm
6.        if peak(i) resides inside exon(J)
7.              report peak(i) with exon(J)
8.        endif
9.        while exon(x) intersecting with peak(i)
10.             report peak(i) with exon(x)
11.             exon(x) =the closest exons (left or right)
12.       endwhile
13.       locate the nearest transcript(H) by the basic bisect algorithm
14.       if peak(i) resides outside transcript(H)
15.             report peak(i) with transcript(H), intergenic region *
16.       else
17.             report peak(i) with transcript(H), intron region
18.       endif
19.       for transcripts(t) within the position of transcript(H)± search radius
20.             if peak(i) resides outside transcripts(t)
21.                   report peak(i) with transcripts(x), intergenic region *
22.             else
23.                   report peak(i) with transcripts(x), intron region
24.             endif
25.       end
26. end
```

*: more details about distance, promoter and bidirectional region judgment

Figure 2: **Pseudo-code of the seq2gene algorithm**.

runseq2gene inputs/parameters

inputfile An R object input file that records genomic region information (coordinates). This object could be a data frame defined as:

column 1 the unique IDs of peaks/mutations/SNPs;

column 2 the chromosome ID (eg. chr5 or 5);

column 3 the start site of genomic regions;

column 4 the end site of genomic regions (for SNP and point mutations, the difference of start and end is 1bp);

column 5 ... custom defined.

There is one demo data in data.frame format in our package.

```
> data(Chipseq_Peak_demo)
> class(Chipseq_Peak_demo)

[1] "data.frame"

> head(Chipseq_Peak_demo)

      peakID chrom    start      end signalvalue
1 Peak_59951 chr14 19003706 19004370    6.611026
2 Peak_59952 chr14 19003800 19024138    3.450042
```

```
3 Peak_59953 chr14 19005068 19005305   10.997456
4 Peak_59954 chr14 19006372 19006587   21.055350
5 Peak_59955 chr14 19013301 19013534    8.242503
```

Or, the input format could be a GRanges object (from R package GenomicRanges). There is a demo data in GRanges formart in our package as well.

```
> data(GRanges_demo)
> class(GRanges_demo)

[1] "GRanges"
attr(,"package")
[1] "GenomicRanges"

> GRanges_demo[1:3,]

GRanges object with 3 ranges and 3 metadata columns:
    seqnames    ranges strand |        name     score        GC
       <Rle> <IRanges>  <Rle> | <character> <integer> <numeric>
  a     chr1    [1, 7]      - |       peak1         1 1.0000000
  b     chr2    [2, 8]      + |       peak2         2 0.8888889
  c     chr2    [3, 9]      + |       peak3         3 0.7777778
  -------
  seqinfo: 3 sequences from an unspecified genome; no seqlengths
```

Note that for this particular GRanges object, the `seqnames`, `ranges`, `strand`, and `name` columns are necessary. And for a data frame object, the first four columns are orderly. Specifically, here are three more examples.

example 1:

peakID	chrom	chromstart	chromend	name	score	strand	thickstart	thickend
peak2	chr7	127477031	127478198	Neg2	0	-	127477031	127478198
peak3	chr7	127478198	127479365	Neg3	0	-	127478198	127479365

example 2:

peakID	Chr	Start	End
MACS_M_1210	chr9	21754771	21755152
MACS_M_1211	chr9	21753771	21754023
MACS_M_1212	chr9	21753901	21754023

example 3:

SNP	chr	Physical_position	position_end
rs953509	9	81560347	81560348
rs719293	2	50516523	50516524
rs1394384	17	28813156	28813157
rs1609772	1	186820222	186820223

search_radius(unit bp) A non-negative integer, with which the input genomic regions can be assigned not only to the matched/nearest gene, but also with all genes within a search radius. Default is 150000. Figure 3 illustrates the definition of search_radius, being calculated from the middle of a genomic region to both sides.

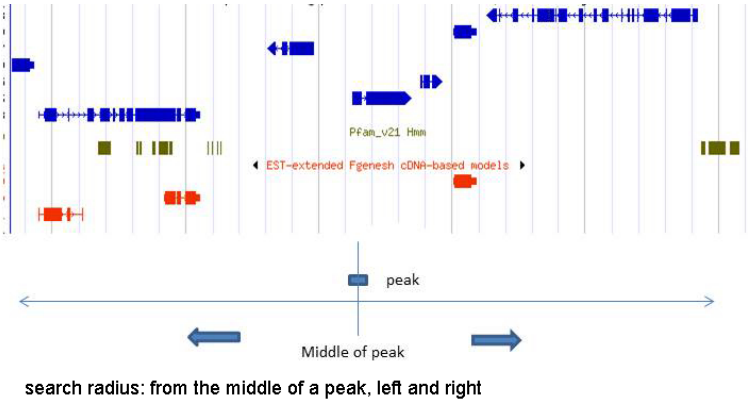


Figure 3: **The illustration of parameter search_radius**.(Modified from genome.igi.doe.gov/help/brwser_viewer.jsp)

promoter_radius(unit bp) A non-negative integer. Default is 200.
Note that promoters are calculated from transcription start site (TSS) of genes (Figure 4). Promoters can be about 100-2000 base pairs upstream of their TSSs[14]. User can assign the promoter_radius to defind promoter regions in the genome.

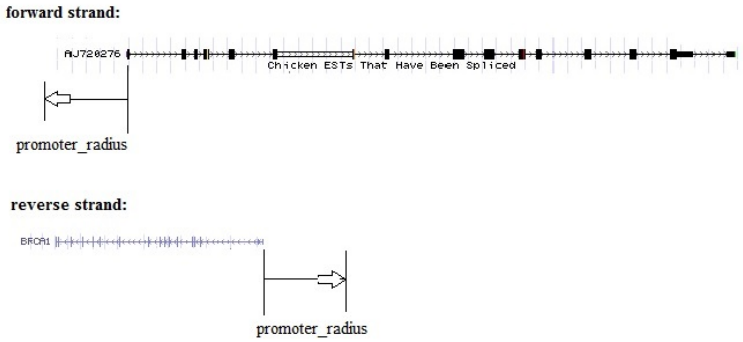


Figure 4: **The illustration of parameter promoter_radius**.(Edited from the UCSC genome browser)

promoter_radius2(unit bp) A non-negative integer. Default is 100. User can as well use this parameter to defined downstream regions of the TSSs as promoter.

genome A character specifies the genome type. Currently, "hg38", "hg19"(human), and "mm10", "mm9"(mouse) are supported.

adjacent A Boolean. Default is FALSE to search all genes within the search_radius. Using "TRUE" to find the adjacent genes only and ignore parameters "SNP" and "search_radius".

SNP A Boolean specifies the input object type. By default is FALSE to keep on searching for intron and neighboring genes. Otherwise, runseq2gene stops searching when the input genomic region is residing on a coding gene exon.

PromoterStop A Boolean, "FALSE" by default to keep on searching neighboring genes using the parameter "search_radius". Otherwise, runseq2gene stops searching for neighboring genes. This parameter has function only if an input genomic region map to promoter of coding gene(s).

NearestTwoDirection A boolean, "TRUE" by default to output the closest left and closest right coding genes with directions. Otherwise, output only the nearest coding gene regardless of direction.

UTR3 A boolean, "FALSE" by default to calculate the distance from genes' 5UTR. Otherwsie, calculate the distance from genes' 3UTR.

runseq2gene outputs

The function runseq2gene outputs a matrix structured below.

`Columns 1-4` The same as the first four columns in the input file.

`Columns 5` **PeakLength** An integer gives the length of the input genomic region. It is the number of base pairs between the start and end of the region.

`Columns 6` **PeakMtoStart_Overlap** An integer gives the distance from the TSS of mapped gene to the middle of the genomic region. A negative signal only shows TSS of the mapped gene is at the right of the peak (Figure 5 A-B). Otherwise, PeakMtoStart_Overlap reports a numeric range showing the location of overlapped coordinates (Figure 5 C).

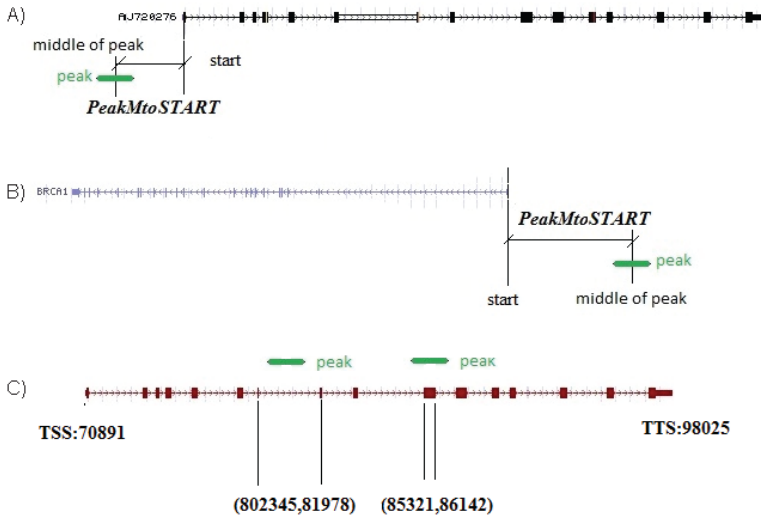


Figure 5: **The calculation of output PeakMtoStart_Overlap**. Scenarios could be an intergenic region of interest resides at the upstream (A) or downstream (B) of a coding gene, or a genomic region overlaps with intron or exon of a coding gene (C).

`Columns 7` **type** A character specifies the relationship between the genomic region and the mapped gene (Figure 6)

"Exon" any part of a genomic region overlaps the exon region of the mapped gene;

"Intron" any part of a genomic region overlaps an intron region but not at exon region of the mapped gene;

"cds" any part of a genomic region overlaps the CDS region;

"utr" any part of a genomic region overlaps a UTR region;

"promoter" any part of a genomic region overlaps the promoter region of the mapped gene based on an intergenic region of mapped gene covers the input genomic region;

"promoter_internal" any part of a genomic region overlaps the promoter region of the mapped gene when an adjacent TTS region of mapped gene covers the input genomic region;

"Neareast" the mapped gene is the nearest gene if the genomic region is located in an intergenic region. "L" and "R" show the relative location of mapped genes;

"Neighbor" any mapped genes within the search radius but belongs to none of the prior types.

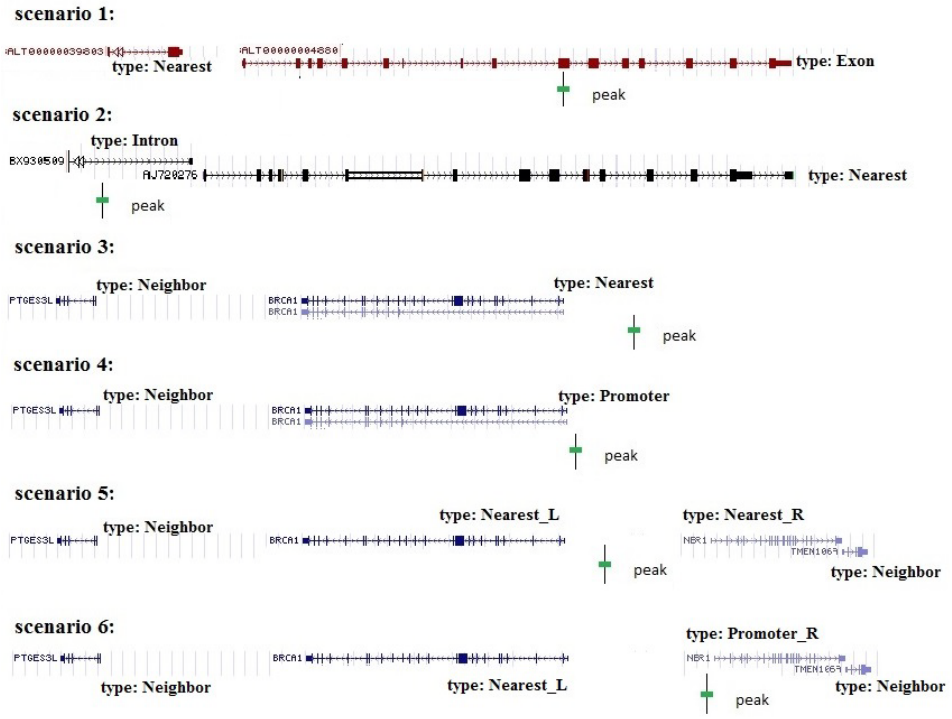


Figure 6: **Six output type values in several scenarios**. In each scenario, we map the genomic region of interest in green to the following types of a coding gene: exon (1), intron (2), the nearest (3), promoter (4), Nearest_L and Nearest_R (5), or Promoter_R (6).

`Columns 8` **BidirectionalRegion** A Boolean indicates whether or not the input genomic region is in bidirectional region (Figure 7).

A "bidirectional gene pair" refers to two adjacent genes coded on opposite strands, with their 5' UTRs oriented toward one another. NA means the genomic region is at exon or intron region.

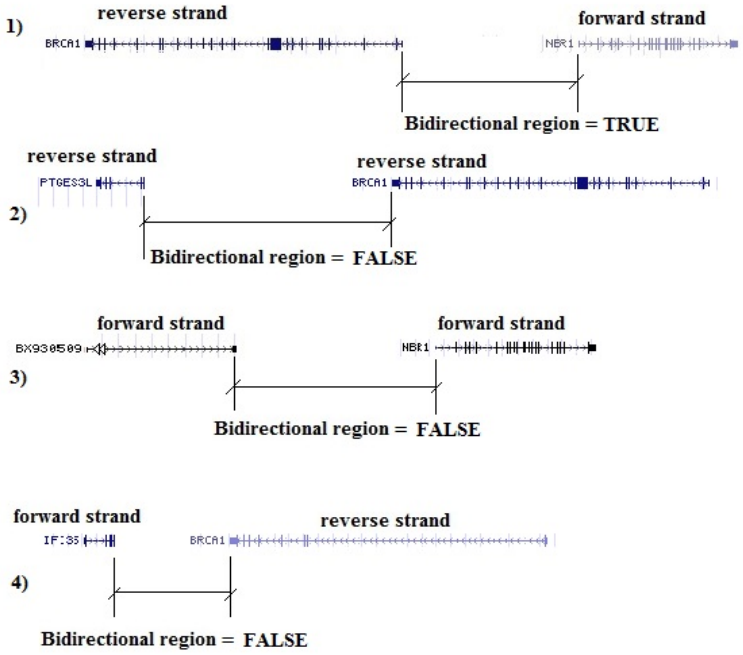


Figure 7: **The definition of output BidirectionalRegion in several scenarios**. (1) Two adjacent genes code on opposite strands, with their 5' ends oriented toward one another: Bidirectional region=TRUE. (2) Both two adjacent genes code on reverse strands: Bidirectional region=FALSE. (3) Both two adjacent genes code on forward strands: Bidirectional region=FALSE. (4) Two adjacent genes code on opposite strands, with their 3' ends oriented toward one another: Bidirectional region=FALSE.

`Columns 9` **Chr** An integer gives chromosome number of mapped gene.

`Columns 10` **TSS** An integer indicates transcription start site of mapped gene regardless of strand.

`Columns 11` **TTS** An integer indicates transcription termination site of mapped gene regardless of strand.

`Columns 12` **strand** a character indicates whether gene is in forward (+) or reverse (-) direction on chromosome.

`Columns 13` **gene_name** A character gives official gene name of mapped genes.

`Columns 14` **source** a character gives gene source (Ensembl classification) of mapped genes.

`Columns 15` **transID** A character gives Ensemble transcript ID of mapped genes.

gene2pathway

The gene2pathway step integrates several featured GSA (geneset analysis) algorithms, characterized by the improved FAIME method (Functional Analysis of Individual Microarray/RNAseq Expression)[3][19]. We initially developed FAIME for transcriptomic analysis, which compares the cumulative quantitative effects of genes inside an ontology (set of functional related genes) with those outside thus overcoming a number of difficulties in prior GSA methods[3]. However, sensitivity of the FAIME algorithm remains a challenge as, at a significance level of false discovery rate (FDR) of 0.05, FAIME could identify hundreds of gene-sets, an impractical number for wet-lab validation. Therefore, we introduce in this package a new weighting parameter into the FAIME algorithm to better control the type-I error, especially for large gene-sets. Additionally, we recently used gene2pathway to integrate microarray and RNA-seq data for gene-set analysis (manuscript submitted).

Here we develop the function gene2path_test as an improved tool for functionally analyzing versatile next generation sequencing data by taking account of quantitative sequence measurements. This function implements the improved FAIME algorithm. This function can run the classical Fisher's exact test or novel gene2pathway tests.

gene2pathway flowchart

Figure 8 gives the flowchart for the gene2pathway process. Hereafter we use "pathway" to refer functional gene-sets for simplification.

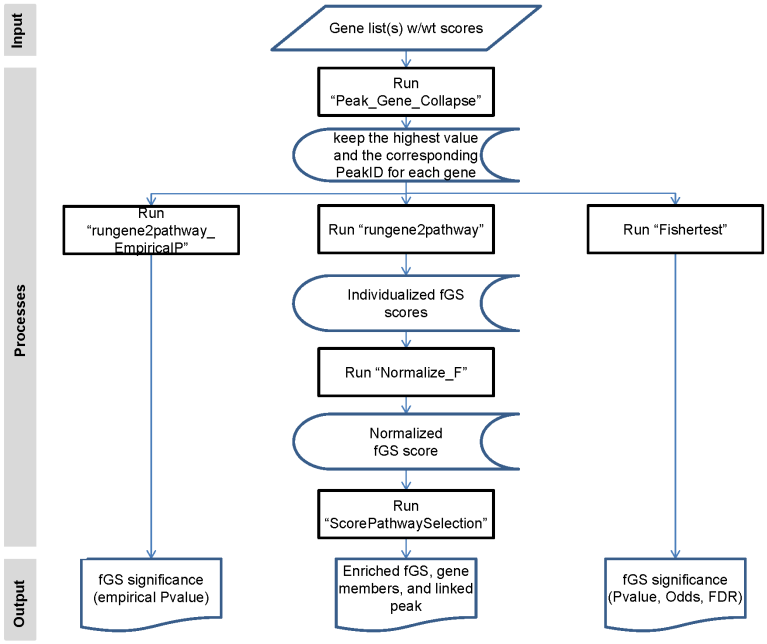


Figure 8: **gene2pathway flowchart**.

gene2pathway_test inputs/parameters

dat A data frame of gene expression or a matrix of sequencing derived gene-level measurements. The rows of dat correspond to genes, and the columns correspond to sample profile (eg. Chip-seq peak scores, somatic mutation p-values, RNS-seq or microarray gene expression values).
Note that official gene symbols must label the **dat** rows. The values contained in dat should be either finite or NA. For example:

```
          Peak.Score
ARHGEF10   65.21356
ARHGAP31   50.42416
B4GALT4    50.42416
```

DataBase A character string assigns an R GSA.genesets object to define gene-set. User can call GSA.read.gmt function in R GSA package to load customized gene-sets with a .gmt format. If not specified, GO defined gene sets (BP, MF, CC) will be used. For example,

```
> data(MsigDB_C5,package="seq2pathway.data")
> class(MsigDB_C5)

[1] "GSA.genesets"
```

FisherTest A Boolean value. By default is TRUE to execute the function of the Fisher's exact test. Otherwise, only executes the function of gene2pathway test.

EmpiricalTest A Boolean value. By default is FALSE for multiple-sample dat. When true, gene2pathway_test calculates empirical p-values for gene-sets.

method A character string determines which method to calculate the pathway scores. Currently, "FAIME" (default), "KS-rank", and "cumulative-rank" are supported.

genome A character specifies the genome type. Currently, choice of "hg38", "hg19", "mm10", and "mm9" is supported.

alpha A positive integer, 5 by default. This is a FAIME-specific parameter. A higher value puts more weights on the most highly-expressed ranks than the lower expressed ranks[3] [15].

logCheck A Boolean value. By default is FALSE. When true, take the log-transformed values of all genes if the maximum value of sample profile is larger than 20.

na.rm A Boolean value indicates whether to keep missing values or not when method="FAIME". By default is FALSE.

B A positive integer assigns the total number of random sampling trials to calculate the empirical p values. By default is 100.

min_Intersect_Count A number decides the cutoff of the minimum number of intersected genes when reporting Fisher's exact tested results.

gene2pathway_test outputs

A list or data frame. If the parameter FisherTest is true, the result is a list including both reports for Fisher's exact test and the gene2pathway test. Otherwise, only reports the gen2pathway test results. For example, below Table 4.2.3 is the head of result of gene2pathway test.

	Des	TCGA 28412 pathscore Normalized	TCGA 28402 pathscore Normalized	TCGA 28432 pathscore Normalized	TCGA 28422 pathscore Normalized	TCGA 28452 pathscore Normalized	Intersect Count	Intersect gene
NUCLEOPLASM	http://www.broadinstitute.org/gsea/msigdb/cards/NUCLEOPLASM	0.3800166	0.7017463	0.60702357	0.72972712	0.8866237	37	ACTB ACTL6A ACTL6B APPL1 APPL2 APTX ARID1A ARID1B ARID4A ARNTL ASF1A ASH2L ATF6 ATXN1 ATXN3 BNIP3 C19ORF2 C1ORF124 CBX1 CCNO CD3EAP CDK8 CDK9 CDKN2A CDKN2AIP CHAF1A CHAF1B CHEK2 CIB1 CIR1 CLOCK COIL CPSF1 CPSF3 CPSF3L CPSF6 DKC1
ORGANELLE_PART	http://www.broadinstitute.org/gsea/msigdb/cards/ORGANELLE_PART	0.7516177	0.8067336	0.82731623	0.86229571	0.8968370	272	A1CF AAAS AADAC ABCA2 ABCB6 ABCB7 ABCB8 ABCC4 ABCD3 ABCF2 ABL1 ACADM ACD ACN9 ACR ACTA1 ACTB ACTC1 ACTL6A ACTL6B ACTN2 ACTN3 ACTR1A ACTR1B ACTR2 ACTR3 ADAM10 ADAP2 AFTPH AGFG1 AIFM2 AIFM3 AKAP9 ALAS2 ALDH4A1 ALG3 ALMS1 ALS2 AMFR AMOT ANAPC11 ANAPC4 ANAPC5 ANG ANKFY1 ANLN AP1G2 AP1S1 AP2S1 AP3B2 AP4B1 AP4M1 APC API5 APOBEC3F APOBEC3G APPBP2 APPL1 APPL2 APTX ARCN1 ARFGEF2 ARFIP1 ARHGEF2 ARID1A ARID1B ARID4A ARL8A ARL8B ARNTL ARPC1B ARPC2 ARPC3 ARPC4 ARPC5 ASF1A ASH2L ASNA1 ASPH ATF6 ATG4A ATG4B ATG4C ATG4D ATP2C1 ATP5A1 ATP5B ATP5C1 ATP5D ATP5E ATP5F1 ATP5G1 ATP5G2 ATP5G3 ATP5J ATP5O ATP6V1B1 ATP7A ATP7B ATRX ATXN1 ATXN2 ATXN3 AURKA AURKC AZI1 B3GALT6 B4GALT1 BARD1 BAX BBS4 BCAS2 BCKDHA BCKDHB BCKDK BCL2 BCL6 BCS1L BET1 BFSP2 BIRC5 BMF BNIP1 BNIP2 BNIP3 BNIP3L BRCA1 BRCA2 BRCC3 BRE BSCL2 BUB1 BUB1B BUB3 C15ORF29 C19ORF2 C1ORF124 CABP1 CACNA1C CALR CAPG CAPZA1 CAPZA2 CAPZB CASP7 CASQ1 CAV1 CBX1 CBX5 CBY1 CCNH CCNO CCNT1 CD2AP CD3EAP CD63 CDC16 CDC20 CDC23 CDC26 CDC27 CDC40 CDCA5 CDK1 CDK5RAP2 CDK8 CDK9 CDKN2A CDKN2AIP CDT1 CENPA CENPC1 CENPE CENPF CEP250 CEP290 CEP57 CEP63 CETN1 CETN3 CHAF1A CHAF1B CHEK1 CHEK2 CHMP1A CHST2 CHST4 CIB1 CIR1 CIRH1A CKAP5 CLASP1 CLASP2 CLCN3 CLIP1 CLIP2 CLN3 CLN5 CLN6 CLOCK CNTROB COG1 COG2 COG3 COG4 COG5 COG6 COG7 COG8 COIL COPA COPB1 COPB2 COPE COPG COPG2 COPS2 COPZ1 CORO1A COX15 COX18 COX6B2 CPSF1 CPSF3 CPSF3L CPSF6 CROCC CS CSPG5 CTAG2 CTDNEP1 CTNS CUZD1 CWC22 CYCS CYLC1 DAD1 DBT DCTN1 DCTN2 DCTN3 DCTN4 DCX DDOST DDX11 DDX19B DDX21 DDX23 DDX24 DDX47 DDX54 DDX56 DEDD DEDD2 DERL1 DERL2 DERL3 DHCR7 DHRS9 DHX15 DHX8 DKC1 DLGAP5 DMBT1 DMC1 DNAH9 DNAI2 DNAJA3 DNAJB9 DNALI1 DNM1L DNMT3A
CELL_PROJECTION_PART	http://www.broadinstitute.org/gsea/msigdb/cards/CELL_PROJECTION_PART	-1.0863671	-1.1430708	-0.89560385	-0.76891405	-0.9987234	9	ACTN2 ATP6V0A4 B4GALT1 CABP4 CDK5R1 CROCC DNAH9 DNAI2 DNALI1
CYTOPLASMIC_VESICLE_MEMBRANE	http://www.broadinstitute.org/gsea/msigdb/cards/CYTOPLASMIC_VESICLE_MEMBRANE	1.5531183	1.5750084	1.51152263	1.79550412	0.2484891	15	ABCC4 AFTPH AP1G2 AP1S1 AP2S1 ARCN1 COPA COPB1 COPB2 COPE COPG COPG2 COPZ1 CSPG5 CUZD1 DMBT1
GOLGI_MEMBRANE	http://www.broadinstitute.org/gsea/msigdb/cards/GOLGI_MEMBRANE	0.1813367	0.1063748	0.03454226	0.29232424	0.2617705	8	AFTPH AP1G2 AP1S1 ARFGEF2 ARFIP1 ATP2C1 ATP7A BET1 BNIP3 CAV1 CLN3 COG2 COPB1 COX18 CSPG5

Table 2: result of gene2pathway

Examples

The most critical issue in functionally interpreting genomic loci is to bridge non-coding regions with gene function. `Seq2pathway` offers the capability to discover pathway enrichment caused by long-distance cis-regulation of functional non-coding loci. Here we demonstrate the application on ChIP-seq and RNA-seq data analysis respectively. For ChIP-seq data, we demonstrate a use of `runseq2gene` and `gene2pathway_test` in series. To facilitate the comparison with conventional Fisher's exact test, we demonstrated the use of two additional functions below.

"FisherTest_GO_BP_MF_CC" The GO enrichment analysis for coding genes using Fisher's exact test.

"FisherTest_MsigDB" The MSigDB[4] defined functional gene-set enrichment analysis for coding genes using the Fisher's exact test.

ChIP-seq data analysis

Map ChIP-seq enriched peaks to genes using `runseq2gene`

`runseq2gene()` is one of the key functions in the seq2pathway package. The `runseq2gene` links sequence-level measurements of genomic regions (including ChIP-seq peaks, SNPs or point mutation coordinates) to gene-level scores. The function has the option to assign non-exon regions to a broader range of neighboring genes than the nearest one, thus facilitating the study of functional non-coding elements. Currently, Seq2pathway only works in Linux or windows

with python2.7 environment, as it has wrapped python scripts to annotate loci to genes.
To execute runseq2gene, we need to assign input file. An example of inputfile, Chipseq_Peak_demo, is included in the package.

```
> data(Chipseq_Peak_demo)
> head(Chipseq_Peak_demo)
```

```
      peakID chrom    start      end signalvalue
1 Peak_59951 chr14 19003706 19004370    6.611026
2 Peak_59952 chr14 19003800 19024138    3.450042
3 Peak_59953 chr14 19005068 19005305   10.997456
4 Peak_59954 chr14 19006372 19006587   21.055350
5 Peak_59955 chr14 19013301 19013534    8.242503
```

Then user can run demo data below:

```
> Chipseq_anno <- runseq2gene(
+                   inputfile=Chipseq_Peak_demo,
+                   genome="hg38", adjacent=FALSE, SNP=FALSE, search_radius=1000,
+                   PromoterStop=FALSE,NearestTwoDirection=TRUE)
```

```
[1] "python process start: 2016-10-17 19:48:29.091374"
[2] "Load Reference"
[3] "Check Reference files"
[4] "fixed reference done: 2016-10-17 19:48:51.971338"
[5] "Start Annotation"
[6] "Finish Annotation"
[7] "python process end: 2016-10-17 19:48:52.066006"
```

```
> class(Chipseq_anno)
```

```
[1] "list"
```

```
> head(Chipseq_anno[[1]])
```

```
      peakID chrom    start      end PeakLength peakMtoStart_Overlap
1 Peak_59951 chr14 19003706 19004370        664               -36604
2 Peak_59951 chr14 19003706 19004370        664               430303
3 Peak_59952 chr14 19003800 19024138      20338 [19024090, 19024286]
4 Peak_59952 chr14 19003800 19024138      20338                10121
5 Peak_59952 chr14 19003800 19024138      20338               -46535
6 Peak_59952 chr14 19003800 19024138      20338               420372
               type BidirenctionalRegion   Chr      TSS      TTS strand
1         Nearest_L                    N chr14 18967434 18999012      +
2         Nearest_R                    N chr14 19402486 19434341      -
3              Exon                 <NA> chr14 19024090 19055551      +
4 Promoter_internal                 <NA> chr14 19024090 19055551      +
5         Nearest_L                    N chr14 18967434 18999012      +
6         Nearest_R                    N chr14 19402486 19434341      -
      gene_name         source            transID
1         POTEG protein_coding  ENSG00000222036.4
2         POTEM protein_coding ENSG00000187537.10
3 CTD-2314B22.1        lincRNA  ENSG00000258314.3
4 CTD-2314B22.1        lincRNA  ENSG00000258314.3
5         POTEG protein_coding  ENSG00000222036.4
6         POTEM protein_coding ENSG00000187537.10
```

Discover enriched GO terms using `gene2pathway_test` **with gene scores**

After mapping peaks to genes, we will practice `gene2pathway_test` function. This function summarizes gene scores to pathway-scores for each sample. The function `gene2pathway_test` includes `rungene2pathway` function, which summarizes gene scores to pathway-scores for each sample, and is another main function in our package. The `rungene2pathway` function provides different methods ("FAIME", "KS-rank", and "cumulative-rank") to convert gene-level measurements to pathway-level scores. The function `gene2pathway_test` also includes `FisherTest` function to perform conventional Fisher's exact test (FET). The `FisherTest` function uses the corrected, common gene background for selected pathways. Hereafter we use "pathway" to refer functional gene-sets including GO for simplification. Following are R exampling codes.

#Example1:
Running FAIME and FET against MSigDB defined gene-sets with empirical p-values

```
> ## give the previously defined gene-sets
> data(MsigDB_C5,package="seq2pathway.data")
> class(MsigDB_C5)

[1] "GSA.genesets"

> ## load the gene-level measurements, here is an example of ChIP-seq scores
> data(dat_chip)
> head(dat_chip)

           peakscore
ABCD4       8.433123
ABHD12B     9.526305
ABHD4       9.988747
AC004817.1 10.086676
AC005477.1 10.086676
AC007375.1 10.186544

> result_FAIME<-gene2pathway_test(dat= dat_chip, DataBase= MsigDB_C5,
                FisherTest=TRUE, EmpiricalTest=TRUE, method="FAIME",
                alpha=5, logCheckALSE,  na.rm=FALSE)
```

The `output` will be a list, which include two data frame. One data set is the result of Fisher's exact test, with the geneset from MSigDB[4], the other is the result of `rungene2pathway` function with method "FAIME". We calculated empirical p-values for a single sample.

#Example2:
Running FAIME and FET against GO defined gene-sets with empirical p-values

```
> result_FAIME<-gene2pathway_test(dat= dat_chip,
                FisherTest=TRUE, EmpiricalTest=TRUE, method="FAIME",
                alpha=5, logCheck=FALSE, na.rm=FALSE)
```

In our package, there is an R resultant object `dat_gene2path_chip` as demo of `result_FAIME`.

```
> data(dat_gene2path_chip,package="seq2pathway.data")
> names(dat_gene2path_chip)

[1] "gene2pathway_result.2"   "gene2pathway_result.FET"

> class(dat_gene2path_chip$gene2pathway_result.2)
```

```
[1] "list"
```

```
> names(dat_gene2path_chip$gene2pathway_result.2)
```

```
[1] "GO_BP" "GO_CC" "GO_MF"
```

```
> dat_gene2path_chip$gene2pathway_result.2$GO_BP[1:3,]
```

```
GO:0000082 The mitotic cell cycle transition by which a cell in G1 commits to S phase. The process beg
GO:0000086           The mitotic cell cycle transition by which a cell in G2 commits to M phase. The p
GO:0000122
           peakscore2pathscore_Normalized peakscore2pathscore_Pvalue
GO:0000082                      0.3201774                       0.12
GO:0000086                     -0.3358601                       0.49
GO:0000122                     -0.1153585                       0.16
           Intersect_Count
GO:0000082              11
GO:0000086               5
GO:0000122              20

GO:0000082                                                 CDKN3 GPR132 MNAT1 POLE2 PSMA3 PSMA6 PSMB5 P
GO:0000086                                                                                       AJUBA DYNC1H
GO:0000122 AJUBA BMP4 DACT1 DICER1 ESR2 FOXA1 GSC JDP2 NKX2-1 PPM1A PRMT5 PSEN1 RCOR1 SALL2 SIX1 SNW1
```

```
> class(dat_gene2path_chip$gene2pathway_result.FET)
```

```
[1] "list"
```

```
> names(dat_gene2path_chip$gene2pathway_result.FET)
```

```
[1] "GO_BP" "GO_CC" "GO_MF"
```

```
> colnames(dat_gene2path_chip$gene2pathway_result.FET$GO_BP)
```

```
[1] "GOID"                 "Description"          "Fisher_Pvalue"
[4] "Fisher_odds"          "FDR"                  "Intersect_Count"
[7] "GO_gene_inBackground" "GO_gene_raw_Count"    "Intersect_gene"
```

```
> dat_gene2path_chip$gene2pathway_result.FET$GO_BP[1:3,-2]
```

```
        GOID   Fisher_Pvalue Fisher_odds           FDR Intersect_Count
1 GO:0030162 0.0000001173994    11.80262 0.00001361833              10
2 GO:0090501 0.0000136148154    15.12201 0.00078965929               6
3 GO:0006521 0.0001252247923     6.11356 0.00338658460               8
  GO_gene_inBackground GO_gene_raw_Count
1                   38                39
2                   19                19
3                   51                51
                                                                    Intersect_gene
1 SERPINA3 SERPINA6 SERPINA5 SERPINA1 SERPINA4 TRAF3 SERPINA10 SERPINA12 SERPINA11 SERPINA9
2                                                 ANG RNASE2 RNASE3 RNASE6 DICER1 RNASE7
3                                      PSMA3 PSMA6 PSMB5 PSMC1 PSMC6 PSME1 PSME2 PSMB11
```

#Example 3:
Running FAIME and FET against GO defined gene-sets without empirical p-values

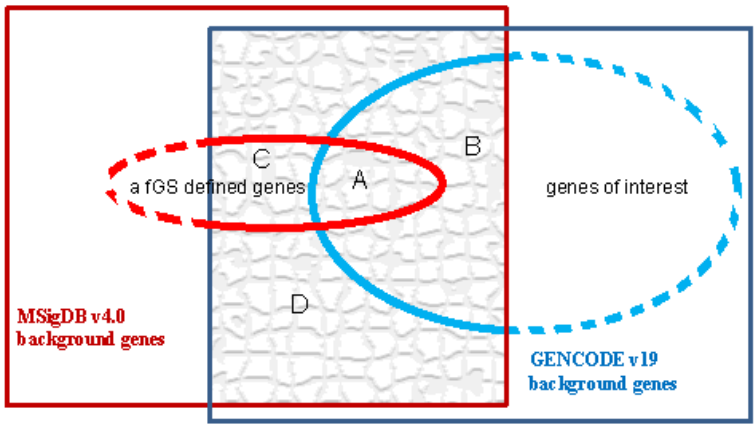
```
> result_FAIME<-gene2pathway_test(dat= dat_chip, FisherTest=TRUE, EmpiricalTestALSE,
                method="FAIME", alpha=5, logCheck=FALSE, na.rm=FALSE)
```

#Example 4:
Running FAIME only against GO defined gene-sets with empirical p-values

```
> result_FAIME<-gene2pathway_test(dat= dat_chip, FisherTest=FALSE,EmpiricalTest=TRUE,
                method="FAIME", alpha=5, logCheck=FALSE, na.rm=FALSE)
```

Discover enriched GO terms using Fisher's Exact test without gene scores

There are two functions to run FET in the package seq2pathway. Both perform conditional FET with modified gene background that is the common genes between genome and the gene-set database, e.g., MSigDB (Figure 9)[2]. The `FisherTest_GO_BP_MF_CC` function uses GO (GO.db_2.14.0) defined gene-sets, and the `FisherTest_MsigDB` function requires MsidDB defined gene-sets as input.



contingency table

	In fGS	Out fGS
DE	A	B
Not DE	C	D

$$p = \frac{(A+B)!(C+D)!(A+C)!(B+D)!}{A!B!C!D!\ (A+B+C+D)!}$$

Figure 9: **Conditional Fisher's exact test with corrected common background**. The common background between genome and the gene-set database, e.g., MSigDB, is illustrated as a grey region, which contains around 22,000 human coding genes or 15,546 mouse coding genes.

FisherTest_MsigDB function:

- Inputs/parameters:

  gsmap An R GSA.genesets object defined by the package "GSA" for functional gene-set (or termed as pathway for simplification). For example,

  ```
  > data(MsigDB_C5,package="seq2pathway.data")
  > class(MsigDB_C5)

  [1] "GSA.genesets"
  ```

  gs A characteristic vector of gene symbols of interest.

genome A character specifies the genome type. Currently, choice of "hg38", "hg19", "mm10", and "mm9" is supported.

min_Intersect_Count A number decides the cutoff of the minimum number of intersected genes when reporting Fisher's exact tested results.

- Output:
  A data frame of Fisher's exact tested result with the following columns:

  GeneSet MsigDB gene-set names (ID)

  Description MSigDB definition and description for the gene-sets

  Fisher_Pvalue the raw P values

  Fisher_odds estimate of the odds ratios

  FDR the multi-test adjusted P values using the Benjamini and Hochberg method[16]

  Intersect_Count the sizes of the overlap between gene-set genes and the input gene list

  MsigDB_gene_inBackground the counts of genes among each MSigDB gene-set that are also within the given genome background

  MsigDB_gene_raw_Count the original counts of genes in each MSigDB geneset

  Intersect_gene the intersecting genes' symbols

- An example:

```
> data(dat_chip)
> head(dat_chip)

           peakscore
ABCD4       8.433123
ABHD12B     9.526305
ABHD4       9.988747
AC004817.1 10.086676
AC005477.1 10.086676
AC007375.1 10.186544

> FS_test<-FisherTest_MsigDB(gsmap=MsigDB_C5, gs=as.vector(rownames(dat_chip)))
> head(FS_test)
```

GeneSet	Description	Fisher _Pvalue	Fisher _odds	FDR	Intersect _Count	MsigDB _gene _inBackground	MsigDB _gene _raw _Count	Intersect _gene
RIBONUCLEASE_ACTIVITY	http://www.broadinstitute.org/gsea/msigdb/cards/RIBONUCLEASE_ACTIVITY	1.881465e-08	19.268873	3.988705e-06	9	25	25	DICER1 ANG RNASE7 RNASE8 APEX1 RNASE1 RNASE2 RNASE3 RNASE6
NUCLEASE_ACTIVITY	http://www.broadinstitute.org/gsea/msigdb/cards/NUCLEASE_ACTIVITY	2.713796e-05	6.682127	2.876623e-03	9	55	55	DICER1 ANG RNASE7 RNASE8 APEX1 RNASE1 RNASE2 RNASE3 RNASE6
ENDONUCLEASE_ACTIVITY	http://www.broadinstitute.org/gsea/msigdb/cards/ENDONUCLEASE_ACTIVITY	6.848601e-04	8.419752	4.839678e-02	5	25	25	DICER1 ANG RNASE8 APEX1 RNASE1
TRANSCRIPTION _COACTIVA-TOR_ACTIVITY	http://www.broadinstitute.org/gsea/msigdb/cards/TRANSCRIPTION_COACTIVATOR_ACTIVITY	3.226841e-03	3.010833	1.710226e-01	10	123	123	YY1 RIPK3 SNW1 MAX GTF2A1 ESR2 MED6 NFATC4 TRIP11 APEX1
SEXUAL_REPRODUCTION	http://www.broadinstitute.org/gsea/msigdb/cards/SEXUAL_REPRODUCTION	2.006048e-02	2.358725	4.252823e-01	9	138	139	JAG2 REC8 PNMA1 BCL2L2 RPL10L ADAM20 ADAM21 SERPINA5 HSPA2
ACTIN_FILAMENT _BASED_PROCESS	http://www.broadinstitute.org/gsea/msigdb/cards/ACTIN_FILAMENT_BASED_PROCESS	1.866597e-02	2.548107	4.252823e-01	8	114	115	MYH7 MYH6 ARF6 EVL CDC42BPB RHOJ ANG PLEK2

FisherTest_GO_BP_MF_CC function:

- Inputs/parameters:

  gs A characteristic vector of gene symbols, the input genelist.
  Note that the `seq2pathway` package has prepared an internal R object `GO_MF_CC_BP_term_gene_lists_Fromorg.Hs.egGO2EG.rData`, which is formatted from `biomaRt_2.20.0` and `org.Hs.eg.db_2.14.0` gene symbols and `GO.db_2.14.0` gene ontologies.

  genome A character specifies the genome type. Currently, choice of "hg38", "hg19", "mm10", and "mm9" is supported.

  min_Intersect_Count A number decides the cutoff of the minimum number of intersected genes when reporting Fisher's exact test results.

  OntologyA character specifies the Gene Ontology, choice of "GOterm", "BP","MF", "CC" and "newOntology" is supported.

  newOntologyA list of two lists with the same ontology IDs. or each ontology ID, the 1st list is the lists of defined genes and the 2nd list is the desceiption.

- Outputs:
  A list of 3 data frames, each is a result of Fisher's exact test, using GO CC, BP, MF respectively. Each data frame reports FET results with the following columns.

  GOID GO term ID

  Description GO definition and description for the gene-sets based on the R object `GO.db_2.14.0`

  Fisher_Pvalue the raw P values

  Fisher_odds estimate of the odds ratios

  FDR the multi-test adjusted P values using the Benjamini and Hochberg method[16]

  Intersect_Count the sizes of the overlap between GO gene members and the input gene list

  GO_gene_inBackground the counts of genes among each GO term that are also within a given genome background

  GO_gene_raw_Count the original counts of genes in each GO term

  Intersect_gene the intersecting genes' symbols

- An example:

```
> data(dat_chip)
> head(dat_chip)

           peakscore
ABCD4       8.433123
ABHD12B     9.526305
ABHD4       9.988747
AC004817.1 10.086676
AC005477.1 10.086676
AC007375.1 10.186544

> FS_test<- FisherTest_GO_BP_MF_CC(gs=as.vector(rownames(dat_chip)),Ontology="BP")
[1] "Fisher's exact test done"
> head(FS_test$GO_BP)
```

GOID	Description	Fisher_Pvalue	Fisher_odds	FDR	Intersect_Count	GO_gene_inBackground	GO_gene_raw_Count	Intersect_gene
GO:0030162	Any process that modulates the frequency, rate or extent of the hydrolysis of a peptide bond or bonds within a protein.	1.173994e-07	11.802616	1.361833e-05	10	38	39	SERPINA3 SERPINA6 SERPINA5 SERPINA1 SERPINA4 TRAF3 SERPINA10 SERPINA12 SERPINA11 SERPINA9
GO:0090501	The RNA metabolic process in which the phosphodiester bonds between ribonucleotides are cleaved by hydrolysis.	1.361482e-05	15.122011	7.896593e-04	6	19	19	ANG RNASE2 RNASE3 RNASE6 DICER1 RNASE7
GO:0006521	Any process that modulates the frequency, rate or extent of the chemical reactions and pathways involving amino acids.	1.252248e-04	6.113560	3.386585e-03	8	51	51	PSMA3 PSMA6 PSMB5 PSMC1 PSMC6 PSME1 PSME2 PSMB11
GO:0006977	A cascade of processes induced by the cell cycle regulator phosphoprotein p53, or an equivalent protein, in response to the detection of DNA damage and resulting in the stopping or reduction in rate of the cell cycle.	1.459735e-04	5.195514	3.386585e-03	9	66	66	PPP2R5C PSMA3 PSMA6 PSMB5 PSMC1 PSMC6 PSME1 PSME2 PSMB11
GO:0034641	The chemical reactions and pathways involving various organic and inorganic nitrogenous compounds, as carried out by individual cells.	1.439655e-04	3.141606	3.386585e-03	16	185	185	ARG2 CKB DIO2 DIO3 DLST GSTZ1 ALDH6A1 PSMA3 PSMA6 PSMB5 PSMC1 PSMC6 PSME1 PSME2 SLC25A21 PSMB11
GO:0010951	Any process that decreases the frequency, rate or extent of endopeptidase activity, the endohydrolysis of peptide bonds within proteins.	2.437280e-04	4.333480	4.712075e-03	10	86	87	SERPINA3 AKT1 SERPINA6 SERPINA5 SERPINA1 SERPINA4 SERPINA10 SERPINA12 SERPINA11 SERPINA9

Add description for genes

The function addDescription is wrapped from R package "biomaRt" [17][18] to converts gene hgnc_symbol to gene description.

- Inputs/parameters:

  genome A character specifies the genome type. Currently, choice "hg19", "mm10", and "mm9" is supported.

  genevector A characteristic vector or list of gene symbols.

- Output:
  A data frame with two columns, first is the input genelist and second is the biomaRt gene description in details.

- An example:

```
> gene_description<-addDescription(genome="hg19",genevector=as.vector(rownames(dat_chip)))
> head(gene_description)
```

hgnc_symbol	description
ABCD4	ATP-binding cassette, sub-family D (ALD), member 4 [Source:HGNC Symbol;Acc:68]
ABHD12B	abhydrolase domain containing 12B [Source:HGNC Symbol;Acc:19837]
ABHD4	abhydrolase domain containing 4 [Source:HGNC Symbol;Acc:20154]
ACIN1	apoptotic chromatin condensation inducer 1 [Source:HGNC Symbol;Acc:17066]
ACOT1	acyl-CoA thioesterase 1 [Source:HGNC Symbol;Acc:33128]
ACOT2	acyl-CoA thioesterase 2 [Source:HGNC Symbol;Acc:18431]

RNA-seq data analysis

RNA-seq is increasingly used for measuring gene expression levels. Normally, RNA-seq measures multiple samples from more than one sample-groups. Base on expressions on the gene-level, user can run the `gene2pathway_test` function and skip the `runseq2gene()` function.

Here is an example to run `gene2pathway_test` function for RNA-seq data, using an example data in the package.

```
> data(dat_RNA)
> head(dat_RNA)
```

	TCGA_2841	TCGA_2840	TCGA_2843	TCGA_2842	TCGA_2845
A1BG	6.3606	10.2275	1.7113	1.7367	4.7184
A1BG-AS	8.7010	10.7700	2.5394	2.8203	7.8670
A1CF	0.0000	0.0000	0.0000	0.0000	0.0000
A2LD1	1.2489	1.3508	2.1397	1.9969	1.0495
A2M	0.2507	2.4767	3.3813	0.6906	1.7197
A2ML1	0.0710	0.0473	0.2541	0.0538	0.1098

Using the inputs similar to the example coding for ChIPseq data, the output of the `gene2pathway_test` function running RNAseq data will be a matrix of pathway scores for multiple samples.

```
> dat_gene2path_RNA <- gene2pathway_test(dat=dat_RNA, DataBase=MsigDB_C5,
            EmpiricalTest=FALSE, alpha=5, logCheck=FALSE, method="FAIME", na.rm=TRUE)
> head(dat_gene2path_RNA$gene2pathway_result.2)
```

	Des	TCGA 28412 pathscore Normalized	TCGA 28402 pathscore Normalized	TCGA 28432 pathscore Normalized	TCGA 28422 pathscore Normalized	TCGA 28452 pathscore Normalized	Intersect Count	Intersect gene
NUCLEOPLASM	http://www.broadinstitute.org/gsea/msigdb/cards/NUCLEOPLASM	0.3800166	0.7017463	0.60702357	0.72972712	0.8866237	37	ACTB ACTL6A ACTL6B APPL1 APPL2 APTX ARID1A ARID1B ARID4A ARNTL ASF1A ASH2L ATF6 ATXN1 ATXN3 BNIP3 C19ORF2 C1ORF124 CBX1 CCNO CD3EAP CDK8 CDK9 CDKN2A CDKN2AIP CHAF1A CHAF1B CHEK2 CIB1 CIR1 CLOCK COIL CPSF1 CPSF3 CPSF3L CPSF6 DKC1
ORGANELLE_PART	http://www.broadinstitute.org/gsea/msigdb/cards/ORGANELLE_PART	0.7516177	0.8067336	0.82731623	0.86229571	0.8968370	272	A1CF AAAS AADAC ABCA2 ABCB6 ABCB7 ABCB8 ABCC4 ABCD3 ABCF2 ABL1 ACADM ACD ACN9 ACR ACTA1 ACTB ACTC1 ACTL6A ACTL6B ACTN2 ACTN3 ACTR1A ACTR1B ACTR2 ACTR3 ADAM10 ADAP2 AFTPH AGFG1 AIFM2 AIFM3 AKAP9 ALAS2 ALDH4A1 ALG3 ALMS1 ALS2 AMFR AMOT ANAPC11 ANAPC4 ANAPC5 ANG ANKFY1 ANLN AP1G2 AP1S1 AP2S1 AP3B2 AP4B1 AP4M1 APC API5 APOBEC3F APOBEC3G APPBP2 APPL1 APPL2 APTX ARCN1 ARFGEF2 ARFIP1 ARHGEF2 ARID1A ARID1B ARID4A ARL8A ARL8B ARNTL ARPC1B ARPC2 ARPC3 ARPC4 ARPC5 ASF1A ASH2L ASNA1 ASPH ATF6 ATG4A ATG4B ATG4C ATG4D ATP2C1 ATP5A1 ATP5B ATP5C1 ATP5D ATP5E ATP5F1 ATP5G1 ATP5G2 ATP5G3 ATP5J ATP5O ATP6V1B1 ATP7A ATP7B ATRX ATXN1 ATXN2 ATXN3 AURKA AURKC AZI1 B3GALT6 B4GALT1 BARD1 BAX BBS4 BCAS2 BCKDHA BCKDHB BCKDK BCL2 BCL6 BCS1L BET1 BFSP2 BIRC5 BMF BNIP1 BNIP2 BNIP3 BNIP3L BRCA1 BRCA2 BRCC3 BRE BSCL2 BUB1 BUB1B BUB3 C15ORF29 C19ORF2 C1ORF124 CABP1 CACNA1C CALR CAPG CAPZA1 CAPZA2 CAPZB CASP7 CASQ1 CAV1 CBX1 CBX5 CBY1 CCNH CCNO CCNT1 CD2AP CD3EAP CD63 CDC16 CDC20 CDC23 CDC26 CDC27 CDC40 CDCA5 CDK1 CDK5RAP2 CDK8 CDK9 CDKN2A CDKN2AIP CDT1 CENPA CENPC1 CENPE CENPF CEP250 CEP290 CEP57 CEP63 CETN1 CETN3 CHAF1A CHAF1B CHEK1 CHEK2 CHMP1A CHST2 CHST4 CIB1 CIR1 CIRH1A CKAP5 CLASP1 CLASP2 CLCN3 CLIP1 CLIP2 CLN3 CLN5 CLN6 CLOCK CNTROB COG1 COG2 COG3 COG4 COG5 COG6 COG7 COG8 COIL COPA COPB1 COPB2 COPE COPG COPG2 COPS2 COPZ1 CORO1A COX15 COX18 COX6B2 CPSF1 CPSF3 CPSF3L CPSF6 CROCC CS CSPG5 CTAG2 CTDNEP1 CTNS CUZD1 CWC22 CYCS CYLC1 DAD1 DBT DCTN1 DCTN2 DCTN3 DCTN4 DCX DDOST DDX11 DDX19B DDX21 DDX23 DDX24 DDX47 DDX54 DDX56 DEDD DEDD2 DERL1 DERL2 DERL3 DHCR7 DHRS9 DHX15 DHX8 DKC1 DLGAP5 DMBT1 DMC1 DNAH9 DNAI2 DNAJA3 DNAJB9 DNALI1 DNM1L DNMT3A
CELL_PROJECTION_PART	http://www.broadinstitute.org/gsea/msigdb/cards/CELL_PROJECTION_PART	-1.0863671	-1.1430708	-0.89560385	-0.76891405	-0.9987234	9	ACTN2 ATP6V0A4 B4GALT1 CABP4 CDK5R1 CROCC DNAH9 DNAI2 DNALI1
CYTOPLASMIC_VESICLE_MEMBRANE	http://www.broadinstitute.org/gsea/msigdb/cards/CYTOPLASMIC_VESICLE_MEMBRANE	1.5531183	1.5750084	1.51152263	1.79550412	0.2484891	15	ABCC4 AFTPH AP1G2 AP1S1 AP2S1 ARCN1 COPA COPB1 COPB2 COPE COPG COPG2 COPZ1 CSPG5 CUZD1 DMBT1
GOLGI_MEMBRANE	http://www.broadinstitute.org/gsea/msigdb/cards/GOLGI_MEMBRANE	0.1813367	0.1063748	0.03454226	0.29232424	0.2617705	8	AFTPH AP1G2 AP1S1 ARFGEF2 ARFIP1 ATP2C1 ATP7A BET1 BNIP3 CAV1 CLN3 COG2 COPB1 COX18 CSPG5

```
> head(dat_gene2path_RNA$gene2pathway_result.FET)
```

GeneSet	Description	Fisher _Pvalue	Fisher _odds	FDR	Intersect _Count	MsigDB _gene _inBackground	MsigDB _gene _raw _Count	Intersect _gene
HYDROLASE_ACTIVITY _ACTING_ON_ACID _ANHYDRIDESCAT- ALYZING _TRANSMEM- BRANE_MOVEMENT_OF _SUBSTANCES	http://www.broadinstitute.org/gsea/ msigdb/cards/HYDROLASE _ACTIVITY_ACTING_ON_ACID _ANHYDRIDESCATALYZ- ING_TRANSMEMBRANE_MOVEMENT _OF_SUBSTANCES	1.606744e-20	56.6790665	1.584249e-17	37	39	39	ABCF1 ABCA8 ATP6V0E1 ATP1B1 ATP1B3 ATP1B2 ATP6V1B2 ABCA3 ATP2B1 ATP6V0C ATP2B3 ATP2B4 ATP8B1 ABCD3 ABCD4 ATP4A ABCB11 ATP4B ATP11B ATP1A3 ATP1A4 ATP1A1 ATP1A2 ABCB7 ABCG1 ABCG2 ATP6V1C1 ATP7A ATP2A2 ATP2A3 ATP2C1 ATP2A1 ABCC3 ABCC1 ABCC2 ATP7B ABCC6
CHEMOKINE_RECEPTOR _BINDING	http://www.broadinstitute.org/gsea/ msigdb/cards/CHEMOKINE _RECEP- TOR_BINDING	9.352298e-20	29.8938093	4.016531e-17	39	43	43	CXCL1 CCL1 CCL3 CCL2 CXCL5 CXCL3 CXCL2 C5 CXCL9 CCL8 CXCL6 CX3CL1 CXCL11 CCL5 CCL4 CCL28 CXCL12 CCL27 CCL7 CCL26 CXCL10 CCL24 CCL25 CCL22 CCL23 CCL20 CCL21 CKLF CCL19 CCL16 CCL15 CCL18 CCL17 CCL11 CCL13 CXCL14 CXCL13 CXCL16 CCR2
PRIMARY_ACTIVE _TRANSMEM- BRANE_TRANSPORTER _ACTIVITY	http://www.broadinstitute.org/gsea/ msigdb/cards/PRIMARY_ACTIVE _TRANSMEM- BRANE_TRANSPORTER_ACTIVITY	1.629424e-19	37.7811394	4.016531e-17	37	40	40	ABCA8 ABCF1 ATP6V0E1 ATP1B1 ATP1B3 ATP1B2 ATP6V1B2 ABCA3 ATP2B1 ATP6V0C ATP2B3 ATP2B4 ATP8B1 ABCD3 ABCD4 ATP4A ATP4B ABCB11 ATP11B ATP1A3 ATP1A4 ATP1A1 ATP1A2 ABCB7 ABCG1 ABCG2 ATP7A ATP6V1C1 ATP2A2 ATP2C1 ATP2A3 ATP2A1 ABCC3 ABCC1 ABCC2 ATP7B ABCC6
ATPASE_ACTIVITY _COUPLED_TO _MOVE- MENT_OF_SUBSTANCES	http://www.broadinstitute.org/gsea/ msigdb/cards/ATPASE_ACTIVITY _COUPLED_TO_MOVEMENT_OF _SUBSTANCES	1.629424e-19	37.7811394	4.016531e-17	37	40	40	ABCA8 ABCF1 ATP1B1 ATP6V0E1 ATP1B3 ATP1B2 ATP6V1B2 ABCA3 ATP2B1 ATP6V0C ATP2B3 ATP2B4 ATP8B1 ABCD3 ABCD4 ATP4A ABCB11 ATP4B ATP11B ATP1A3 ATP1A4 ATP1A1 ATP1A2 ABCB7 ABCG1 ABCG2 ATP7A ATP6V1C1 ATP2A2 ATP2C1 ATP2A3 ATP2A1 ABCC3 ABCC1 ABCC2 ABCC6 ATP7B
CHEMOKINE_ACTIVITY	http://www.broadinstitute.org/gsea/ msigdb/cards/CHEMOKINE_ACTIVITY	3.451615e-19	29.1130402	6.806585e-17	38	42	42	CXCL1 CCL1 CCL3 CCL2 CXCL5 CXCL3 CXCL2 C5 CXCL9 CCL8 CXCL6 CX3CL1 CCL5 CXCL11 CXCL12 CCL28 CCL4 CCL27 CCL7 CXCL10 CCL26 CCL24 CCL25 CCL22 CCL23 CCL20 CCL21 CKLF CCL19 CCL16 CCL15 CCL18 CCL17 CCL11 CCL13 CXCL14 CXCL13 CXCL16
BIOPOLYMER_METABOLIC _PROCESS	http://www.broadinstitute.org/gsea/ msigdb/cards/BIOPOLYMER _METABOLIC_PROCESS	1.206876e-15	0.5818938	1.983299e-13	294	1673	1684	BTK DHX38 BRAF ARIH1 DHX8 CELF1 ATR C19ORF2 ATM CDC42BPG CDC42BPA CDC42BPB CWC15 AUH BRD7 BRD8 ATF7IP BRF1 AIFM1 ARHGEF11 DHX15 DHX16 ATRX CSNK1D CSNK1E CDKN2A CDKN2D ATG7 BCL10 CSDA BICD1 CCL2 CXXC1 AIMP1 ATG3 ATF6 ATF5 ATF4 ATF7 ADRA1D DDB1 DDB2 DMC1 BRSK2 BRSK1 CEBPZ DCLK1 CEBPA CEBPB CEBPD CEBPG CBL BAX ALKBH1 DDX17 ANAPC2 BCR ANAPC5 ANAPC4 CD37 CAMK4 CAMK1 AMFR DEAF1 ACD CIDEA CCNO CTBP1 CCNK CCNH APTX CDK16 CDK17 AGA CSNK1A1 COG3 COG7 COG2 APOBEC3G APOBEC3F ATOH1 CSNK1G2 AHR CSNK1G3 CSTA BLM BMX BRCA2 BRCA1 DGCR8 ANG ALX1 ALK CD3EAP CD80 CD81 CDK11A CDK11B CAMK2B CAMK2A ATXN3 BMPR1B BMPR1A CRNKL1 CDC6 CCND1 CCND3 CCND2 CLOCK CREM CDC45 CCL11 B3GALNT1 ARID1A DDIT3 ACHE CNBP CCRN4L B4GALT7 ARID4A ALG1 ALG2 ALG5 ALG6 DEK ALG8 CLCF1 ARID5B ARID5A CDK9 CDK7 CDK2 CHRM3 CHRM1 APH1A ADAMTS13 APH1B A4GNT DBP CDX2 B3GALT5 B3GALT4 COL4A3BP CSGALNACT1 AGGF1 BMP4 BMP2 ADPRH BMP6 ADAR CCNT2 COPS2 COPS5 CCNT1 DMAP1 CAMKK2 CDT1 ASH2L ADAM10 CTNNBIP1 ASH1L CHM BCAS2 CIR1 CRABP2 DMPK DARS ARNTL DERL2 DERL1 ANAPC10 ANAPC11 CSTF3 CSTF2 AKTIP CSTF1 AKT1 CSK AKT3 AKT2 DAPK2 DAPK3 DAPK1 B3GALT2 DIS3 CRK BACE2 CPSF3L CRTC1 BNIP3 CUZD1 DAXX CARD14 ABI3 ABI2 ABI1 ARAF DBF4 CSNK2A1 CIB1 DFFA DFFB ABCA2 ASF1A BRIP1 DNASE2 ABL1 ABL2 DKC1 CYCS AGER CHIA AURKC AURKA CDK12 ABT1 AFAP1L2 CALR BCOR ALG12 ACVRL1 ART1 ART3 CTDP1 ATG12 BCL6 DNMT1 DBR1 CHUK CHST4 CHST5 CHST1 BPTF CHST7 CUX1 CITED1 CITED2 DNMT3B ARHGEF10L DNMT3A CREG1 BATF3 CSGALNACT2 CIAO1 CAND1 BTRC CBFB DDX23 CPA2 DDX20 CECR2 AIPL1 DDX54 CTCF CDK2AP1 CPSF6 CPSF3 CARM1 CPSF1 ACVR1 ADAT1 CHD2 CHD1 CHD4 CHD3 ANXA1 CREB5 AARS CD40LG CHEK1 CDC40 CREBBP CDC23 CDC20 CBLC BLZF1 ABCE1 CHMP1A AFF4 CDKL3 CDKL5 ADRM1 CDKL1 CHIT1 B3GNT8 B3GNT5 BTF3 BUD31 ACVR1B CCDC88A CCDC88C BTG2 APEX1 CBY1

R environment session

```
> require(seq2pathway)
```

```
> sessionInfo();
```

```
R version 3.3.1 (2016-06-21)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 16.04.1 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] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] seq2pathway_1.6.0      seq2pathway.data_1.5.0

loaded via a namespace (and not attached):
 [1] Rcpp_0.12.7           XVector_0.14.0        GenomeInfoDb_1.10.0
 [4] RColorBrewer_1.1-2    plyr_1.8.4            zlibbioc_1.20.0
 [7] bitops_1.0-6          iterators_1.0.8       tools_3.3.1
[10] biomaRt_2.30.0        rpart_4.1-10          preprocessCore_1.36.0
[13] RSQLite_1.0.0         gtable_0.2.0          lattice_0.20-34
[16] Matrix_1.2-7.1        foreach_1.4.3         DBI_0.5-1
[19] parallel_3.3.1        WGCNA_1.51            gridExtra_2.2.1
[22] cluster_2.0.5         S4Vectors_0.12.0      IRanges_2.8.0
[25] stats4_3.3.1          grid_3.3.1            nnet_7.3-12
[28] impute_1.48.0         data.table_1.9.6      Biobase_2.34.0
[31] AnnotationDbi_1.36.0  XML_3.98-1.4          survival_2.39-5
[34] foreign_0.8-67        GSA_1.03              latticeExtra_0.6-28
[37] Formula_1.2-1         GO.db_3.4.0           ggplot2_2.1.0
[40] fastcluster_1.1.21    GenomicRanges_1.26.0  Hmisc_3.17-4
[43] scales_0.4.0          codetools_0.2-15      matrixStats_0.51.0
[46] splines_3.3.1         BiocGenerics_0.20.0   dynamicTreeCut_1.63-1
[49] colorspace_1.2-7      acepack_1.3-3.3       RCurl_1.95-4.8
[52] doParallel_1.0.10     munsell_0.4.3         chron_2.3-47
```

References

[1] B. Wang, J. M. Cunningham, X. Yang, *Seq2pathway: an R/Bioconductor package for pathway analysis of next-generation sequencing data*, Bioinformatics **pii: btv289** (2015).

[2] X. Yang, B. Wang, J. M. Cunningham, *Identification of epigenetic modifications that contribute to pathogenesis in therapy-related AML: Effective integration of genome-wide histone modification with transcriptional profiles*, BMC Med Genomics **8S2:S6** (2015), e1002350.

[3] X. Yang, K. Regan, Y. Huang, Q. Zhang, J. Li, T. Y. Seiwert, et al., *Single sample expression-anchored mechanisms predict survival in head and neck cancer*, PLoS Comput Biol **8** (2012), e1002350.

[4] A. Liberzon, A. Subramanian, R. Pinchback, H. Thorvaldsdottir, P. Tamayo, J. P. Mesirov, *Molecular signatures database (MSigDB) 3.0*, Bioinformatics **27** (2011), 1739–1740.

[5] Langfelder P, Horvath S, *WGCNA: an R package for weighted correlation network analysis*, BMC Bioinformatics **9** (2008), 559.

[6] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, et al., *Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles*, Proc Natl Acad Sci USA **102** (2005), 15545–15550.

[7] E. S. Lander, L. M. Linton, B. Birren, C. Nusbaum, M. C. Zody, J. Baldwin, et al., *Initial sequencing and analysis of the human genome*, Nature **409** (2001), 860–921.

[8] Hoffmann A, Yang X, Burnicka-Turek O, Bosman J, Ren X, Hanson E, et al., *Foxf genes integrate Tbx5 and Hedgehog pathways in the second heart field for atrial septation*, PLoS Genetics (2014), DOI: 10.1371/journal.pgen.1004604.

[9] van den Boogaard M, Smemo S, et al., *Initial sequencing and analysis of the human genome*, J Clin Invest. **124** (2014), 1844–1852.

[10] W. de Laat, D. Duboule, *Topology of mammalian developmental enhancers and their regulatory landscapes*, Nature **502** (2013), 499–506.

[11] N. D. Heintzman, B. Ren, *Finding distal regulatory elements in the human genome*, Current opinion in genetics & development **19** (2009), 541–549.

[12] A. Visel, E. M. Rubin, L. A. Pennacchio, *Genomic views of distant-acting enhancers*, Nature **461** (2009), 199–205.

[13] T. Derrien, R. Johnson, G. Bussotti, A. Tanzer, S. Djebali, H. Tilgner, et al., *The GENCODE v7 catalog of human long noncoding RNAs: analysis of their gene structure, evolution, and expression*, Genome Res **22** (2012), 1775–1789.

[14] J. D. Walton, D. R. Kattan, S. K. Thomas, B. A. Spengler, H. F. Guo, J. L. Biedler, et al., *Characteristics of stem cells from human neuroblastoma cell lines and in tumors*, Neoplasia **6** (2004), 838–845.

[15] C. Lottaz, X. Yang, S. Scheid, R. Spang, *OrderedList-a bioconductor package for detecting similarity in ordered gene lists*, Bioinformatics **22** (2006), 2315–2316.

[16] Y. Benjamini, Y. Hochberg, *Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing*, Journal of the Royal Statistical Society. Series B (Methodological) **57** (1995), 289–300.

[17] S. Durinck, P. T. Spellman, E. Birney, W. Huber, *Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt*, Nat Protoc **4** (2009), 1184–1191.

[18] S. Durinck, Y. Moreau, A. Kasprzyk, S. Davis, B. De Moor, A. Brazma, et al., *BioMart and Bioconductor: a powerful link between biological databases and microarray data analysis*, Bioinformatics **21** (2005), 3439–3440.

[19] A. Perez-Rathke, H. Li, Y. Lussier, *Interpreting personal transcriptomes: personalized mechanismscale profiling of RNAseq data*, Pac Symp Biocomput. (2013) 159–170.