

chipenrich: A package for gene set enrichment testing of genome region data

R.P. Welch, C. Lee, R.G. Cavalcante, R.A. Smith, P. Imbriano,
S. Patil, T. Weymouth, L.J. Scott, M.A. Sartor

May 13, 2015

Contents

1	Introduction	1
2	Synopsis	1
3	Locus definitions	3
4	Mappability	4
4.1	Base pair mappability	4
4.2	Locus mappability	4
5	Examples	4
5.1	Diagnostic plots	4
5.2	ChIP-Enrich	4
5.3	ChIP-Enrich with mappability	9
5.4	Broad-Enrich	9
5.5	Fisher's exact test	10
6	Output	10
6.1	Peak-to-gene assignments	11
6.2	Gene set enrichment test results	11
7	References	12

1 Introduction

This document describes how to use **chipenrich** to analyze results from ChIP-Seq experiments and other DNA sequencing experiments that result in a set of genomic regions. **chipenrich** includes two methods that adjust for potential confounders of gene set enrichment testing (locus length and mappability of the sequence reads). The first method **chipenrich** is designed for use with transcription-factor based ChIP-seq experiments and other DNA sequencing experiments with narrow genomic regions. The second method **broadenrich** is designed for use with histone modification based ChIP-seq experiments and other DNA sequencing experiments with broad genomic regions.

2 Synopsis

After starting R, the package should be loaded using the following:

```
> library(chipenrich)
```

This will load **chipenrich**, the **chipenrich.data** package, as well as all other dependency packages. The main function for conducting all gene set enrichment testing is **chipenrich()**.

The defaults for the **chipenrich()** function are

```
chipenrich(peaks, out_name = "chipenrich", out_path = getwd(), genome = "hg19",
genesets = c('GOBP', 'GOCC', 'GOMF'), locusdef = "nearest_tss ",
method = "chipenrich", fisher_alt = "two.sided", use_mappability = F,
read_length = 24, genome_length = NULL, qc_plots = T, n_cores=1)
```

The “peaks” option should be either a data frame or character vector representing the path to a file containing the peaks. The file (or data frame) should have at least 3 columns: “chrom,” “start,” and “end,” denoting the chromosome, starting position, and ending position of the peaks. Chromosome should be in UCSC format, e.g. chrX, chrY, chr22, etc. If a file, it must be tab-delimited, and the header must exist. The input file may also be a .bed, .broadPeak, or .narrowPeak file. Additional columns can exist, so long as they do not contain tab characters. Two example datasets, “peaks_E2F4” and “peaks_H3K4me3_GM12878”, are included in the package.

```
> data(peaks_E2F4)
> data(peaks_H3K4me3_GM12878)
```

```
> data(peaks_E2F4)
> head(peaks_E2F4)
```

	chrom	start	end
1	chr1	156186314	156186469
2	chr1	10490456	10490550
3	chr1	46713352	46713436
4	chr1	226496843	226496924
5	chr1	200589825	200589928
6	chr1	47779789	47779907

The first task of chipenrich() is to assign the peaks to genes. Currently supported genomes are listed below. Data from older genome versions may be converted using UCSC’s liftover tool: <http://genome.ucsc.edu/cgi-bin/hgLiftOver>.

```
> supported_genomes()

[1] "dm3" "hg19" "mm10" "mm9" "rn4"
```

Peaks are assigned to genes according to a pre-defined locus definition, i.e. the region where peaks have to occur in order to be assigned to a gene. The following locus definitions are supported in chipenrich:

```
> supported_locusdefs()

[1] "10kb" "10kb_and_more_upstream"
[3] "1kb" "5kb"
[5] "exon" "intron"
[7] "nearest_gene" "nearest_tss"
```

Using the options “1kb”, “5kb”, or “10kb” will only assign peaks to genes if the peaks are within 1 kilobases (kb), 5kb, or 10kb of a gene’s transcription start site (TSS), respectively. The option “exon” or “intron” will assign peaks to genes if the peaks occur within a gene’s exons or introns, respectively. The option “10kb_and_more_upstream” will assign peaks to genes if the peaks occur in a region more than 10kb upstream from a TSS to the midpoint between the adjacent TSS. Using “nearest_gene” or “nearest_tss” will assign peaks to genes according to the nearest gene or the nearest TSS. Only the “nearest_gene” and “nearest_tss” locus definitions retain all peaks, others use only a subset of peaks that fall within the defined region. All gene loci are non-overlapping. The command “help(chipenrich.data)” may also provide more information on the locus definitions. Users may also create their own custom locus definitions.

Many gene set annotation databases are supported by chipenrich:

```
> supported_genesets()
```

```

[1] "GOBP"                "GOCC"
[3] "GOMF"                "biocarta_pathway"
[5] "cytoband"            "drug_bank"
[7] "ehmn_pathway_gene"    "gene_expression"
[9] "kegg_pathway"         "mesh"
[11] "metabolite"          "mirbase"
[13] "panther_pathway"      "pfam"
[15] "protein_interaction_mimi" "reactome"
[17] "transcription_factors"

```

Three methods for gene set enrichment testing are provided: the main ChIP-Enrich method (“chipenrich”), the Broad-Enrich method (“broadenrich”), and Fisher’s exact test (“fet”). The “chipenrich” method is designed for datasets with narrow genomic regions such as transcription factor ChIP-seq peaks. The “broadenrich” method is designed for datasets with broad genomic regions such as histone modification ChIP-seq peaks. Finally, the “fet” method is also offered for comparison purposes and/or for use in limited situations when its assumptions are met (see examples).

```
> supported_methods()
```

```
[1] "chipenrich" "fet"        "broadenrich"
```

The default gene set database is Gene Ontology (GO) terms, comprising of all three GO branches (GOBP, GOCC, and GOMF).

Accounting for mappability of reads is optional and can only be accomplished using the ChIP-Enrich or Broad-Enrich method. Mappabilities for the following read lengths are available:

```
> supported_read_lengths()
```

```
[1] 24 36 40 50 75 100
```

Read lengths of 24mer are only available for the hg19 genome.

See the section on mappability for more information on how it is calculated.

3 Locus definitions

We define a gene *locus* as the region from which we predict a gene could be regulated. ChIP-seq peaks, or other types of genomic regions, falling within a locus for a gene are assigned to that gene.

We provide a number of different definitions of a gene locus:

nearest_tss The locus is the region spanning the midpoints between the TSSs of adjacent genes.

nearest_gene The locus is the region spanning the midpoints between the boundaries of each gene, where a gene is defined as the region between the furthest upstream TSS and furthest downstream TES for that gene. If two gene loci overlap each other, we take the midpoint of the overlap as the boundary between the two loci. When a gene locus is completely nested within another, we create a disjoint set of 3 intervals, where the outermost gene is separated into 2 intervals broken apart at the endpoints of the nested gene.

1kb The locus is the region within 1 kb of any of the TSSs belonging to a gene. If TSSs from two adjacent genes are within 2 kb of each other, we use the midpoint between the two TSSs as the boundary for the locus for each gene.

5kb The locus is the region within 5 kb of any of the TSSs belonging to a gene. If TSSs from two adjacent genes are within 10 kb of each other, we use the midpoint between the two TSSs as the boundary for the locus for each gene.

10kb The locus is the region within 10 kb of any of the TSSs belonging to a gene. If TSSs from two adjacent genes are within 20 kb of each other, we use the midpoint between the two TSSs as the boundary for the locus for each gene.

10kb_and_more_upstream The locus is the region more than 10kb upstream from a TSS to the midpoint between the adjacent TSS.

exon Each gene has multiple loci corresponding to its exons.

intron Each gene has multiple loci corresponding to its introns.

4 Mappability

4.1 Base pair mappability

We define base pair mappability as the average read mappability of all possible reads of size K that encompass a specific base pair location, b . Mappability files from UCSC Genome Browser mappability track were used to calculate base pair mappability. The mappability track provides values for theoretical read mappability, or the number of places in the genome that could be mapped by a read that begins with the base pair location b . For example, a value of 1 indicates a Kmer read beginning at b is mappable to one area in the genome. A value of 0.5 indicates a Kmer read beginning at b is mappable to two areas in the genome. For our purposes, we are only interested in uniquely mappable reads; therefore, all reads with mappability less than 1 were set to 0 to indicate non-unique mappability. Then, base pair mappability is calculated as:

$$M_i = \left(\frac{1}{2K-1}\right) \sum_{j=i-K+1}^{i+(K-1)} M_j \quad (1)$$

where M_i is the mappability of base pair i , and M_j is mappability (from UCSC's mappability track) of read j where j is the start position of the K length read. We calculated base pair mappability for reads of lengths 24, 36, 40, 50, 75, and 100 base pairs for *Homo sapiens* (build hg19) and for reads of lengths 36, 40, 50, 75, and 100 base pairs for *Mus musculus* (build mm9). Mappability is unavailable for *Rattus norvegicus* (build rn4) and (*Mus musculus* (build mm10)).

4.2 Locus mappability

We define locus mappability as the average of all base pair mappability values for a gene's locus. Locus mappability is calculated for each available locus definition.

5 Examples

5.1 Diagnostic plots

If "method = chipenrich" and "qc_plots = T" then two pdf files will be output: One with a binomial smoothing spline fitted to the probability of a peak given gene length and one showing the distribution of the distance of peaks to the nearest TSS of a gene. These plots may also be generated using separate functions as illustrated below. Figure 1 shows the distribution of peaks to the nearest TSS. In figure 2, a spline is fitted to the data given gene locus length. The same is shown in figure 3 but here we account for the mappable locus length ($mappability \times locuslength$).

If "method = broadenrich" and "qc_plots = T" then one pdf file is output: A plot showing the relationship between the gene locus length and the proportion of the locus covered by a peak. Figure 4 shows this relationship.

There are many combinations of methods, genesets, and mappability settings that may be used to do gene set enrichment testing using **chipenrich**. In the following, we include some examples of gene set enrichment testing using the peaks_E2F4 and peaks_H3K4me3_GM12878 example datasets. Note that use of multiple cores (**n_cores** option) is not available on Windows.

5.2 ChIP-Enrich

Gene set enrichment of Biocarta pathways using ChIP-Enrich:

```
> plot_dist_to_tss(peaks = peaks_E2F4, genome = 'hg19')
```

Distribution of Distance from Peaks to Nearest TSS

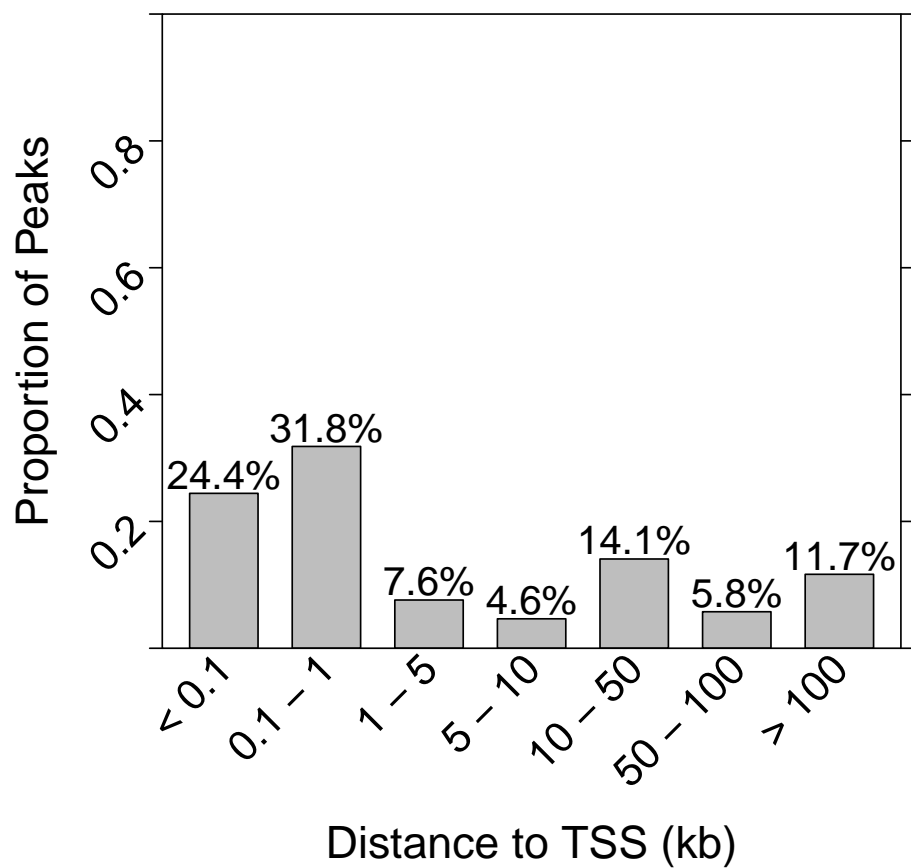


Figure 1: Distribution of distance from each peak in the dataset to the nearest TSS. All peaks are considered in creating this plot, regardless of locus definition.

```
> plot_spline_length(peaks = peaks_E2F4, locusdef = 'nearest_tss', genome = 'hg19')
```

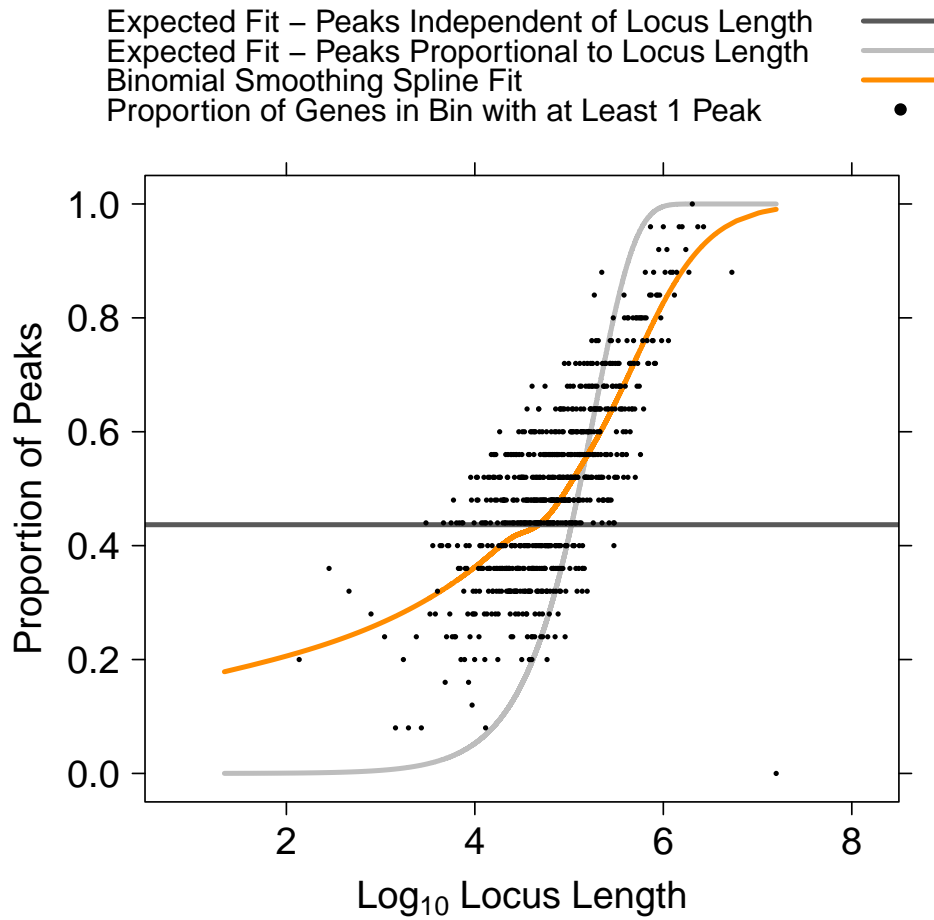


Figure 2: The relationship between the probability of a gene being assigned a peak and locus length. In the E2F4 dataset, we can see such a relationship exists (orange), and does not match either the assumption that each gene has the same probability of a peak (dark grey horizontal line), or that the probability is proportional to locus length (light grey curve.)

```
> plot_spline_length(peaks = peaks_E2F4, locusdef = 'nearest_tss', genome = 'hg19',
use_mappability = T, read_length = 24)
```

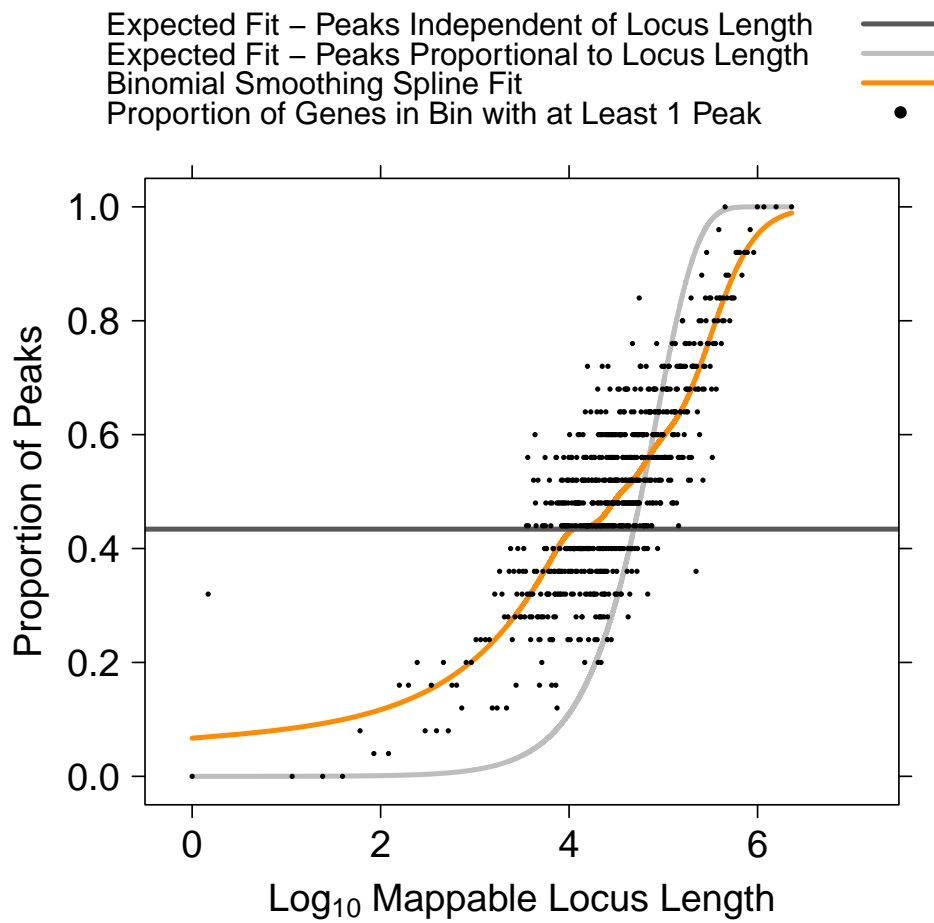


Figure 3: Spline fit using the mappable locus length as opposed to simply the locus length of each gene. In many cases, using mappability improves the spline fit, such as for those long locus length genes with poor mappability (duplicated and/or pseudo-genes.)

```
> plot_gene_coverage(peaks = peaks_H3K4me3_GM12878, locusdef = 'nearest_tss', genome = 'hg19')
```

Binned Locus Length versus Peak Coverage

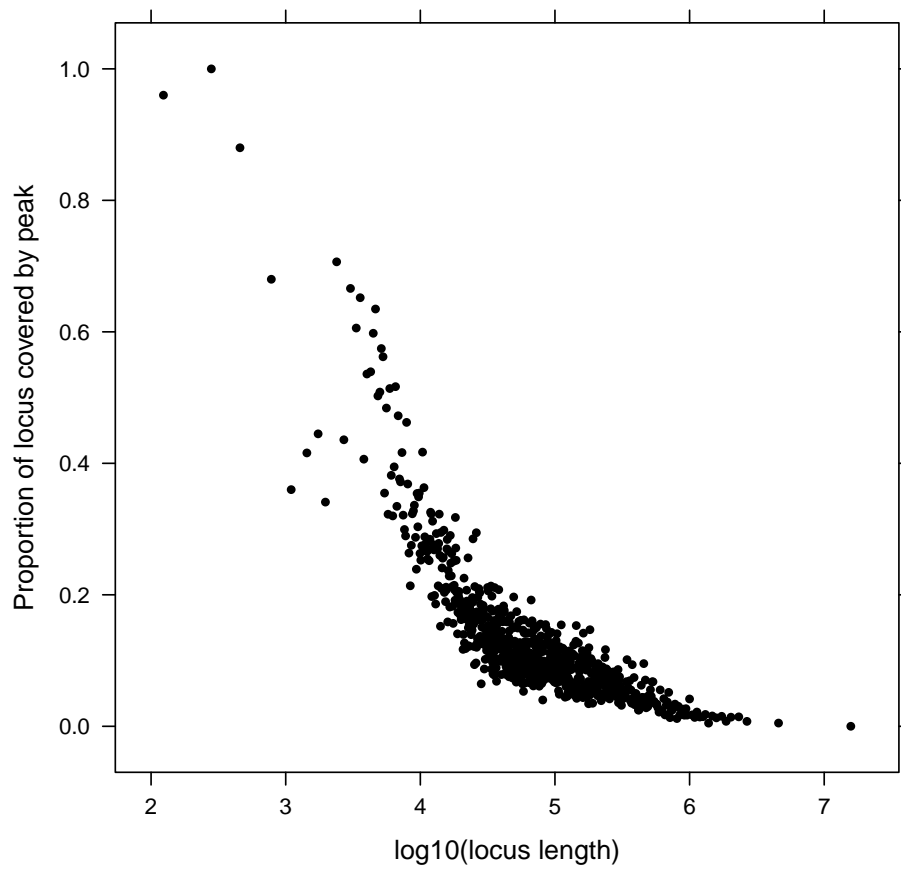


Figure 4: The relationship between the locus length and the proportion of the locus covered by a peak.


```
> results = chipenrich(peaks = peaks_E2F4, genesets = "biocarta_pathway",
+ locusdef = "nearest_tss", max_geneset_size = 100, qc_plots = F,
+ out_name = NULL, n_cores=2)
> results.ce = results$results[order(results$results$P.value),]
> results.ce[1:5,1:5]
```

	Geneset.Type	Geneset.ID	
goterm179	Biocarta Pathway	BC_80	
goterm177	Biocarta Pathway	BC_77	
goterm47	Biocarta Pathway	BC_164	
goterm176	Biocarta Pathway	BC_76	
goterm160	Biocarta Pathway	BC_56	
			Description
goterm179			Dendritic cells in regulating TH1 and TH2 Development
goterm177			Cytokines and Inflammatory Response
goterm47			MAPKinase Signaling Pathway
goterm176			Cytokine Network
goterm160			Cell Cycle: G2/M Checkpoint
	P.value	FDR	
goterm179	0.0004601544	0.08675509	
goterm177	0.0011033837	0.08675509	
goterm47	0.0013485248	0.08675509	
goterm176	0.0019271089	0.09298300	
goterm160	0.0033401493	0.11045172	

5.3 ChIP-Enrich with mappability

Gene set enrichment of Biocarta pathways using ChIP-Enrich, accounting for mappability:

```
> results = chipenrich(peaks = peaks_E2F4, genesets = "biocarta_pathway",
+ locusdef = "nearest_tss", max_geneset_size = 100, use_mappability=T,
+ read_length=24, qc_plots = F, out_name = NULL, n_cores=2)
> results.cem = results$results[order(results$results$P.value),]
> results.cem[1:5,1:5]
```

	Geneset.Type	Geneset.ID	
goterm179	Biocarta Pathway	BC_80	
goterm177	Biocarta Pathway	BC_77	
goterm47	Biocarta Pathway	BC_164	
goterm176	Biocarta Pathway	BC_76	
goterm160	Biocarta Pathway	BC_56	
			Description
goterm179			Dendritic cells in regulating TH1 and TH2 Development
goterm177			Cytokines and Inflammatory Response
goterm47			MAPKinase Signaling Pathway
goterm176			Cytokine Network
goterm160			Cell Cycle: G2/M Checkpoint
	P.value	FDR	
goterm179	0.0004093013	0.07899515	
goterm177	0.0010673953	0.08095247	
goterm47	0.0016763775	0.08095247	
goterm176	0.0016777713	0.08095247	
goterm160	0.0030869101	0.09366777	

5.4 Broad-Enrich

Gene set enrichment of Biocarta pathways using Broad-Enrich:

```
> results = chipenrich(peaks = peaks_H3K4me3_GM12878, genesets = "biocarta_pathway",
+ method='broadenrich', locusdef = "nearest_tss", max_geneset_size = 100,
```

```
+      qc_plots = F, out_name = NULL, n_cores=2)
> results.be = results$results[order(results$results$P.value),]
> results.be[1:5,1:5]
```

	Geneset.Type	Geneset.ID			
ratio40	Biocarta Pathway	BC_152			
ratio168	Biocarta Pathway	BC_65			
ratio171	Biocarta Pathway	BC_7			
ratio167	Biocarta Pathway	BC_64			
ratio1	Biocarta Pathway	BC_100			
			Description		
ratio40	Intrinsic Prothrombin Activation Pathway				
ratio168	Complement Pathway				
ratio171	Acute Myocardial Infarction				
ratio167	Classical Complement Pathway				
ratio1	Extrinsic Prothrombin Activation Pathway				
	P.value	FDR			
ratio40	0.0006564768	0.1168791			
ratio168	0.0012111819	0.1168791			
ratio171	0.0019843759	0.1276615			
ratio167	0.0040397125	0.1559275			
ratio1	0.0052068819	0.1559275			

5.5 Fisher's exact test

Fisher's Exact test assumes that each gene is equally likely to have a peak. We recommend using Fisher's exact test with the "1kb" or "5kb" locus definitions only. This will force all genes to have approximately the same locus length and avoid returning bias results.

Gene set enrichment of KEGG pathways using Fisher's exact test:

```
> results = chipenrich(peaks = peaks_E2F4, genesets = c("kegg_pathway"),
+      locusdef = "5kb", method = "fet", fisher_alt = "two.sided",
+      max_geneset_size = 100, qc_plots = F, out_name = NULL)
> results.fet = results$results[order(results$results$P.value),]
> results.fet[1:5,1:5]
```

	Geneset.Type	Geneset.ID			
102	KEGG Pathway	path:hsa01430			
98	KEGG Pathway	path:hsa00980			
44	KEGG Pathway	path:hsa00510			
166	KEGG Pathway	path:hsa05310			
167	KEGG Pathway	path:hsa05320			
			Description		
102			Cell Communication		
98	Metabolism of xenobiotics by cytochrome P450				
44	N-Glycan biosynthesis				
166	Asthma				
167	Autoimmune thyroid disease				
	P.value	FDR			
102	8.529743e-08	9.278250e-06			
98	1.111168e-07	9.278250e-06			
44	3.073858e-07	1.711114e-05			
166	2.582753e-06	1.078299e-04			
167	9.507701e-06	2.930124e-04			

6 Output

The output of `chipenrich()` is an R object containing the results of the test and the peak to gene assignments. Both of these are also written to text files in the working directory (unless specified

otherwise) after the test is completed.

6.1 Peak-to-gene assignments

Peak assignments are stored in `$peaks`. The following is an example of how to access the peak to gene assignments in R after a gene set enrichment test has already been performed:

```
> peaks_to_genes = results$peaks
> head(peaks_to_genes)
```

	peak_id	chrom	peak_start	peak_end	peak_midpoint
1	827	chr1	3817811	3817975	3817893
2	403	chr1	3817484	3817622	3817553
3	130	chr1	77685073	77685175	77685124
4	337	chr1	29562944	29563102	29563023
5	129	chr1	183604939	183605051	183604995
6	375	chr1	180992048	180992155	180992101

	geneid	gene_symbol	gene_locus_start	gene_locus_end
1	100133612	L0C100133612	3816912	3821967
2	100133612	L0C100133612	3816912	3821967
3	10026	PIGK	77680132	77690132
4	10076	PTPRU	29560250	29568046
5	10092	ARPC5	183599985	183605095
6	10228	STX6	180987046	180997046

	nearest_tss	dist_to_tss	nearest_tss_gene
1	3816967	925	100133612
2	3816967	585	100133612
3	77685132	7	10026
4	29563027	-3	10076
5	183605076	80	10092
6	180992046	-54	10228

	nearest_tss_gene_strand
1	+
2	+
3	-
4	+
5	-
6	-

6.2 Gene set enrichment test results

The results of a `chipenrich()` R object is stored in `$results`. It contains 10 columns when enrichment is done with ChIP-Enrich and 11 columns with Broad-Enrich:

```
> colnames(results$results)
```

[1]	"Geneset.Type"	"Geneset.ID"
[3]	"Description"	"P.value"
[5]	"FDR"	"Odds.Ratio"
[7]	"Status"	"N.Geneset.Genes"
[9]	"N.Geneset.Peak.Genes"	"Geneset.Peak.Genes"

Geneset.Type specifies from which database the “Geneset.ID” originates. For example, “Gene Ontology Biological Process.”

Geneset.ID is the identifier for a given gene set from the selected database. For example, “GO:0000003.”

Description gives a definition of the “Geneset.ID.” For example, “reproduction.”

P.Value is the probability of observing the degree of enrichment (see “Odds.Ratio”) of the gene set given the null hypothesis that peaks are not associated with any gene sets.

FDR is the false discovery rate proposed by Benjamini & Hochberg for adjusting the p-value to control for family-wise error rate.

Odds.Ratio is the estimated odds that peaks are associated with a given gene set compared to the odds that peaks are associated with other gene sets, after controlling for locus length and/or mappability. An odds ratio greater than 1 indicates enrichment, and less than 1 indicates depletion.

Status denotes whether the gene set was enriched or depleted.

N.Geneset.Genes is the number of genes in the gene set.

N.Geneset.Peak.Genes is the number of genes in the genes set that were assigned at least one peak.

Geneset.Avg.Gene.Coverage (broadenrich only) is the mean proportion of the gene loci in the gene set covered by a peak.

Geneset.Peak.Genes is the list of genes from the gene set that had at least one peak assigned.

7 References

- R.P. Welch*, C. Lee*, R.A. Smith, P. Imbriano, S. Patil, T. Weymouth, L.J. Scott, M.A. Sartor. "ChIP-Enrich: gene set enrichment testing for ChIP-seq data." Nucl. Acids Res. (2014) 42(13):e105. doi:10.1093/nar/gku463
- R.G. Cavalcante, C. Lee, R.P. Welch, S. Patil, T. Weymouth, L.J. Scott, and M.A. Sartor. "Broad-Enrich: functional interpretation of large sets of broad genomic regions." Bioinformatics (2014) 30(17):i393-i400 doi:10.1093/bioinformatics/btu444