

ChIPseeker: an R package for ChIP peak Annotation, Comparison and Visualization

Guangchuang Yu

School of Public Health
The University of Hong Kong
guangchuangyu@gmail.com

August 26, 2015

Abstract

ChIPseeker is an R package for annotating ChIP-seq data analysis. It supports annotating ChIP peaks and provides functions to visualize ChIP peaks coverage over chromosomes and profiles of peaks binding to TSS regions. Comparison of ChIP peak profiles and annotation are also supported. Moreover, it supports evaluating significant overlap among ChIP-seq datasets. Currently, ChIPseeker contains 17,000 bed file information from GEO database. These datasets can be downloaded and compare with user's own data to explore significant overlap datasets for inferring co-regulation or transcription factor complex for further investigation.

ChIPseeker version: 1.4.7

If you use *ChIPseeker* in published research, please cite:

G Yu, LG Wang, QY He. **ChIPseeker: an R/Bioconductor package for ChIP peak annotation, comparison and visualization.**

Bioinformatics 2015. 31(14): 2382-2383.

<http://dx.doi.org/10.1093/bioinformatics/btv145>

Contents

1	Introduction	3
2	ChIP profiling	3
2.1	ChIP peaks coverage plot	4
2.2	Profile of ChIP peaks binding to TSS regions	6
2.2.1	Heatmap of ChIP binding to TSS regions	6
2.2.2	Average Profile of ChIP peaks binding to TSS region	7
3	Peak Annotation	7
3.1	Visualize Genomic Annotation	9
3.2	Visualize distribution of TF-binding loci relative to TSS	9
4	Functional enrichment analysis	10
5	ChIP peak data set comparison	12
5.1	Profile of several ChIP peak data binding to TSS region	12
5.1.1	Average profiles	12
5.1.2	Peak heatmaps	12
5.2	ChIP peak annotation comparision	13
5.3	Functional profiles comparison	13
5.4	Overlap of peaks and annotated genes	13
6	Statistical testing of ChIP seq overlap	14
6.1	Shuffle genome coordination	14
6.2	Peak overlap enrichment analysis	14
7	Data Mining with ChIP seq data deposited in GEO	15
7.1	GEO data collection	15
7.2	Download GEO ChIP data sets	19
7.3	Overlap significant testing	19
8	External documents	19
9	Session Information	20

1 Introduction

Chromatin immunoprecipitation followed by high-throughput sequencing (ChIP-seq) has become standard technologies for genome wide identification of DNA-binding protein target sites. After read mappings and peak callings, the peak should be annotated to answer the biological questions. Annotation also create the possibility of integrate expression profile data to predict gene expression regulation. *ChIPseeker* was developed for annotating nearest genes and genomic features to peaks.

ChIP peak data set comparison is also very important. We can use it as an index to estimate how well biological replications are. Even more important is applying to infer cooperative regulation. If two ChIP seq data, obtained by two different binding proteins, overlap significantly, these two proteins may form a complex or have interaction in regulation chromosome remodelling or gene expression. *ChIPseeker* support statistical testing of significant overlap among ChIP seq data sets, and incorporate open access database GEO for users to compare their own dataset to those deposited in database. Protein interaction hypothesis can be generated by mining data deposited in database. Converting genome coordinations from one genome version to another is also supported, making this comparison available for different genome version and different species.

Several visualization functions are implemented to visualize the coverage of the ChIP seq data, peak annotation, average profile and heatmap of peaks binding to TSS region.

Functional enrichment analysis of the peaks can be performed by my Bioconductor packages *DOSE* [1], *ReactomePA*, *clusterProfiler* [2] .

```
## loading packages
require(ChIPseeker)
require(TxDb.Hsapiens.UCSC.hg19.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene
require(clusterProfiler)
```

2 ChIP profiling

The datasets CBX6 and CBX7 in this vignettes were downloaded from GEO (GSE40740) [3] while ARmo_0M, ARmo_1nM and ARmo_100nM were downloaded from GEO (GSE48308) [4] . *ChIPseeker* provides readPeakFile to load the peak and store in GRanges object.

```
files <- getSampleFiles()
print(files)

## $ARmo_0M
## [1] "/private/tmp/RtmpSo4ab7/Rinst68c840deaa86/ChIPseeker/extdata/GEO_sample_data/GSM117
##
## $ARmo_1nM
## [1] "/private/tmp/RtmpSo4ab7/Rinst68c840deaa86/ChIPseeker/extdata/GEO_sample_data/GSM117
```

```
##
## $ARmo_100nM
## [1] "/private/tmp/RtmpSo4ab7/Rinst68c840deaa86/ChIPseeker/extdata/GEO_sample_data/GSM117
##
## $CBX6_BF
## [1] "/private/tmp/RtmpSo4ab7/Rinst68c840deaa86/ChIPseeker/extdata/GEO_sample_data/GSM129
##
## $CBX7_BF
## [1] "/private/tmp/RtmpSo4ab7/Rinst68c840deaa86/ChIPseeker/extdata/GEO_sample_data/GSM129

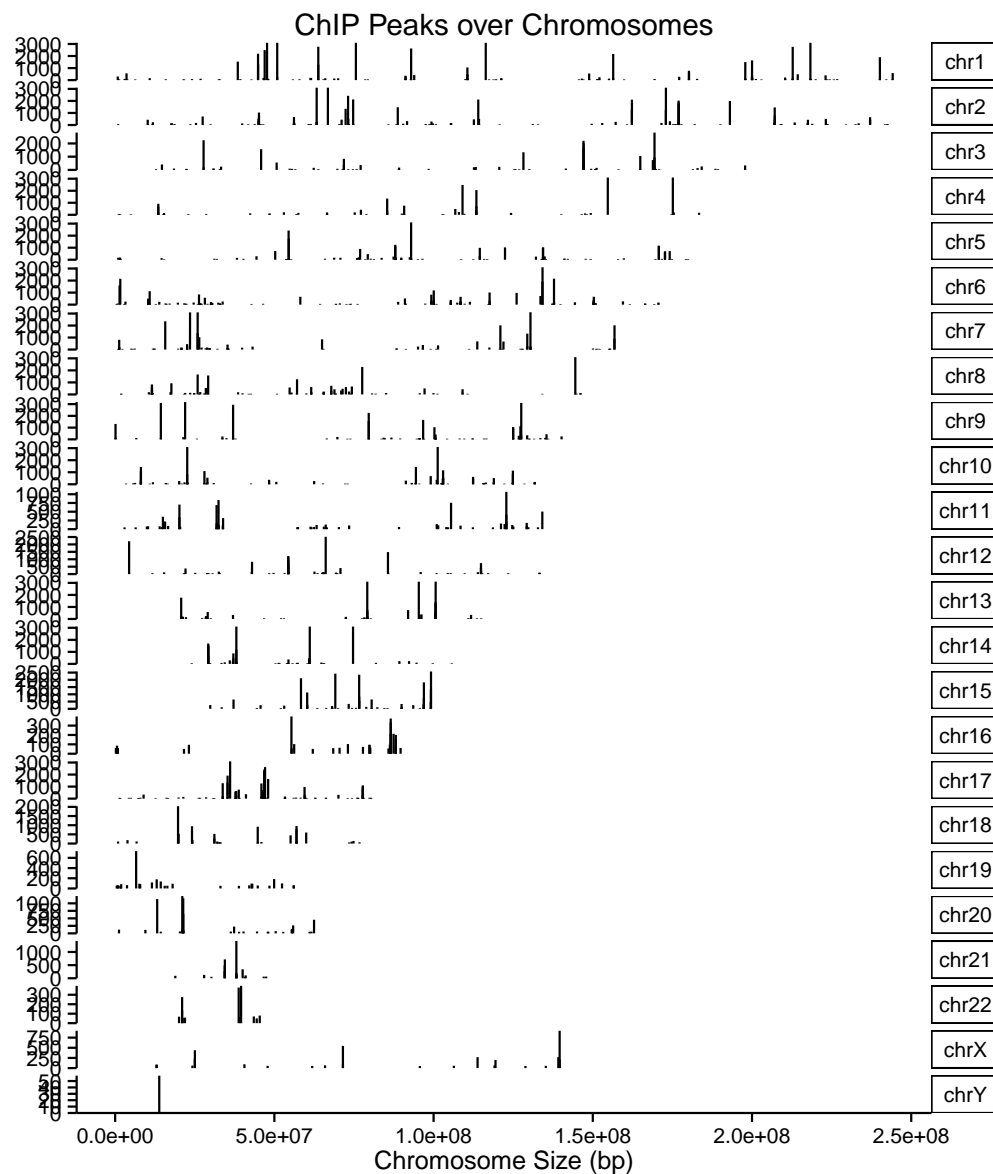
peak <- readPeakFile(files[[4]])
peak

## GRanges object with 1331 ranges and 2 metadata columns:
##           seqnames           ranges strand |           V4           V5
##           <Rle>             <IRanges> <Rle> | <factor> <numeric>
##      [1]      chr1      [ 815093,   817883]      * |   MACS_peak_1      295.8
##      [2]      chr1      [1243288, 1244338]      * |   MACS_peak_2       63.2
##      [3]      chr1      [2979977, 2981228]      * |   MACS_peak_3     100.2
##      [4]      chr1      [3566182, 3567876]      * |   MACS_peak_4     558.9
##      [5]      chr1      [3816546, 3818111]      * |   MACS_peak_5       57.6
##      ...      ...      ...      ...      ...      ...
## [1327]     chrX [135244783, 135245821]      * | MACS_peak_1327      55.5
## [1328]     chrX [139171964, 139173506]      * | MACS_peak_1328     270.2
## [1329]     chrX [139583954, 139586126]      * | MACS_peak_1329     918.7
## [1330]     chrX [139592002, 139593238]      * | MACS_peak_1330     210.9
## [1331]     chrY [ 13845134,  13845777]      * | MACS_peak_1331       58.4
## -----
## seqinfo: 24 sequences from an unspecified genome; no seqlengths
```

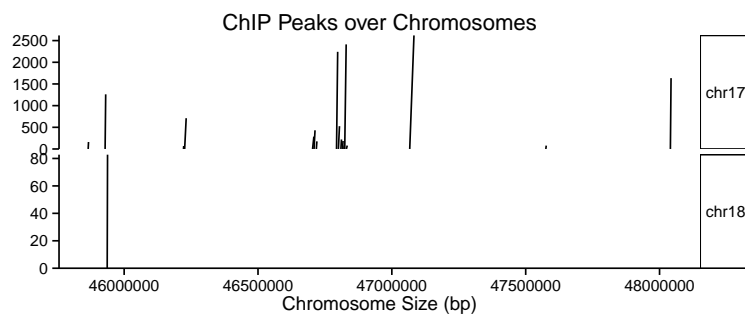
2.1 ChIP peaks coverage plot

After peak calling, we would like to know the peak locations over the whole genome, `covplot` function calculates the coverage of peak regions over chromosomes and generate a figure to visualize.

```
covplot(peak, weightCol="V5")
```



```
covplot(peak, weightCol="V5", chrs=c("chr17", "chr18"), xlim=c(4.5e7, 5e7))
```



2.2 Profile of ChIP peaks binding to TSS regions

First of all, for calculate the profile of ChIP peaks binding to TSS regions, we should prepare the TSS regions, which are defined as the flanking sequence of the TSS sites. Then align the peaks that are mapping to these regions, and generate the tagMatrix.

```
## promoter <- getPromoters(TxDb=txdb, upstream=3000, downstream=3000)
## tagMatrix <- getTagMatrix(peak, windows=promoter)
##
## to speed up the compilation of this vignettes, we use a precalculated tagMatrix
data("tagMatrixList")
tagMatrix <- tagMatrixList[[4]]
```

In the above code, you should notice that tagMatrix is not restricted to TSS regions. The regions can be other types that defined by the user.

2.2.1 Heatmap of ChIP binding to TSS regions

```
tagHeatmap(tagMatrix, xlim=c(-3000, 3000), color="red")
```



Figure 1: Heatmap of ChIP peaks binding to TSS regions

[ChIPseeker](#) provide a one step function to generate this figure from bed file. The following function will generate the same figure as above.

```
peakHeatmap(files[[4]], TxDb=txdb, upstream=3000, downstream=3000, color="red")
```

2.2.2 Average Profile of ChIP peaks binding to TSS region

```
plotAvgProf(tagMatrix, xlim=c(-3000, 3000),
            xlab="Genomic Region (5'→3')", ylab = "Read Count Frequency")
```

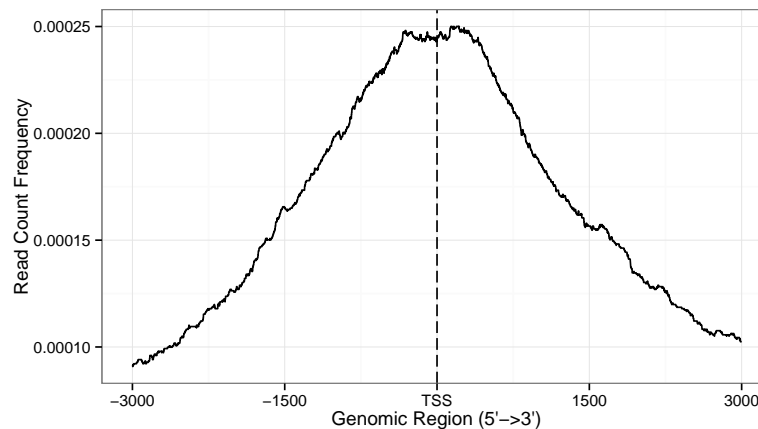


Figure 2: Average Profile of ChIP peaks binding to TSS region

The function `plotAvgProf2` provide a one step from bed file to average profile plot. The following command will generate the same figure as shown above.

```
plotAvgProf2(files[[4]], TxDb=txdb, upstream=3000, downstream=3000,
             xlab="Genomic Region (5'→3')", ylab = "Read Count Frequency")
```

Confidence interval estimated by bootstrap method is also supported for characterizing ChIP binding profiles.

```
plotAvgProf(tagMatrix, xlim=c(-3000, 3000), conf = 0.95, resample = 500)
```

3 Peak Annotation

```
peakAnno <- annotatePeak(files[[4]], tssRegion=c(-3000, 3000),
                        TxDb=txdb, annoDb="org.Hs.eg.db")

## >> loading peak file... 2015-08-26 21:26:15
## >> preparing features information... 2015-08-26 21:26:15
## >> identifying nearest features... 2015-08-26 21:26:15
## >> calculating distance from peak to TSS... 2015-08-26 21:26:15
## >> assigning genomic annotation... 2015-08-26 21:26:15
```

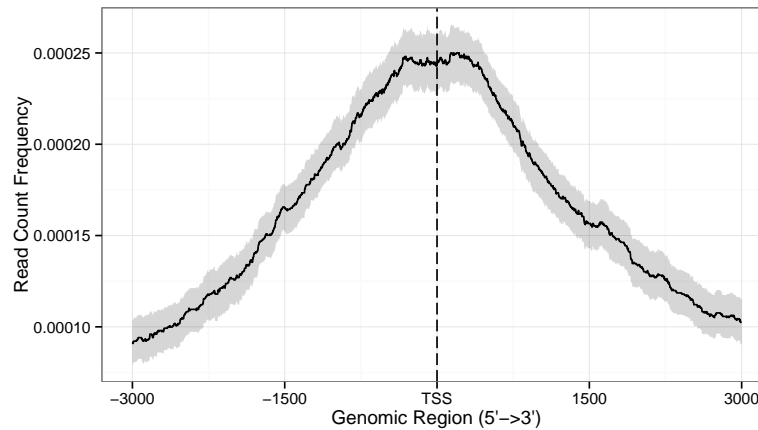


Figure 3: Average Profile of ChIP peaks binding to TSS region

```
## >> adding gene annotation... 2015-08-26 21:26:40
## >> assigning chromosome lengths 2015-08-26 21:26:41
## >> done... 2015-08-26 21:26:41
```

Peak Annotation is performed by `annotatePeak`. User can define TSS (transcription start site) region, by default TSS is defined from -3kb to +3kb. The output of `annotatePeak` is `csAnno` instance. *ChIPseeker* provides `as.GRanges` to convert `csAnno` to `GRanges` instance, and `as.data.frame` to convert `csAnno` to `data.frame` which can be exported to file by `write.table`.

`TxDb` object contained transcript-related features of a particular genome. Bioconductor provides several package that containing `TxDb` object of model organisms with multiple commonly used genome version, for instance *[TxDb.Hsapiens.UCSC.hg38.knownGene](#)*, *[TxDb.Hsapiens.UCSC.hg19.knownGene](#)* for human genome hg38 and hg19, *[TxDb.Mmusculus.UCSC.mm10.knownGene](#)* and

[TxDb.Mmusculus.UCSC.mm9.knownGene](#) for mouse genome mm10 and mm9, etc. User can also prepare their own `TxDb` object by retrieving information from UCSC Genome Bioinformatics and BioMart data resources by R function

`makeTranscriptDbFromBiomart` and `makeTranscriptDbFromUCSC`. `TxDb` object should be passed for peak annotation.

All the peak information contained in peakfile will be retained in the output of `annotatePeak`. The position and strand information of nearest genes are reported. The distance from peak to the TSS of its nearest gene is also reported. The genomic region of the peak is reported in annotation column. Since some annotation may overlap, *ChIPseeker* adopted the following priority in genomic annotation.

- Promoter
- 5' UTR
- 3' UTR
- Exon
- Intron

- Downstream
- Intergenic

Downstream is defined as the downstream of gene end.

ChIPseeker also provides parameter `genomicAnnotationPriority` for user to prioritize this hierarchy.

`annotatePeak` report detail information when the annotation is Exon or Intron, for instance "Exon (uc002sbe.3/9736, exon 69 of 80)", means that the peak is overlap with an Exon of transcript uc002sbe.3, and the corresponding Entrez gene ID is 9736 (Transcripts that belong to the same gene ID may differ in splice events), and this overlapped exon is the 69th exon of the 80 exons that this transcript uc002sbe.3 possess.

Parameter `annoDb` is optional, if provided, extra columns including SYMBOL, GENENAME, ENSEMBL/ENTREZID will be added. The `geneID` column in annotation output will be consistent with the `geneID` in TxDb. If it is ENTREZID, ENSEMBL will be added if `annoDb` is provided, while if it is ENSEMBL ID, ENTREZID will be added.

3.1 Visualize Genomic Annotation

To annotate the location of a given peak in terms of genomic features, `annotatePeak` assigns peaks to genomic annotation in "annotation" column of the output, which includes whether a peak is in the TSS, Exon, 5' UTR, 3' UTR, Intronic or Intergenic. Many researchers are very interesting in these annotations. TSS region can be defined by user and `annotatePeak` output in details of which exon/intron of which genes as illustrated in previous section.

Pie and Bar plot are supported to visualize the genomic annotation.

```
plotAnnoPie(peakAnno)
```

```
plotAnnoBar(peakAnno)
```

Since some annotation overlap, user may interested to view the full annotation with their overlap, which can be partially resolved by `vennpie` function.

```
vennpie(peakAnno)
```

3.2 Visualize distribution of TF-binding loci relative to TSS

The distance from the peak (binding site) to the TSS of the nearest gene is calculated by `annotatePeak` and reported in the output. We provide `plotDistToTSS` to calculate the percentage of binding sites upstream and downstream from the TSS of the nearest genes, and visualize the distribution.

```
plotDistToTSS(peakAnno,
               title="Distribution of transcription factor-binding loci\nrelative to TSS")
```

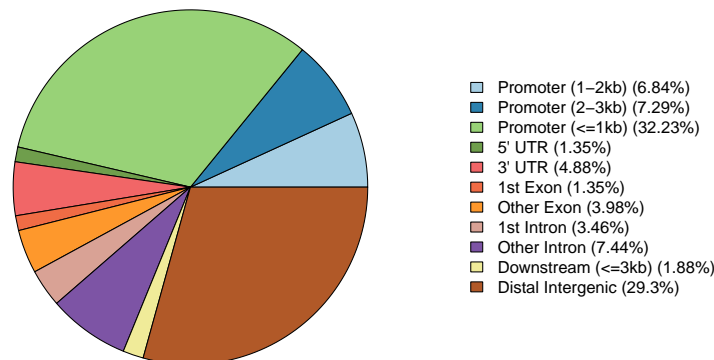


Figure 4: Genomic Annotation by pieplot

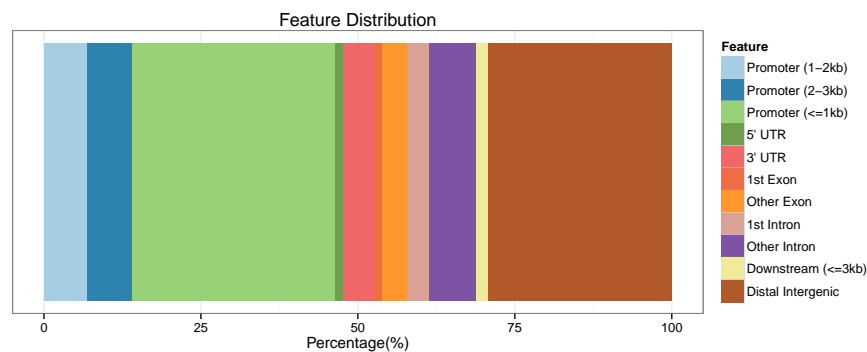


Figure 5: Genomic Annotation by barplot

4 Functional enrichment analysis

Once we have obtained the annotated nearest genes, we can perform functional enrichment analysis to identify predominant biological themes among these genes by incorporating biological knowledge provided by biological ontologies. For instance, Gene Ontology (GO) [5] annotates genes to biological processes, molecular functions, and cellular components in a directed acyclic graph structure, Kyoto Encyclopedia of Genes and Genomes (KEGG) [6] annotates genes to pathways, Disease Ontology (DO) [7] annotates genes with human disease association, and Reactome [8] annotates gene to pathways and reactions.

Enrichment analysis is a widely used approach to identify biological themes. I have developed several Bioconductor packages for investigating whether the number of selected genes associated with a particular biological term is larger than expected, including [DOSE](#) [1] for Disease Ontology, [ReactomePA](#)

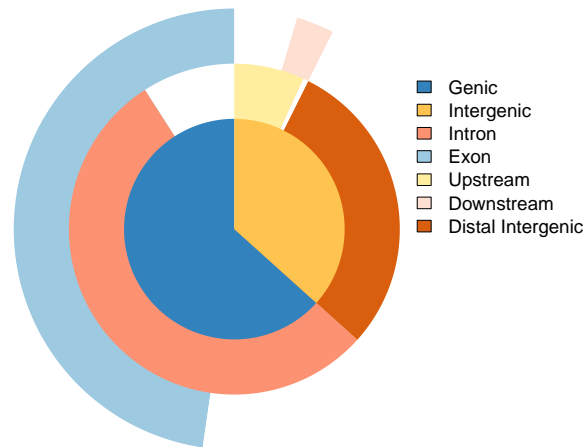


Figure 6: Genomic Annotation by vennpie

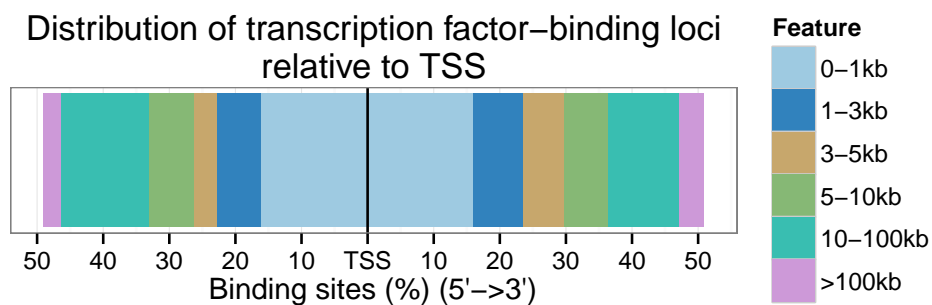


Figure 7: Distribution of Binding Sites

for reactome pathway, [clusterProfiler](#) [?] for Gene Ontology and KEGG enrichment analysis.

```
require(clusterProfiler)
bp <- enrichGO(as.data.frame(peakAnno)$geneId, ont="BP", readable=TRUE)
head(summary(bp), n=3)
```

##	ID	Description	GeneRatio
##	G0:0044767	G0:0044767 single-organism developmental process	450/783
##	G0:0032502	G0:0032502 developmental process	454/783
##	G0:0007275	G0:0007275 multicellular organismal development	408/783

```
##           BgRatio    pvalue p.adjust    qvalue
## G0:0044767 5303/18585 5.49e-67 1.88e-63 1.38e-63
## G0:0032502 5389/18585 7.38e-67 1.88e-63 1.38e-63
## G0:0007275 4545/18585 4.66e-65 7.93e-62 5.83e-62
##
## G0:0044767          SP9/DNM1L/EDIL3/OLIG2/SPRY1/CDKN2A/CCNO/WARS2/KLF2/MER
## G0:0032502 SP9/DNM1L/EDIL3/OLIG2/SPRY1/CDKN2A/CCNO/WARS2/KLF2/MERTK/ZBTB18/HOXB13/IGF2B
## G0:0007275
##           Count
## G0:0044767    450
## G0:0032502    454
## G0:0007275    408
```

More information can be found in the vignettes of Bioconductor packages [DOSE](#) [1], [ReactomePA](#), [clusterProfiler](#) [2], which also provide several methods to visualize enrichment results. The [clusterProfiler](#) [2] is designed for comparing and visualizing functional profiles among gene clusters, and can directly applied to compare biological themes at GO, DO, KEGG, Reactome perspective.

5 ChIP peak data set comparison

5.1 Profile of several ChIP peak data binding to TSS region

Function `plotAvgProf` and `tagHeatmap` can accept a list of `tagMatrix` and visualize profile or heatmap among several ChIP experiments, while `plotAvgProf2` and `peakHeatmap` can accept a list of bed files and perform the same task in one step.

5.1.1 Average profiles

```
## promoter <- getPromoters(TxDb=txdb, upstream=3000, downstream=3000)
## tagMatrixList <- lapply(files, getTagMatrix, windows=promoter)
##
## to speed up the compilation of this vignette, we load a precalculated tagMatrixList
data("tagMatrixList")
plotAvgProf(tagMatrixList, xlim=c(-3000, 3000))

## resample = 500 by default, here use 100 to speed up the compilation of this vignette.
plotAvgProf(tagMatrixList, xlim=c(-3000, 3000), conf=0.95, resample=100, facet="row")
```

5.1.2 Peak heatmaps

```
tagHeatmap(tagMatrixList, xlim=c(-3000, 3000), color=NULL)
```

5.2 ChIP peak annotation comparison

The `plotAnnoBar` and `plotDistToTSS` can also accept input of a named list of annotated peaks (output of `annotatePeak`).

```
peakAnnoList <- lapply(files, annotatePeak, TxDb=txdb,
                       tssRegion=c(-3000, 3000), verbose=FALSE)
```

We can use `plotAnnoBar` to comparing their genomic annotation.

```
plotAnnoBar(peakAnnoList)
```

R function `plotDistToTSS` can use to comparing distance to TSS profiles among ChIPseq data.

```
plotDistToTSS(peakAnnoList)
```

5.3 Functional profiles comparison

As shown in section 4, the annotated genes can analyzed by [clusterProfiler](#) [?], [DOSE](#) and [ReactomePA](#) for Gene Ontology, KEGG, Disease Ontology and Reactome Pathway enrichment analysis.

The [clusterProfiler](#) [?] package provide `compareCluster` function for comparing biological themes among gene clusters, and can be easily adopted to compare different ChIP peak experiments.

```
genes = lapply(peakAnnoList, function(i) as.data.frame(i)$geneId)
names(genes) = sub("_", "\n", names(genes))
compGO <- compareCluster(geneCluster = genes,
                        fun          = "enrichGO",
                        ont          = "MF",
                        organism     = "human",
                        pvalueCutoff = 0.05,
                        pAdjustMethod = "BH")
plot(compGO, showCategory = 20, title = "Molecular Function Enrichment")
```

5.4 Overlap of peaks and annotated genes

User may want to compare the overlap peaks of replicate experiments or from different experiments. [ChIPseeker](#) provides `peak2GRanges` that can read peak file and stored in `GRanges` object. Several files can be read simultaneously using `lapply`, and then passed to `vennplot` to calculate their overlap and draw venn plot.

vennplot accept a list of object, can be a list of GRanges or a list of vector. Here, I will demonstrate using vennplot to visualize the overlap of the nearest genes stored in peakAnnoList.

```
genes= lapply(peakAnnoList, function(i) as.data.frame(i)$geneId)
vennplot(genes)
```

6 Statistical testing of ChIP seq overlap

Overlap is very important, if two ChIP experiment by two different proteins overlap in a large fraction of their peaks, they may cooperative in regulation. Calculating the overlap is only touch the surface. *ChIPseeker* implemented statistical methods to measure the significance of the overlap.

6.1 Shuffle genome coordination

```
p <- GRanges(seqnames=c("chr1", "chr3"),
             ranges=IRanges(start=c(1, 100), end=c(50, 130)))
shuffle(p, TxDb=txdb)

## GRanges object with 2 ranges and 0 metadata columns:
##      seqnames          ranges strand
##      <Rle>             <IRanges> <Rle>
## [1]      chr1 [157570854, 157570904]      *
## [2]      chr3 [117076405, 117076436]      *
## -----
## seqinfo: 2 sequences from an unspecified genome; no seqlengths
```

We implement the shuffle function to randomly permute the genomic locations of ChIP peaks defined in a genome which stored in TxDb object.

6.2 Peak overlap enrichment analysis

With the ease of this shuffle method, we can generate thousands of random ChIP data and calculate the background null distribution of the overlap among ChIP data sets.

```
enrichPeakOverlap(queryPeak = files[[5]],
                  targetPeak = unlist(files[1:4]),
                  TxDb       = txdb,
                  pAdjustMethod = "BH",
                  nShuffle    = 50,
                  chainFile    = NULL,
                  verbose      = FALSE)
```

```
##                                     qSample
## ARmo_OM      GSM1295077_CBX7_BF_ChipSeq_mergedReps_peaks.bed.gz
## ARmo_1nM     GSM1295077_CBX7_BF_ChipSeq_mergedReps_peaks.bed.gz
## ARmo_100nM   GSM1295077_CBX7_BF_ChipSeq_mergedReps_peaks.bed.gz
## CBX6_BF      GSM1295077_CBX7_BF_ChipSeq_mergedReps_peaks.bed.gz
##                                     tSample qLen tLen N_OL
## ARmo_OM      GSM1174480_ARmo_OM_peaks.bed.gz 1663 812 0
## ARmo_1nM     GSM1174481_ARmo_1nM_peaks.bed.gz 1663 2296 8
## ARmo_100nM   GSM1174482_ARmo_100nM_peaks.bed.gz 1663 1359 3
## CBX6_BF      GSM1295076_CBX6_BF_ChipSeq_mergedReps_peaks.bed.gz 1663 1331 968
## pvalue p.adjust
## ARmo_OM      0.90 0.900
## ARmo_1nM     0.20 0.400
## ARmo_100nM   0.38 0.507
## CBX6_BF      0.00 0.000
```

Parameter `queryPeak` is the query ChIP data, while `targetPeak` is bed file name or a vector of bed file names from comparison; `nShuffle` is the number to shuffle the peaks in `targetPeak`. To speed up the compilation of this vignettes, we only set `nShuffle` to 50 as an example for only demonstration. User should set the number to 1000 or above for more robust result. Parameter `chainFile` are chain file name for mapping the `targetPeak` to the genome version consistent with `queryPeak` when their genome version are different. This creat the possibility of comparison among different genome version and cross species.

In the output, `qSample` is the name of `queryPeak` and `qLen` is the the number of peaks in `queryPeak`. `N_OL` is the number of overlap between `queryPeak` and `targetPeak`.

7 Data Mining with ChIP seq data deposited in GEO

There are many ChIP seq data sets that have been published and deposited in GEO database. We can compare our own dataset to those deposited in GEO to search for significant overlap data. Significant overlap of ChIP seq data by different binding proteins may be used to infer cooperative regulation and thus can be used to generate hypotheses.

We collect about 15,000 bed files deposited in GEO, user can use `getGEOspecies` to get a summary based on speices.

7.1 GEO data collection

```
getGEOspecies()

##                                     species Freq
## 1                                Aedes aegypti 11
```

```

## 2          Anabaena 6
## 3      Anolis carolinensis 2
## 4      Anopheles gambiae 2
## 5          Apis mellifera 5
## 6      Apis mellifera scutellata 1
## 7      Arabidopsis lyrata 4
## 8      Arabidopsis thaliana 158
## 9      Atelerix albiventris 2
## 10         Bos taurus 2
## 11         Brassica rapa 8
## 12      Caenorhabditis elegans 164
## 13         Candida albicans 25
## 14         Candida dubliniensis 20
## 15         Canis lupus familiaris 7
## 16         Chlorocebus aethiops 2
## 17         Cleome hassleriana 1
## 18         Columba livia 6
## 19         Crassostrea gigas 1
## 20         Cryptococcus neoformans 51
## 21         Danio rerio 143
## 22         Drosophila melanogaster 642
## 23         Drosophila pseudoobscura 7
## 24         Drosophila simulans 12
## 25         Drosophila virilis 26
## 26         Drosophila yakuba 8
## 27         Equus caballus 1
## 28         Escherichia coli 13
## 29         Escherichia coli BW25113 4
## 30         Escherichia coli K-12 2
## 31     Escherichia coli str. K-12 substr. MG1655 8
## 32         Gallus gallus 55
## 33         Geobacter sulfurreducens PCA 3
## 34         Gorilla gorilla 2
## 35         Histophilus somni 1
## 36         Homo sapiens 8775
## 37         Human herpesvirus 6B 2
## 38         Human herpesvirus 8 6
## 39         Legionella pneumophila 5
## 40         Leishmania amazonensis 4
## 41         Leishmania major 2
## 42         Leishmania tarentolae 15
## 43         Macaca mulatta 75
## 44         Monodelphis domestica 8
## 45         Moraxella catarrhalis 035E 6

```



```
## 46          Mus musculus 6431
## 47      Mus musculus x Mus spretus      1
## 48      Mycobacterium tuberculosis      2
## 49          Myotis brandtii      15
## 50      Naumovozya castellii      1
## 51      Nematostella vectensis      23
## 52      Ornithorhynchus anatinus      5
## 53          Oryza sativa      23
## 54      Oryzias latipes      2
## 55      Pan troglodytes      51
## 56          Papio anubis      1
## 57      Plasmodium falciparum      5
## 58      Plasmodium falciparum 3D7      29
## 59      Pseudomonas putida KT2440      2
## 60          Pyrococcus furiosus      4
## 61          Rattus norvegicus      52
## 62      Rhodopseudomonas palustris      6
## 63      Rhodopseudomonas palustris CGA009      3
## 64          Saccharomyces cerevisiae      473
## 65      Saccharomyces cerevisiae x Saccharomyces paradoxus      16
## 66      Saccharomyces cerevisiae;\tMus musculus      12
## 67          Saccharomyces kudriavzevii      1
## 68          Saccharomyces paradoxus      8
## 69          Saccharomyces uvarum      1
## 70      Schizosaccharomyces japonicus      2
## 71      Schizosaccharomyces pombe      89
## 72      Schmidtea mediterranea      7
## 73          Sorghum bicolor      2
## 74      Streptomyces coelicolor A3(2)      6
## 75          Sus scrofa      23
## 76          Tupaia chinensis      7
## 77      Xenopus (Silurana) tropicalis      62
## 78          Xenopus laevis      2
## 79          Zea mays      56
```

The summary can also be based on genome version as illustrated below:

```
getGEOgenomeVersion()

##          organism genomeVersion Freq
## 1      Anolis carolinensis      anoCar2      2
## 2          Bos taurus      bosTau7      2
## 3      Caenorhabditis elegans      ce10      4
## 4      Caenorhabditis elegans      ce6      64
## 5          Danio rerio      danRer6      6
```

```
## 6          Danio rerio          danRer7    61
## 7      Drosophila melanogaster      dm3    415
## 8          Gallus gallus          galGal3    32
## 9          Gallus gallus          galGal4    15
## 10         Homo sapiens          hg18    2421
## 11         Homo sapiens          hg19    5737
## 12         Homo sapiens          hg38      4
## 13         Mus musculus          mm10     59
## 14         Mus musculus          mm8     485
## 15         Mus musculus          mm9    5325
## 16      Monodelphis domestica      monDom5     8
## 17          Pan troglodytes      panTro3    48
## 18          Macaca mulatta      rheMac2    71
## 19          Rattus norvegicus        rn5      1
## 20      Saccharomyces cerevisiae      sacCer2    141
## 21      Saccharomyces cerevisiae      sacCer3    158
## 22          Sus scrofa          susScr2     17
## 23 Xenopus (Silurana) tropicalis      xenTro3     3
```

User can access the detail information by `getGEOInfo`, for each genome version.

```
hg19 <- getGEOInfo(genome="hg19", simplify=TRUE)
head(hg19)
```

```
##      series_id      gsm      organism
## 111 GSE16256 GSM521889 Homo sapiens
## 112 GSE16256 GSM521887 Homo sapiens
## 113 GSE16256 GSM521883 Homo sapiens
## 114 GSE16256 GSM1010966 Homo sapiens
## 115 GSE16256 GSM896166 Homo sapiens
## 116 GSE16256 GSM910577 Homo sapiens
```

```
##
## 111      Reference Epigenome: ChIP-Seq Analysis of H3K27me3 in IMR90 Cells; renlab.F
## 112      Reference Epigenome: ChIP-Seq Analysis of H3K27ac in IMR90 Cells; renlab
## 113      Reference Epigenome: ChIP-Seq Analysis of H3K14ac in IMR90 Cells; renlab
## 114      polyA RNA sequencing of STL003 Pancreas Cultured Cells; polyA-R
## 115      Reference Epigenome: ChIP-Seq Analysis of H4K8ac in hESC H1 Cells; renlab.F
## 116 Reference Epigenome: ChIP-Seq Analysis of H3K4me1 in Human Spleen Tissue; renlab.H3F
##
## 111      ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM521nnn/GSM521889/suppl/GSM521889_U
## 112      ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM521nnn/GSM521887/suppl/GSM521887_
## 113      ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM521nnn/GSM521883/suppl/GSM521883_
## 114 ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM1010nnn/GSM1010966/suppl/GSM1010966_UCSD.F
## 115      ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM896nnn/GSM896166/suppl/GSM896
## 116      ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM910nnn/GSM910577/suppl/GSM910577_UCS
```

```
##      genomeVersion pubmed_id
## 111          hg19  19829295
## 112          hg19  19829295
## 113          hg19  19829295
## 114          hg19  19829295
## 115          hg19  19829295
## 116          hg19  19829295
```

If `simplify` is set to `FALSE`, extra information including `source_name`, `extract_protocol`, `description`, `data_processing`, and `submission_date` will be incorporated.

7.2 Download GEO ChIP data sets

ChIPseeker provide function `downloadGEObedFiles` to download all the bed files of a particular genome.

```
downloadGEObedFiles(genome="hg19", destDir="hg19")
```

Or a vector of GSM accession number by `downloadGSMbedFiles`.

```
gsm <- hg19$gsm[sample(nrow(hg19), 10)]
downloadGSMbedFiles(gsm, destDir="hg19")
```

7.3 Overlap significant testing

After download the bed files from GEO, we can pass them to `enrichPeakOverlap` for testing the significant of overlap. Parameter `targetPeak` can be the folder, e.g. `hg19`, that containing bed files. `enrichPeakOverlap` will parse the folder and compare all the bed files. It is possible to test the overlap with bed files that are mapping to different genome or different genome versions, `enrichPeakOverlap` provide a parameter `chainFile` that can pass a chain file and `liftOver` the `targetPeak` to the genome version consistent with `queryPeak`. Significant overlap can be use to generate hypothesis of cooperative regulation. By mining the data deposited in GEO, we can identify some putative complex or interacted regulators in gene expression regulation or chromosome remodelling for further validation.

8 External documents

- [Bug of R package ChIPpeakAnno](#)
- [ChIPseeker for ChIP peak annotation](#)
- [visualization methods in ChIPseeker](#)
- [multiple annotation in ChIPseeker](#)

9 Session Information

Here is the output of `sessionInfo()` on the system on which this document was compiled:

- R version 3.2.2 (2015-08-14), x86_64-apple-darwin13.4.0
- Locale: C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, stats4, utils
- Other packages: AnnotationDbi 1.30.1, Biobase 2.28.0, BiocGenerics 0.14.0, ChIPseeker 1.4.7, DBI 0.3.1, GO.db 3.1.2, GenomInfoDb 1.4.2, GenomicFeatures 1.20.3, GenomicRanges 1.20.5, IRanges 2.2.7, RSQLite 1.0.0, S4Vectors 0.6.3, TxDb.Hsapiens.UCSC.hg19.knownGene 3.1.2, clusterProfiler 2.2.5, org.Hs.eg.db 3.1.2
- Loaded via a namespace (and not attached): BiocParallel 1.2.20, BiocStyle 1.6.0, Biostrings 2.36.4, DO.db 2.9, DOSE 2.6.6, GOSemSim 1.26.0, GenomicAlignments 1.4.1, KEGGREST 1.8.0, KernSmooth 2.23-15, MASS 7.3-43, R6 2.1.1, RColorBrewer 1.1-2, RCurl 1.95-4.7, Rcpp 0.12.0, Rsamtools 1.20.4, XML 3.98-1.3, XVector 0.8.0, assertthat 0.1, biomaRt 2.24.0, bitops 1.0-6, boot 1.3-17, caTools 1.17.1, colorspace 1.2-6, digest 0.6.8, dplyr 0.4.2, evaluate 0.7.2, formatR 1.2, futile.logger 1.4.1, futile.options 1.0.0, gdata 2.17.0, ggplot2 1.0.1, gplots 2.17.0, grid 3.2.2, gtable 0.1.2, gtools 3.5.0, highr 0.5, httr 1.0.0, igraph 1.0.1, knitr 1.11, labeling 0.3, lambda.r 1.1.7, lazyeval 0.1.10, magrittr 1.5, munsell 0.4.2, plotrix 3.5-12, plyr 1.8.3, png 0.1-7, proto 0.3-10, qvalue 2.0.0, reshape2 1.4.1, rtracklayer 1.28.9, scales 0.3.0, splines 3.2.2, stringi 0.5-5, stringr 1.0.0, tools 3.2.2, zlibbioc 1.14.0

References

- [1] Guangchuang Yu, Li-Gen Wang, Guang-Rong Yan, and Qing-Yu He. DOSE: an r/bioconductor package for disease ontology semantic and enrichment analysis. 31(4):608–609. URL: <http://bioinformatics.oxfordjournals.org.eproxy2.lib.hku.hk/content/31/4/608>, doi:10.1093/bioinformatics/btu684.
- [2] Guangchuang Yu, Li-Gen Wang, Yanyan Han, and Qing-Yu He. clusterProfiler: an r package for comparing biological themes among gene clusters. *OMICS: A Journal of Integrative Biology*, 16(5):284–287, May 2012. URL: <http://online.liebertpub.com/doi/abs/10.1089/omi.2011.0118>, doi:10.1089/omi.2011.0118.
- [3] Helen Pemberton, Emma Anderton, Harshil Patel, Sharon Brookes, Hollie Chandler, Richard Palermo, Julie Stock, Marc Rodriguez-Niedenhr, Tomas Racek, Lucas de Breed, Aengus Stewart, Nik Matthews, and Gordon Peters. Genome-wide co-localization of polycomb orthologs and their effects on gene expression in human fibroblasts. 15(2):R23. PMID: 24485159. doi:10.1186/gb-2014-15-2-r23.
- [4] A Urbanucci, B Sahu, J Seppl, A Larjo, L M Latonen, K K Waltering, T L J Tammela, R L Vessella, H Lhdesmki, O A Jinne, and T Visakorpi. Overexpression of androgen receptor enhances the

binding of the receptor to the chromatin in prostate cancer. 31(17):2153–2163. PMID: 21909140. doi:10.1038/onc.2011.401.

- [5] Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, Midori A. Harris, David P. Hill, Laurie Issel-Tarver, Andrew Kasarskis, Suzanna Lewis, John C. Matese, Joel E. Richardson, Martin Ringwald, Gerald M. Rubin, and Gavin Sherlock. Gene ontology: tool for the unification of biology. 25:25–29. URL: <http://dx.doi.org/10.1038/75556>, doi:10.1038/75556.
- [6] Minoru Kanehisa, Susumu Goto, Shuichi Kawashima, Yasushi Okuno, and Masahiro Hattori. The KEGG resource for deciphering the genome. 32:D277–D280. PMID: 14681412. URL: http://nar.oxfordjournals.org/content/32/suppl_1/D277, doi:10.1093/nar/gkh063.
- [7] L. M. Schriml, C. Arze, S. Nadendla, Y.-W. W. Chang, M. Mazaitis, V. Felix, G. Feng, and W. A. Kibbe. Disease ontology: a backbone for disease semantic integration. 40:D940–D946. URL: <http://nar.oxfordjournals.org/content/40/D1/D940.long>, doi:10.1093/nar/gkr972.
- [8] D. Croft, A. F. Mundo, R. Haw, M. Milacic, J. Weiser, G. Wu, M. Caudy, P. Garapati, M. Gillespie, M. R. Kamdar, B. Jassal, S. Jupe, L. Matthews, B. May, S. Palatnik, K. Rothfels, V. Shamovsky, H. Song, M. Williams, E. Birney, H. Hermjakob, L. Stein, and P. D'Eustachio. The reactome pathway knowledgebase. 42:D472–D477. URL: <http://nar.oxfordjournals.org/content/42/D1/D472.long>, doi:10.1093/nar/gkt1102.

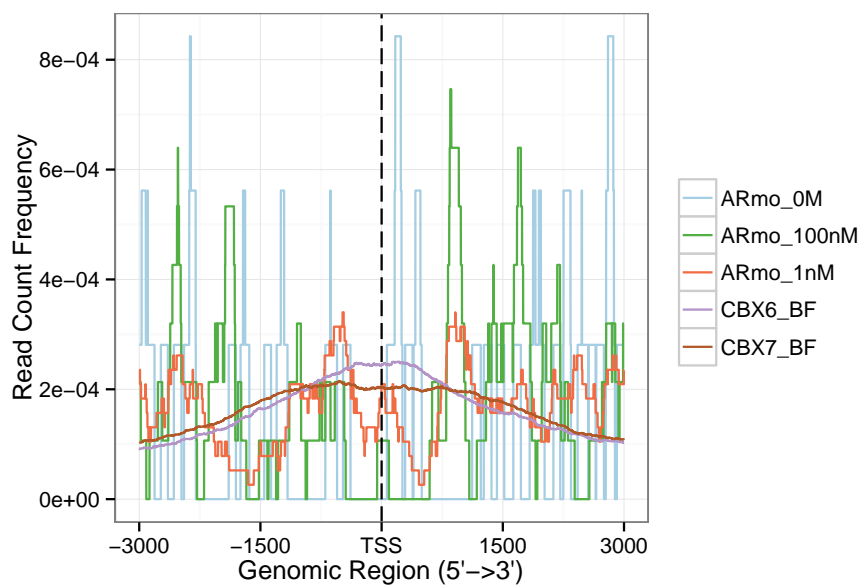


Figure 8: Average Profiles of ChIP peaks among different experiments

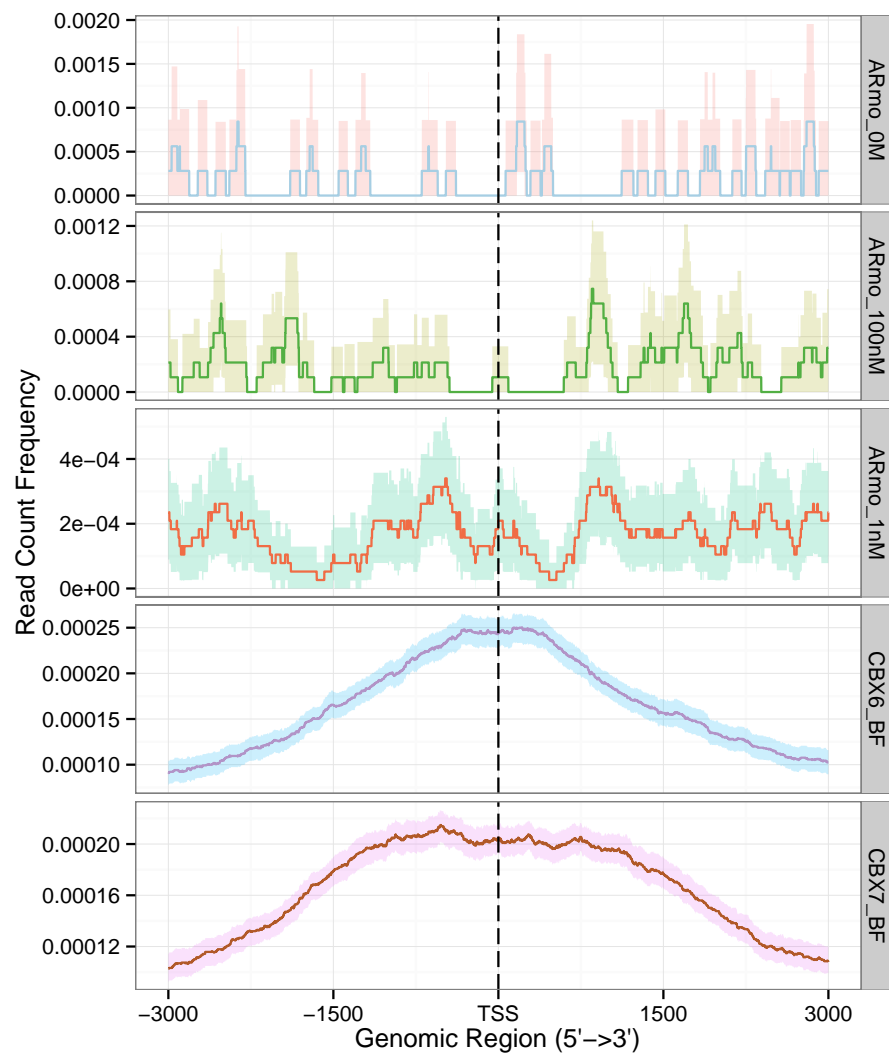


Figure 9: Average Profiles of ChIP peaks among different experiments

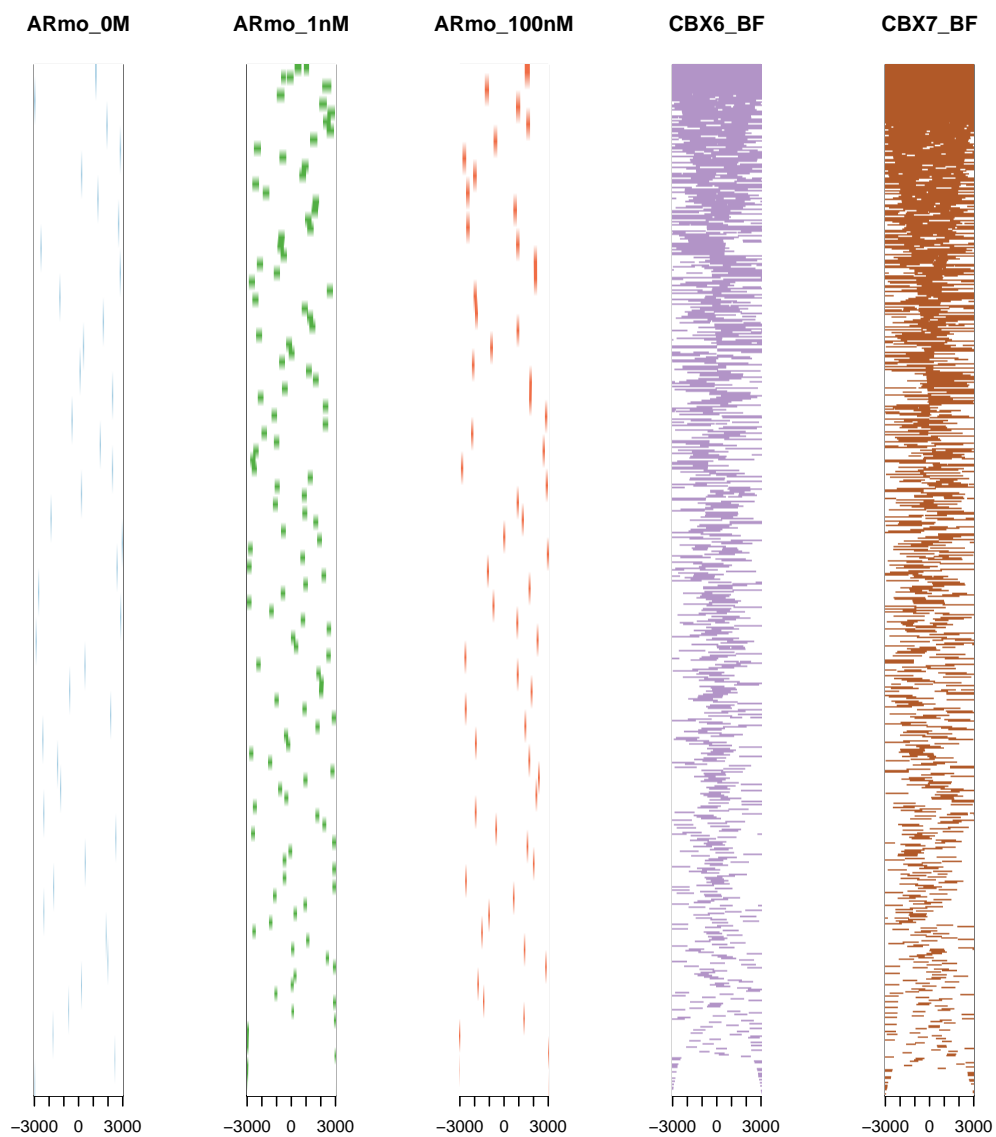


Figure 10: Heatmap of ChIP peaks among different experiments

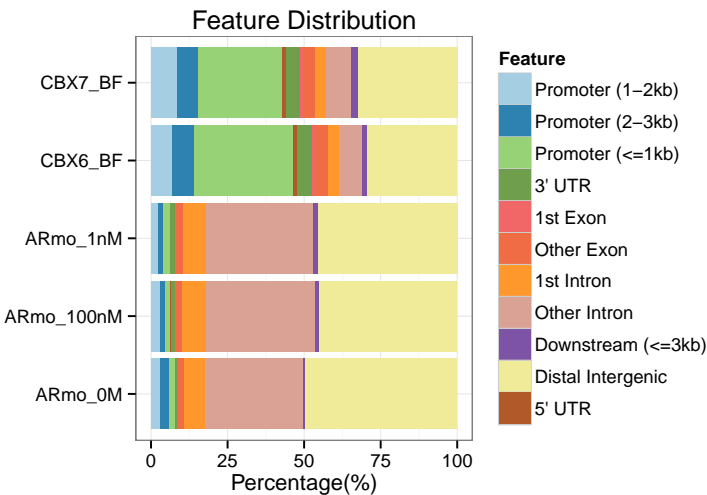


Figure 11: Genomic Annotation among different ChIPseq data

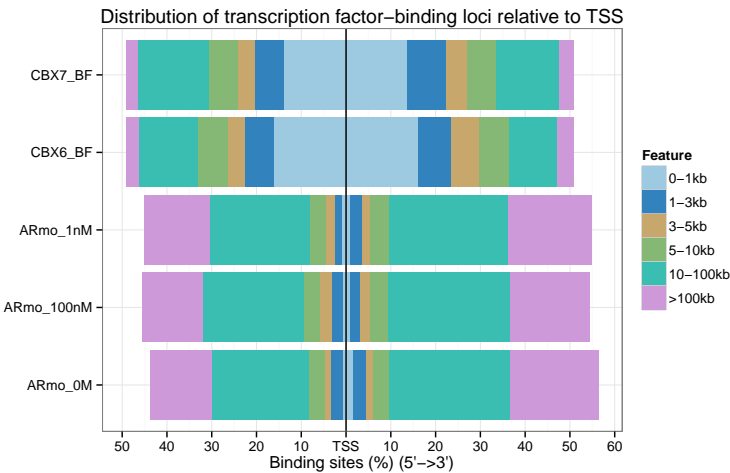


Figure 12: Distribution of Binding Sites among different ChIPseq data

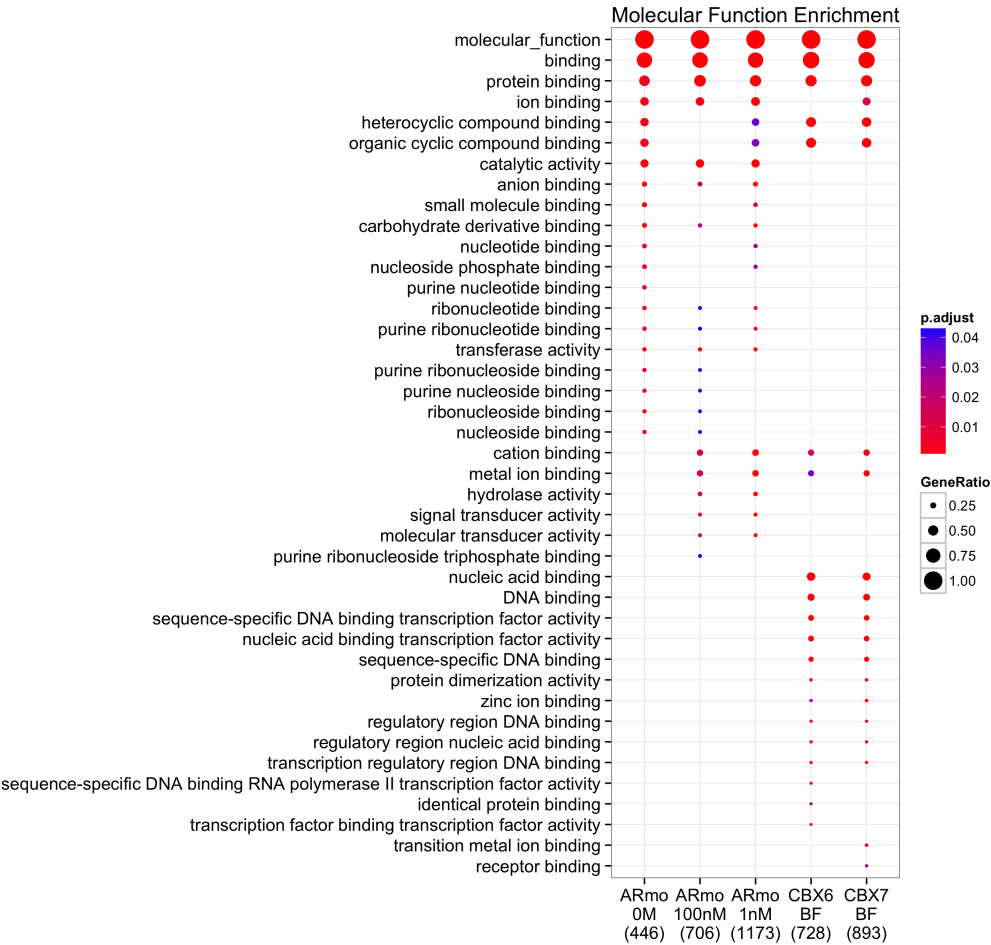


Figure 13: Compare Biological themes among different experiments

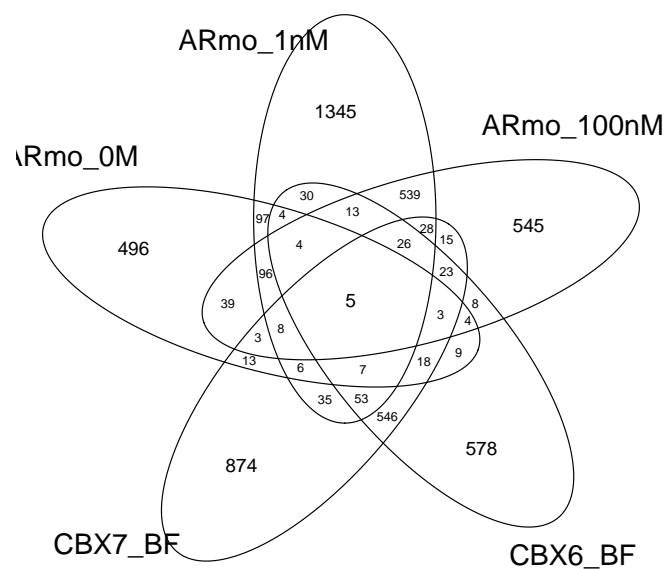


Figure 14: Overlap of annotated genes