The ChIPpeakAnno user's guide

Lihua Julie Zhu[*], Jianhong Ou[†]

May 18, 2015

Contents

[*]julie.zhu@umassmed.edu

[†]jianhong.ou@umassmed.edu

1 Introduction

Chromatin immunoprecipitation (ChIP) followed by high-throughput tag sequencing (ChIP-seq) and ChIP followed by genome tiling array analysis (ChIP-chip) become more and more prevalent high throughput technologies for identifying the binding sites of DNA-binding proteins in a genome-wide bases. A number of algorithms have been published to facilitate the identification of the binding sites of the DNA-binding proteins of interest. The identified binding sites in the list of peaks are usually converted to BED or WIG file format to be loaded to UCSC genome browser as custom tracks for investigators to view the proximity to various genomic features such as genes, exons and conserved elements. However, clicking through the genome browser could be a daunting task for the biologist if the number of peaks gets large or the peaks spread widely across the genome.

Here we have developed a Bioconducor package called ChIPpeakAnno to facilitate the batch annotation of the peaks identified from either ChIP-seq or ChIP-chip experiments. We have implemented functionality to find the nearest gene, exon, miRNA, gene end or custom features supplied by users such as most conserved elements and other transcription factor binding sites leveraging IRanges. Since the genome annotation gets updated from time to time, we have leveraged the *biomaRt* package from Bioconductor to retrieve the annotation data on the fly if the annotation of interest is available via the *biomaRt* package. The users also have the flexibility to pass their own annotation data as GRanges (or RangedData) or pass in annotation data from *GenomicFeatures*. We have also leveraged *BSgenome* and *biomaRt* package on implementing functions to retrieve the sequences around the peak identified for peak validation. To understand whether the identified peaks are enriched around genes with certain GO terms, we have implemented GO enrichment test in *ChIPpeakAnno* package leveraging the hypergeometric test phyper in *stats* package and integrated with Gene Ontology (GO) annotation from *GO.db* package and multiplicity adjustment functions from *multtest* package.

2 Quick start

```
> library(ChIPpeakAnno)
> ## import the MACS output
> macs <- system.file("extdata", "MACS_peaks.xls", package="ChIPpeakAnno")
> macsOutput <- toGRanges(macs, format="MACS")
> ## annotate the peaks with ensembl annotation
> data(TSS.human.GRCh38)
> macs.anno <- annotatePeakInBatch(macsOutput, AnnotationData=TSS.human.GRCh38,
+                                  output="overlapping", maxgap=5000L)
> ## add gene symbols
> library(org.Hs.eg.db)
> macs.anno <- addGeneIDs(annotatedPeak=macs.anno,
+                         orgAnn="org.Hs.eg.db",
+                         IDs2Add="symbol")
> head(macs.anno)

GRanges object with 6 ranges and 16 metadata columns:
```

```
                      seqnames                ranges strand |   length   summit     tags
                         <Rle>             <IRanges>  <Rle> | <factor> <factor> <factor>
  X01.ENSG00000117616     chr1 [ 25323511,  25324015]      * |      505      252       45
  X01.ENSG00000187010     chr1 [ 25323511,  25324015]      * |      505      252       45
  X02.ENSG00000183726     chr1 [ 25362685,  25362997]      * |      313      211       33
  X02.ENSG00000188672     chr1 [ 25362685,  25362997]      * |      313      211       33
               X03.NA     chr1 [145558152, 145558537]      * |      386       59       39
               X04.NA    chr10 [ 47088702,  47089329]      * |      628      484       68
                        qvalue fold_enrichment      FDR        peak          feature
                      <factor>        <factor> <factor> <character>      <character>
  X01.ENSG00000117616    59.17           17.01      5.8         X01 ENSG00000117616
  X01.ENSG00000187010    59.17           17.01      5.8         X01 ENSG00000187010
  X02.ENSG00000183726    60.63           22.41      4.2         X02 ENSG00000183726
  X02.ENSG00000188672    60.63           22.41      4.2         X02 ENSG00000188672
               X03.NA    53.10           20.68      2.3         X03             <NA>
               X04.NA    56.09           16.37     0.75         X04             <NA>
                      start_position end_position feature_strand insideFeature
                           <integer>    <integer>    <character>      <factor>
  X01.ENSG00000117616       25242237     25338213              -        inside
  X01.ENSG00000187010       25272393     25330445              +        inside
  X02.ENSG00000183726       25337917     25362361              +    downstream
  X02.ENSG00000188672       25362249     25430192              -        inside
               X03.NA           <NA>         <NA>           <NA>          <NA>
               X04.NA           <NA>         <NA>           <NA>          <NA>
                      distancetoFeature shortestDistance fromOverlappingOrNearest
                              <numeric>        <integer>              <character>
  X01.ENSG00000117616             14702            14198              Overlapping
  X01.ENSG00000187010             51118             6430              Overlapping
  X02.ENSG00000183726             24768              324              Overlapping
  X02.ENSG00000188672             67507              436              Overlapping
               X03.NA              <NA>             <NA>                     <NA>
               X04.NA              <NA>             <NA>                     <NA>
                                  symbol
                                <factor>
  X01.ENSG00000117616 LOC101928189;RSRP1
  X01.ENSG00000187010           RHCE;RHD
  X02.ENSG00000183726            TMEM50A
  X02.ENSG00000188672               RHCE
               X03.NA               <NA>
               X04.NA               <NA>
  -------
  seqinfo: 12 sequences from an unspecified genome; no seqlengths
```

```
> if(interactive()){## annotate the peaks with UCSC annotation
+     library(GenomicFeatures)
+     library(TxDb.Hsapiens.UCSC.hg38.knownGene)
+     ucsc.hg38.knownGene <- genes(TxDb.Hsapiens.UCSC.hg38.knownGene)
+     macs.anno <- annotatePeakInBatch(macsOutput,
+                                      AnnotationData=ucsc.hg38.knownGene,
+                                      output="overlapping", maxgap=5000L)
+     macs.anno <- addGeneIDs(annotatedPeak=macs.anno,
+                             orgAnn="org.Hs.eg.db",
+                             feature_id_type="entrez_id",
+                             IDs2Add="symbol")
+     head(macs.anno)
+ }
```

3 Examples of using ChIPpeakAnno

3.1 Task 1: Find the nearest feature such as gene and the distance to the feature such as the transcription start site (TSS) of the nearest gene

We have a list of peaks identified from ChIP-seq or ChIP-chip experiments and we would like to retrieve the nearest gene and distance to the corresponding gene transcription start site. We have retrieved all the genomic locations of the genes for human genome as TSS.human.NCBI36 data package for repeated use with function `getAnnotation`, now we just pass the annotation to the `annotatePeakInBatch` function.

```
> library(ChIPpeakAnno)
> data(myPeakList)
> data(TSS.human.NCBI36)
> annotatedPeak <- annotatePeakInBatch(myPeakList[1:6,],
+                                     AnnotationData=TSS.human.NCBI36)
> annotatedPeak
```

```
GRanges object with 6 ranges and 9 metadata columns:
                               seqnames             ranges strand |         peak
                                  <Rle>          <IRanges>  <Rle> |  <character>
   X1_93_556427.ENSG00000212875     chr1 [ 556660,  556760]      * |  X1_93_556427
   X1_41_559455.ENSG00000212678     chr1 [ 559774,  559874]      * |  X1_41_559455
   X1_12_703729.ENSG00000197049     chr1 [ 703885,  703985]      * |  X1_12_703729
   X1_20_925025.ENSG00000188290     chr1 [ 926058,  926158]      * |  X1_20_925025
  X1_11_1041174.ENSG00000131591     chr1 [1041646, 1041746]      * | X1_11_1041174
  X1_14_1269014.ENSG00000107404     chr1 [1270239, 1270339]      * | X1_14_1269014
                                       feature start_position end_position
                                   <character>      <integer>    <integer>
   X1_93_556427.ENSG00000212875 ENSG00000212875         556318       557859
   X1_41_559455.ENSG00000212678 ENSG00000212678         559620       560165
   X1_12_703729.ENSG00000197049 ENSG00000197049         711184       712376
   X1_20_925025.ENSG00000188290 ENSG00000188290         924209       925333
  X1_11_1041174.ENSG00000131591 ENSG00000131591        1007062      1041341
  X1_14_1269014.ENSG00000107404 ENSG00000107404        1260523      1274623
                               feature_strand insideFeature distancetoFeature
                                  <character>      <factor>         <numeric>
   X1_93_556427.ENSG00000212875              +        inside               342
   X1_41_559455.ENSG00000212678              +        inside               154
   X1_12_703729.ENSG00000197049              +      upstream             -7299
   X1_20_925025.ENSG00000188290              -      upstream              -725
  X1_11_1041174.ENSG00000131591              -      upstream              -305
  X1_14_1269014.ENSG00000107404              -        inside              4384
                               shortestDistance fromOverlappingOrNearest
                                      <integer>              <character>
   X1_93_556427.ENSG00000212875              342          NearestLocation
   X1_41_559455.ENSG00000212678              154          NearestLocation
   X1_12_703729.ENSG00000197049             7199          NearestLocation
   X1_20_925025.ENSG00000188290              725          NearestLocation
  X1_11_1041174.ENSG00000131591              305          NearestLocation
  X1_14_1269014.ENSG00000107404             4284          NearestLocation
  -------
  seqinfo: 24 sequences from an unspecified genome; no seqlengths
```

To annotate the peaks with other genomic feature, you will need to call function `getAnnotation` with featureType, e.g., "Exon" for finding the nearest exon, and "miRNA" for finding the nearest miRNA, "5utr" or "3utr"for finding the overlapping 5 prime UTR or 3 prime UTR. Please refer to `getAnnotation` function for more details.

We have presented the examples using human genome as annotation source. To annotate your data with other species, you will need to pass to the function `getAnnotation` the appropriate dataset for example, drerio_gene_ensembl for zebrafish genome, mmusculus_gene_ensembl for mouse genome and rnorvegicus_gene_ensembl for rat genome.

For a list of available biomart and dataset, please refer to the *biomaRt* package documentation (Durinck S. et al., 2005). For fast access, in addition to TSS.human.NCBI36, TSS.human.GRCh37, TSS.human.GRCh38, TSS.mouse.NCBIM37, TSS.mouse.GRCm38, TSS.rat.RGSC3.4, TSS.rat.Rnor_5.0, TSS.zebrafish.Zv8, and TSS.zebrafish.Zv9 are included as annotation data packages.

You could also pass your own annotation data into the function `annotatePeakInBatch`. For example, if you have a list of transcription factor biding sites from literature and are interested in obtaining the nearest binding site of the transcription factor and distance to it for the list of peaks.

```
> myPeak1 <- GRanges(seqnames=c("1", "2", "3", "4", "5", "6",
+                             "2", "6", "6", "6", "6", "5"),
+                  ranges=IRanges(start=c(967654, 2010897, 2496704, 3075869,
+                                         3123260, 3857501, 201089, 1543200,
+                                         1557200, 1563000, 1569800, 167889600),
+                                 end= c(967754, 2010997, 2496804, 3075969,
+                                        3123360, 3857601, 201089, 1555199,
+                                        1560599, 1565199, 1573799, 167893599),
+                                 names=paste("Site", 1:12, sep="")))
> TFbindingSites <- GRanges(seqnames=c("1", "2", "3", "4", "5", "6", "1", "2", "3",
+                                   "4", "5", "6", "6", "6", "6", "6", "5"),
+                         ranges=IRanges(start=c(967659, 2010898, 2496700,
+                                                3075866, 3123260, 3857500,
+                                                96765, 201089, 249670, 307586,
+                                                312326, 385750, 1549800,
+                                                1554400, 1565000, 1569400,
+                                                167888600),
+                                        end=c(967869, 2011108, 2496920,
+                                              3076166,3123470, 3857780,
+                                              96985, 201299, 249890, 307796,
+                                              312586, 385960, 1550599, 1560799,
+                                              1565399, 1571199, 167888999),
+                                        names=paste("t", 1:17, sep="")),
+                         strand=c("+", "+", "+", "+", "+", "+", "-", "-", "-",
+                                  "-", "-", "-", "+", "+", "+", "+", "+"))
> annotatedPeak2 <- annotatePeakInBatch(myPeak1, AnnotationData=TFbindingSites)
> annotatedPeak2

GRanges object with 12 ranges and 9 metadata columns:
             seqnames                 ranges strand   |        peak     feature
                <Rle>              <IRanges>  <Rle>   | <character> <character>
    Site1.t1     chr1     [ 967654,  967754]      *   |       Site1          t1
    Site2.t2     chr2     [2010897, 2010997]      *   |       Site2          t2
    Site3.t3     chr3     [2496704, 2496804]      *   |       Site3          t3
    Site4.t4     chr4     [3075869, 3075969]      *   |       Site4          t4
    Site5.t5     chr5     [3123260, 3123360]      *   |       Site5          t5
         ...      ...                    ...    ... ...         ...         ...
   Site8.t14     chr6 [  1543200,   1555199]      *   |       Site8         t14
   Site9.t14     chr6 [  1557200,   1560599]      *   |       Site9         t14
  Site10.t15     chr6 [  1563000,   1565199]      *   |      Site10         t15
  Site11.t16     chr6 [  1569800,   1573799]      *   |      Site11         t16
  Site12.t17     chr5 [167889600, 167893599]      *   |      Site12         t17
             start_position end_position feature_strand insideFeature distancetoFeature
                  <integer>    <integer>    <character>      <factor>         <numeric>
    Site1.t1         967659       967869              +   overlapStart                -5
```

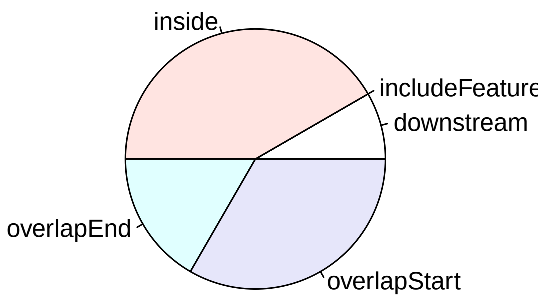

Figure 1: Pie chart of peak distribution among features.

```
   Site2.t2       2010898      2011108             +  overlapStart              -1
   Site3.t3       2496700      2496920             +        inside               4
   Site4.t4       3075866      3076166             +        inside               3
   Site5.t5       3123260      3123470             +        inside               0
        ...           ...          ...           ...           ...             ...
  Site8.t14       1554400      1560799             +  overlapStart          -11200
  Site9.t14       1554400      1560799             +        inside            2800
 Site10.t15       1565000      1565399             +  overlapStart           -2000
 Site11.t16       1569400      1571199             +    overlapEnd             400
 Site12.t17     167888600    167888999             +    downstream            1000
            shortestDistance fromOverlappingOrNearest
                   <integer>              <character>
   Site1.t1                5          NearestLocation
   Site2.t2                1          NearestLocation
   Site3.t3                4          NearestLocation
   Site4.t4                3          NearestLocation
   Site5.t5                0          NearestLocation
        ...              ...                      ...
  Site8.t14              799          NearestLocation
  Site9.t14              200          NearestLocation
 Site10.t15              199          NearestLocation
 Site11.t16              400          NearestLocation
 Site12.t17              601          NearestLocation
 -------
 seqinfo: 6 sequences from an unspecified genome; no seqlengths
```

```
> pie(table(as.data.frame(annotatedPeak2)$insideFeature))
```

Both BED format and GFF format are common file format that provides a flexible way to define the peaks and annotations as the data lines. Therefore, conversion functions `toGRanges` were implemented for converting these data format to GRanges before calling `annotatePeakInBatch`

Once you annotated the peak list, you can plot the distance to nearest feature such as TSS.

3.2 Task 2: Obtain overlapping peaks for potential transcription factor complex and determine the significance of the overlapping and generate Venn Diagram

Here is an example of obtaining overlapping peaks with maximum gap 1kb for two peak ranges.

```
> peaks1 <- GRanges(seqnames=c("1", "2", "3", "4", "5", "6",
+                              "2", "6", "6", "6", "6", "5"),
+                   ranges=IRanges(start=c(967654, 2010897, 2496704, 3075869,
+                                          3123260, 3857501, 201089, 1543200,
+                                          1557200, 1563000, 1569800, 167889600),
+                                  end= c(967754, 2010997, 2496804, 3075969,
+                                         3123360, 3857601, 201089, 1555199,
+                                         1560599, 1565199, 1573799, 167893599),
+                                  names=paste("Site", 1:12, sep="")),
+                  strand="+")
> peaks2 <- GRanges(seqnames=c("1", "2", "3", "4", "5", "6", "1", "2", "3",
+                                     "4", "5", "6", "6", "6", "6", "6", "5"),
+                          ranges=IRanges(start=c(967659, 2010898, 2496700,
+                                                 3075866, 3123260, 3857500,
+                                                 96765, 201089, 249670, 307586,
+                                                 312326, 385750, 1549800,
+                                                 1554400, 1565000, 1569400,
+                                                 167888600),
+                                         end=c(967869, 2011108, 2496920,
+                                               3076166,3123470, 3857780,
+                                               96985, 201299, 249890, 307796,
+                                               312586, 385960, 1550599, 1560799,
+                                               1565399, 1571199, 167888999),
+                                         names=paste("t", 1:17, sep="")),
+                          strand=c("+", "+", "+", "+", "+", "+", "-", "-", "-",
+                                   "-", "-", "-", "+", "+", "+", "+", "+"))
> ol <- findOverlapsOfPeaks(peaks1, peaks2, maxgap=1000)
> peaklist <- ol$peaklist
```

Here is a list of overlapping peaks with maximum gap 1kb and a pie graph describing the distribution of relative position of peaks1 to peaks2 for overlapping peaks.

```
> overlappingPeaks <- ol$overlappingPeaks
> overlappingPeaks

$`peaks1///peaks2`
                                   peaks1 seqnames     start       end width strand
peaks1__Site1_peaks2__t1    peaks1__Site1        1    967654    967754   101      +
peaks1__Site7_peaks2__t8    peaks1__Site7        2    201089    201089     1      +
peaks1__Site2_peaks2__t2    peaks1__Site2        2   2010897   2010997   101      +
peaks1__Site3_peaks2__t3    peaks1__Site3        3   2496704   2496804   101      +
peaks1__Site4_peaks2__t4    peaks1__Site4        4   3075869   3075969   101      +
peaks1__Site5_peaks2__t5    peaks1__Site5        5   3123260   3123360   101      +
peaks1__Site12_peaks2__t17 peaks1__Site12        5 167889600 167893599  4000      +
peaks1__Site8_peaks2__t13   peaks1__Site8        6   1543200   1555199 12000      +
peaks1__Site8_peaks2__t14   peaks1__Site8        6   1543200   1555199 12000      +
peaks1__Site9_peaks2__t14   peaks1__Site9        6   1557200   1560599  3400      +
peaks1__Site10_peaks2__t15 peaks1__Site10        6   1563000   1565199  2200      +
peaks1__Site11_peaks2__t16 peaks1__Site11        6   1569800   1573799  4000      +
peaks1__Site6_peaks2__t6    peaks1__Site6        6   3857501   3857601   101      +
                                peaks2 seqnames     start       end width strand
peaks1__Site1_peaks2__t1    peaks2__t1        1    967659    967869   211      +
peaks1__Site7_peaks2__t8    peaks2__t8        2    201089    201299   211      -
peaks1__Site2_peaks2__t2    peaks2__t2        2   2010898   2011108   211      +
peaks1__Site3_peaks2__t3    peaks2__t3        3   2496700   2496920   221      +
peaks1__Site4_peaks2__t4    peaks2__t4        4   3075866   3076166   301      +
```

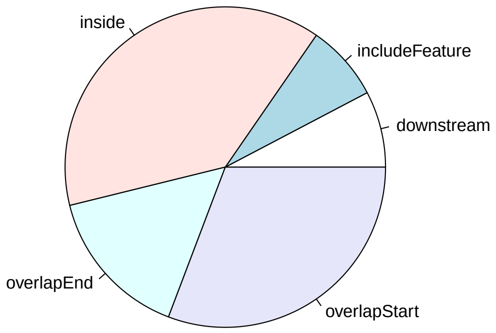

Figure 2: Pie chart of common peaks among features.

```
peaks1__Site5_peaks2__t5    peaks2__t5        5   3123260   3123470   211      +
peaks1__Site12_peaks2__t17 peaks2__t17        5 167888600 167888999   400      +
peaks1__Site8_peaks2__t13  peaks2__t13        6   1549800   1550599   800      +
peaks1__Site8_peaks2__t14  peaks2__t14        6   1554400   1560799  6400      +
peaks1__Site9_peaks2__t14  peaks2__t14        6   1554400   1560799  6400      +
peaks1__Site10_peaks2__t15 peaks2__t15        6   1565000   1565399   400      +
peaks1__Site11_peaks2__t16 peaks2__t16        6   1569400   1571199  1800      +
peaks1__Site6_peaks2__t6    peaks2__t6        6   3857500   3857780   281      +
                           overlapFeature shortestDistance
peaks1__Site1_peaks2__t1     overlapStart                5
peaks1__Site7_peaks2__t8       overlapEnd                0
peaks1__Site2_peaks2__t2     overlapStart                1
peaks1__Site3_peaks2__t3           inside                4
peaks1__Site4_peaks2__t4           inside                3
peaks1__Site5_peaks2__t5           inside                0
peaks1__Site12_peaks2__t17     downstream              601
peaks1__Site8_peaks2__t13  includeFeature             4600
peaks1__Site8_peaks2__t14    overlapStart              799
peaks1__Site9_peaks2__t14          inside              200
peaks1__Site10_peaks2__t15   overlapStart              199
peaks1__Site11_peaks2__t16     overlapEnd              400
peaks1__Site6_peaks2__t6           inside                1
```

```
> pie(table(overlappingPeaks[["peaks1///peaks2"]]$overlapFeature))
```

Here is the merged overlapping peaks, which can be used to obtain overlapping peaks with another TF binding sites from a protein complex.

```
> peaklist[["peaks1///peaks2"]]
```

```
GRanges object with 11 ranges and 1 metadata column:
       seqnames               ranges strand   |
          <Rle>            <IRanges>  <Rle>   |
   [1]        1     [ 967654,  967869]      +   |
   [2]        2     [ 201089,  201299]      *   |
   [3]        2     [2010897, 2011108]      +   |
   [4]        3     [2496700, 2496920]      +   |
   [5]        4     [3075866, 3076166]      +   |
   ...      ...                    ...    ... ...
   [7]        5 [167888600, 167893599]      +   |
   [8]        6 [  1543200,   1560799]      +   |
   [9]        6 [  1563000,   1565399]      +   |
  [10]        6 [  1569400,   1573799]      +   |
  [11]        6 [  3857500,   3857780]      +   |
                                       peakNames
                                 <CharacterList>
   [1]                 peaks1__Site1,peaks2__t1
   [2]                 peaks1__Site7,peaks2__t8
   [3]                 peaks1__Site2,peaks2__t2
   [4]                 peaks2__t3,peaks1__Site3
   [5]                 peaks2__t4,peaks1__Site4
   ...                                      ...
   [7]               peaks2__t17,peaks1__Site12
   [8] peaks1__Site8,peaks2__t13,peaks2__t14,...
   [9]               peaks1__Site10,peaks2__t15
  [10]               peaks2__t16,peaks1__Site11
  [11]                 peaks2__t6,peaks1__Site6
  -------
  seqinfo: 6 sequences from an unspecified genome; no seqlengths
```

Here is the peaks in peaks1 that not overlaps with peaks in peaks2

```
> peaklist[["peaks1"]]

NULL
```

Here is the peaks in peaks2 that not overlap with peaks in peaks1

```
> peaklist[["peaks2"]]

GRanges object with 5 ranges and 1 metadata column:
      seqnames           ranges strand |       peakNames
         <Rle>        <IRanges>  <Rle> | <CharacterList>
  [1]        1 [ 96765,  96985]      - |      peaks2__t7
  [2]        3 [249670, 249890]      - |      peaks2__t9
  [3]        4 [307586, 307796]      - |     peaks2__t10
  [4]        5 [312326, 312586]      - |     peaks2__t11
  [5]        6 [385750, 385960]      - |     peaks2__t12
  -------
  seqinfo: 6 sequences from an unspecified genome; no seqlengths
```

Venn Diagram can be generated by the following function call using the results of `findOverlapsOfPeaks` as an input (Figure 3). P-values indicate whether the extent of overlapping is significant.

```
> makeVennDiagram(ol, totalTest=1e+2)

$p.value
     peaks1 peaks2         pval
[1,]      1      1 5.890971e-12

$vennCounts
     peaks1 peaks2 Counts
[1,]      0      0     83
[2,]      0      1      5
[3,]      1      0      0
[4,]      1      1     12
attr(,"class")
[1] "VennCounts"
```

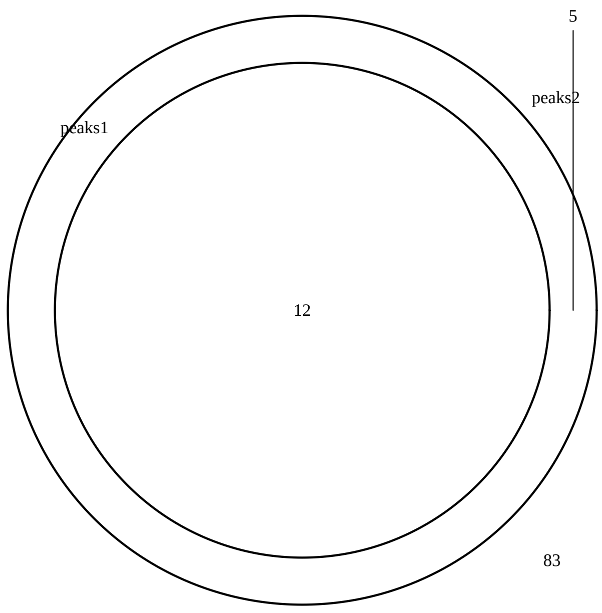

Figure 3: venn diagram of overlaps

Users can also try other tools to draw vennDiagrams such as *Vennerable*.

```
> #     install.packages("Vennerable", repos="http://R-Forge.R-project.org", type="source")
> #     library(Vennerable)
> #     venn_cnt2venn <- function(venn_cnt){
> #         n <- which(colnames(venn_cnt)=="Counts") - 1
> #         SetNames=colnames(venn_cnt)[1:n]
> #         Weight=venn_cnt[,"Counts"]
> #         names(Weight) <- apply(venn_cnt[,1:n], 1, paste, collapse="")
> #         Venn(SetNames=SetNames, Weight=Weight)
> #     }
> #
> #     v <- venn_cnt2venn(ol$venn_cnt)
> #     plot(v)
```

The `findOverlapsOfPeaks` function can be called to obtain overlaps upto 5 peak lists for example, the overlap peaks in peaks1, peaks2 and peaks3 (Figure 4).

```
> peaks3 <- GRanges(seqnames=c("1", "2", "3", "4", "5",
+                              "6", "1", "2", "3", "4"),
+                   ranges=IRanges(start=c(967859, 2010868, 2496500, 3075966,
+                                          3123460, 3851500, 96865, 201189,
+                                          249600, 307386),
+                                  end= c(967969, 2011908, 2496720, 3076166,
+                                         3123470, 3857680, 96985, 201299,
+                                         249890, 307796),
+                                  names=paste("p", 1:10, sep="")),
+                   strand=c("+", "+", "+", "+", "+",
+                            "+", "-", "-", "-", "-"))
> ol <- findOverlapsOfPeaks(peaks1, peaks2, peaks3, maxgap=1000, connectedPeaks="min")
> makeVennDiagram(ol, totalTest=1e+2)

$p.value
     peaks1 peaks2 peaks3         pval
[1,]      0      1      1 1.123492e-09
```

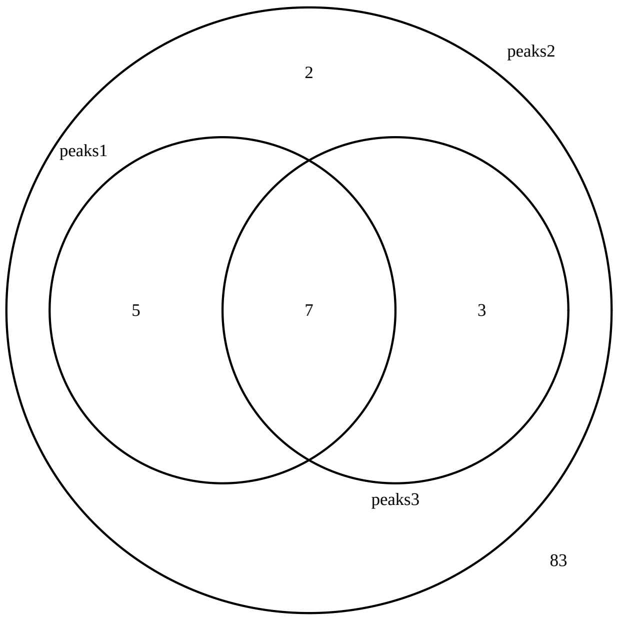

Figure 4: venn diagram of overlaps for three input peak lists

```
[2,]      1      0      1 5.131347e-06
[3,]      1      1      0 5.890971e-12

$vennCounts
     peaks1 peaks2 peaks3 Counts
[1,]      0      0      0     83
[2,]      0      0      1      0
[3,]      0      1      0      2
[4,]      0      1      1      3
[5,]      1      0      0      0
[6,]      1      0      1      0
[7,]      1      1      0      5
[8,]      1      1      1      7
attr(,"class")
[1] "VennCounts"
```

Venn Diagram can also be generated by the following function call with p-value that indicates whether the extent of overlapping is significant (Figure 5,6). Note, the maxgap is changed to 0.

```
> makeVennDiagram(list(peaks1, peaks2), NameOfPeaks=c("TF1", "TF2"),
+                 maxgap=0, minoverlap =1, totalTest=100)

$p.value
     TF1 TF2         pval
[1,]   1   1 9.837922e-10

$vennCounts
     TF1 TF2 Counts
[1,]   0   0     82
[2,]   0   1      6
[3,]   1   0      1
[4,]   1   1     11
attr(,"class")
[1] "VennCounts"
```

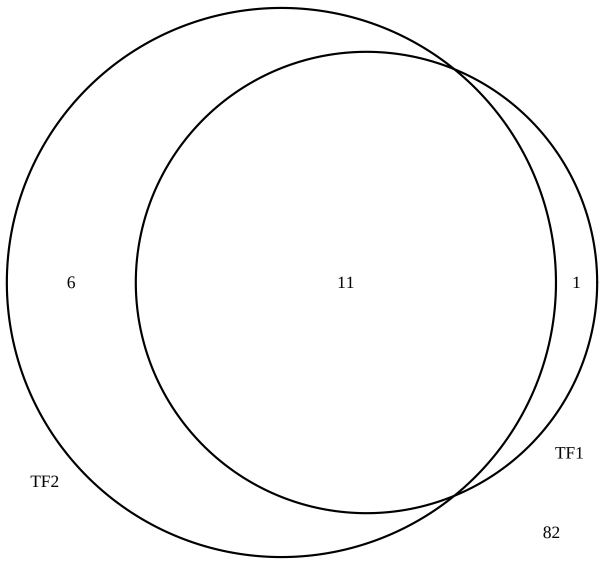

Figure 5: Venn diagram to depict the overlaps between two peak lists

```
> makeVennDiagram(list(peaks1, peaks2, peaks3),
+                 NameOfPeaks=c("TF1", "TF2", "TF3"),
+                 maxgap=0, minoverlap =1, totalTest=100)

$p.value
     TF1 TF2 TF3         pval
[1,]   0   1   1 1.123492e-09
[2,]   1   0   1 5.131347e-06
[3,]   1   1   0 9.837922e-10

$vennCounts
     TF1 TF2 TF3 Counts
[1,]   0   0   0     82
[2,]   0   0   1      0
[3,]   0   1   0      3
[4,]   0   1   1      3
[5,]   1   0   0      1
[6,]   1   0   1      0
[7,]   1   1   0      4
[8,]   1   1   1      7
attr(,"class")
[1] "VennCounts"
```

3.3 Task 3: Obtain sequences surrounding the peaks for PCR validation or motif discovery

Here is an example of obtaining sequences surrounding the peak intervals including 20 bp upstream and downstream sequence.

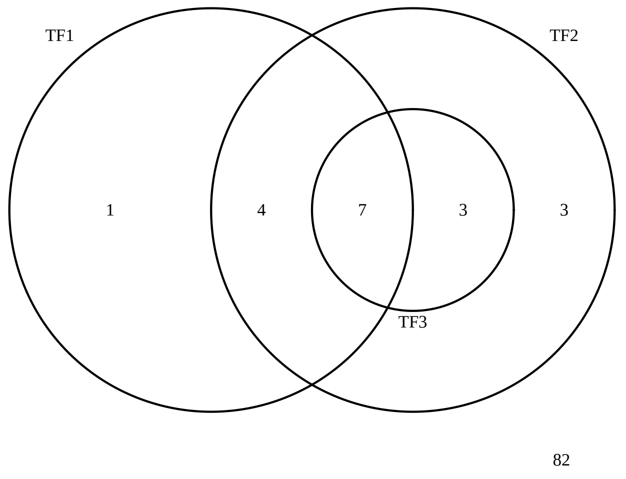

Figure 6: venn diagram of overlaps for three input peaklists directly

```
> peaks <- GRanges(seqnames=c("NC_008253", "NC_010468"),
+                  ranges=IRanges(start=c(100, 500),
+                                 end=c(300, 600),
+                                 names=c("peak1", "peak2")))
> library(BSgenome.Ecoli.NCBI.20080805)
> peaksWithSequences <- getAllPeakSequence(peaks, upstream=20,
+                                          downstream=20, genome=Ecoli)
```

You can easily convert the obtained sequences into fasta format for motif discovery by calling the function `write2FASTA`.

```
> write2FASTA(peaksWithSequences,"test.fa")
```

3.4 Task 4: Obtain enriched gene ontology (GO) terms or KEGG terms near the peaks

Once you have obtained the annotated peak data from the example above, you can also use the function `getEnriched` to obtain a list of enriched gene ontology (GO) terms via *GOstats*. The ontology could also be set as KEGG or reactome.

Once you have obtained the annotated peak data from the example above, you can also use the function `getEnrichedGO` to obtain a list of enriched gene ontology (GO) terms using hypergeometric test.

library(org.Hs.eg.db)

$enrichedGO = getEnrichedGO$ (annotatedPeak, $orgAnn = $ "$org.Hs.eg.db$", $maxP = 0.01$, $multiAdj = TRUE$, $minGOterm = 10$, $multiAdjMethod = $ "BH")

```
> library(org.Hs.eg.db)
> over <- getEnrichedGO(annotatedPeak, orgAnn="org.Hs.eg.db",
+                       maxP=0.01, multiAdj=FALSE, minGOterm=10, multiAdjMethod="")
> head(over[["bp"]])
```

```
       go.id                            go.term
1 GO:0001736     establishment of planar polarity
2 GO:0001840             neural plate development
3 GO:0001941   postsynaptic membrane organization
4 GO:0001964                     startle response
5 GO:0007164     establishment of tissue polarity
6 GO:0031122 cytoplasmic microtubule organization

1
2 The process whose specific outcome is the progression of the neural plate over time, from its formation to the mature structur
3
4
5
6
  Ontology count.InDataset count.InGenome      pvalue totaltermInDataset
1       BP               1             28 0.008619994                405
2       BP               1             11 0.003395307                405
3       BP               1             22 0.006779114                405
4       BP               1             23 0.007086164                405
5       BP               1             28 0.008619994                405
6       BP               1             32 0.009845356                405
  totaltermInGenome EntrezID
1           1310084     1855
2           1310084     1855
3           1310084     1855
4           1310084     1855
5           1310084     1855
6           1310084     1855
```

```
> head(over[["cc"]])
```

```
       go.id                go.term
1 GO:0016328 lateral plasma membrane

1 The portion of the plasma membrane at the lateral side of the cell. In epithelial cells, lateral plasma membranes are on the s
  Ontology count.InDataset count.InGenome      pvalue totaltermInDataset
1       CC               1             48 0.008016845                 61
  totaltermInGenome EntrezID
1            363819     1855
```

```
> head(over[["mf"]])
```

```
       go.id           go.term
1 GO:0005109   frizzled binding
2 GO:0017048 Rho GTPase binding
3 GO:0048365 Rac GTPase binding

1
2 Interacting selectively and non-covalently with Rho protein, any member of the Rho subfamily of the Ras superfamily of monomer
3                                                                                                                          Interact
  Ontology count.InDataset count.InGenome      pvalue totaltermInDataset
1       MF               1             37 0.003861301                 24
2       MF               1             71 0.007396923                 24
3       MF               1             32 0.003340340                 24
  totaltermInGenome EntrezID
1            229560     1855
2            229560     1855
```

```
3            229560     1855
```

Please note that org.Hs.eg.db is the GO gene mapping for Human, for other organisms, please refer to http://www.bioconductor.org/packages/release/data/annotation/ for additional org.xx.eg.db packages. Or you can try `egOrgMap` to get the annotation database.

```
> egOrgMap("Mus musculus")

[1] "org.Mm.eg.db"

> egOrgMap("Homo sapiens")

[1] "org.Hs.eg.db"
```

3.5 Task 5: Find peaks with bi-directional promoters

Here is an example to find peaks with bi-directional promoters and output percent of peaks near bi-directional promoters.

```
> data(myPeakList)
> data(TSS.human.NCBI36)
> annotatedBDP <- peaksNearBDP(myPeakList[1:10,],
+                              AnnotationData=TSS.human.NCBI36,
+                              MaxDistance=5000,
+                              PeakLocForDistance="middle",
+                              FeatureLocForDistance="TSS")
> annotatedBDP$peaksWithBDP

GRanges object with 6 ranges and 9 metadata columns:
                               seqnames             ranges strand |          peak
                                  <Rle>          <IRanges>  <Rle> |   <character>
  X1_14_1300250.ENSG00000218550     chr1 [1300503, 1300603]      * | X1_14_1300250
  X1_14_1300250.ENSG00000175756     chr1 [1300503, 1300603]      * | X1_14_1300250
   X1_41_559455.ENSG00000212678     chr1 [ 559774,  559874]      * |  X1_41_559455
   X1_41_559455.ENSG00000209350     chr1 [ 559774,  559874]      * |  X1_41_559455
   X1_93_556427.ENSG00000212875     chr1 [ 556660,  556760]      * |  X1_93_556427
   X1_93_556427.ENSG00000209349     chr1 [ 556660,  556760]      * |  X1_93_556427
                                       feature start_position end_position
                                   <character>      <integer>    <integer>
  X1_14_1300250.ENSG00000218550 ENSG00000218550        1303908      1304275
  X1_14_1300250.ENSG00000175756 ENSG00000175756        1298974      1300443
   X1_41_559455.ENSG00000212678 ENSG00000212678         559620       560165
   X1_41_559455.ENSG00000209350 ENSG00000209350         557860       557930
   X1_93_556427.ENSG00000212875 ENSG00000212875         556318       557859
   X1_93_556427.ENSG00000209349 ENSG00000209349         556240       556304
                               feature_strand insideFeature distancetoFeature
                                  <character>      <factor>         <numeric>
  X1_14_1300250.ENSG00000218550              +      upstream             -3355
  X1_14_1300250.ENSG00000175756              -      upstream              -110
   X1_41_559455.ENSG00000212678              +        inside               204
   X1_41_559455.ENSG00000209350              -      upstream             -1894
   X1_93_556427.ENSG00000212875              +        inside               392
   X1_93_556427.ENSG00000209349              -      upstream              -406
                               shortestDistance fromOverlappingOrNearest
                                      <integer>              <character>
  X1_14_1300250.ENSG00000218550            3305          NearestLocation
  X1_14_1300250.ENSG00000175756              60          NearestLocation
   X1_41_559455.ENSG00000212678             154          NearestLocation
   X1_41_559455.ENSG00000209350            1844          NearestLocation
   X1_93_556427.ENSG00000212875             342          NearestLocation
   X1_93_556427.ENSG00000209349             356          NearestLocation
  -------
  seqinfo: 24 sequences from an unspecified genome; no seqlengths
```

```
> c(annotatedBDP$percentPeaksWithBDP,
+   annotatedBDP$n.peaks,
+   annotatedBDP$n.peaksWithBDP)
```

```
[1]  0.3 10.0  3.0
```

3.6 Task 6: Output a summary of motif occurrence in the peaks.

Here is an example to search the peaks for the motifs in examplepattern.fa file.

```
> peaks <- GRanges(seqnames=c("NC_008253", "NC_010468"),
+                  ranges=IRanges(start=c(100, 500),
+                                 end=c(300, 600),
+                                 names=c("peak1", "peak2")))
> filepath <- system.file("extdata", "examplePattern.fa", package="ChIPpeakAnno")
> library(BSgenome.Ecoli.NCBI.20080805)
> summarizePatternInPeaks(patternFilePath=filepath, format="fasta", skip=0L,
+                         BSgenomeName=Ecoli, peaks=peaks)
```

```
     n.peaksWithPattern n.totalPeaks Pattern
[1,] "0"                "2"          "GGNCCK"
[2,] "1"                "2"          "AACCNM"
```

3.7 Task 7: Add other IDs to annotated peaks or enrichedGO

Here is an example to add gene symbol to annotated peaks .

```
> data(annotatedPeak)
> library(org.Hs.eg.db)
> addGeneIDs(annotatedPeak[1:6,], orgAnn="org.Hs.eg.db", IDs2Add=c("symbol"))
```

```
GRanges object with 6 ranges and 9 metadata columns:
                                  seqnames                 ranges strand |          peak
                                     <Rle>              <IRanges>  <Rle> |   <character>
  X1_11_100272487.ENSG00000202254        1 [100272801, 100272900]      + | 1_11_100272487
  X1_11_108905539.ENSG00000186086        1 [108906026, 108906125]      + | 1_11_108905539
  X1_11_110106925.ENSG00000065135        1 [110107267, 110107366]      + | 1_11_110106925
  X1_11_110679983.ENSG00000197106        1 [110680469, 110680568]      + | 1_11_110679983
  X1_11_110681677.ENSG00000197106        1 [110682125, 110682224]      + | 1_11_110681677
  X1_11_110756560.ENSG00000116396        1 [110756823, 110756922]      + | 1_11_110756560
                                         feature start_position end_position
                                     <character>      <numeric>    <numeric>
  X1_11_100272487.ENSG00000202254 ENSG00000202254      100257218    100257309
  X1_11_108905539.ENSG00000186086 ENSG00000186086      108918435    109013624
  X1_11_110106925.ENSG00000065135 ENSG00000065135      110091233    110136975
  X1_11_110679983.ENSG00000197106 ENSG00000197106      110693108    110744824
  X1_11_110681677.ENSG00000197106 ENSG00000197106      110693108    110744824
  X1_11_110756560.ENSG00000116396 ENSG00000116396      110753965    110776666
                                  insideFeature distancetoFeature shortestDistance
                                    <character>         <numeric>        <numeric>
  X1_11_100272487.ENSG00000202254    downstream             15582            15491
  X1_11_108905539.ENSG00000186086      upstream            -12410            12310
  X1_11_110106925.ENSG00000065135        inside             16033            16033
  X1_11_110679983.ENSG00000197106      upstream            -12640            12540
  X1_11_110681677.ENSG00000197106      upstream            -10984            10884
  X1_11_110756560.ENSG00000116396        inside              2857             2857
```

```
                                  fromOverlappingOrNearest   symbol
                                               <character> <factor>
  X1_11_100272487.ENSG00000202254             NearestStart     <NA>
  X1_11_108905539.ENSG00000186086             NearestStart    NBPF6
  X1_11_110106925.ENSG00000065135             NearestStart    GNAI3
  X1_11_110679983.ENSG00000197106             NearestStart  SLC6A17
  X1_11_110681677.ENSG00000197106             NearestStart  SLC6A17
  X1_11_110756560.ENSG00000116396             NearestStart    KCNC4
  -------
  seqinfo: 24 sequences from an unspecified genome; no seqlengths
```

```
> addGeneIDs(annotatedPeak$feature[1:6], orgAnn="org.Hs.eg.db", IDs2Add=c("symbol"))
```

```
  ensembl_gene_id  symbol
1 ENSG00000065135   GNAI3
2 ENSG00000116396   KCNC4
3 ENSG00000197106 SLC6A17
4 ENSG00000186086   NBPF6
5 ENSG00000202254    <NA>
```

3.8 Task 8: annotate ChIP results from BED or GFF files or MACS output xls file

Here is an example to annotate peaks in BED file format and GFF file format.

```
> bed <- system.file("extdata", "MACS_output.bed", package="ChIPpeakAnno")
> gr1 <- toGRanges(bed, format="BED", header=FALSE)
> ## one can also try import from rtracklayer
> library(rtracklayer)
> gr1.import <- import(bed, format="BED")
> identical(start(gr1), start(gr1.import))
```

```
[1] TRUE
```

```
> gr1[1:2]
```

```
GRanges object with 2 ranges and 1 metadata column:
              seqnames         ranges strand |     score
                 <Rle>      <IRanges>  <Rle> | <numeric>
  MACS_peak_1     chr1 [28341, 29610]      * |    160.81
  MACS_peak_2     chr1 [90821, 91234]      * |    133.12
  -------
  seqinfo: 1 sequence from an unspecified genome; no seqlengths
```

```
> gr1.import[1:2] #note the name slot is different from gr1
```

```
GRanges object with 2 ranges and 2 metadata columns:
      seqnames         ranges strand |        name     score
         <Rle>      <IRanges>  <Rle> | <character> <numeric>
  [1]     chr1 [28341, 29610]      * | MACS_peak_1    160.81
  [2]     chr1 [90821, 91234]      * | MACS_peak_2    133.12
  -------
  seqinfo: 1 sequence from an unspecified genome; no seqlengths
```

```
> gff <- system.file("extdata", "GFF_peaks.gff", package="ChIPpeakAnno")
> gr2 <- toGRanges(gff, format="GFF", header=FALSE, skip=3)
> ol <- findOverlapsOfPeaks(gr1, gr2)
> makeVennDiagram(ol)
```

```
$p.value
     gr1 gr2 pval
[1,]   1   1    0
```

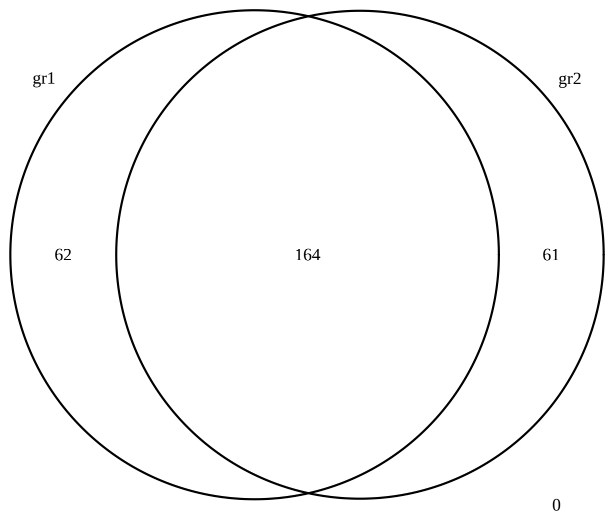

Figure 7: venn diagram of overlaps for duplicated experiments

```
$vennCounts
     gr1 gr2 Counts
[1,]   0   0      0
[2,]   0   1     61
[3,]   1   0     62
[4,]   1   1    164
attr(,"class")
[1] "VennCounts"
```

```
> pie(table(ol$overlappingPeaks[["gr1///gr2"]]$overlapFeature))
```

Find all features within 5kb away from the overlapping peaks using `annotatePeakInBatch`.

```
> data(TSS.human.GRCh37)
> overlaps <- ol$peaklist[["gr1///gr2"]]
> overlaps.anno <- annotatePeakInBatch(overlaps, AnnotationData=TSS.human.GRCh37,
+                                       output="overlapping", maxgap=5000L)
> overlaps.anno <- addGeneIDs(overlaps.anno, "org.Hs.eg.db", "symbol")
> head(overlaps.anno)
```

```
GRanges object with 6 ranges and 11 metadata columns:
                      seqnames           ranges strand |
                         <Rle>        <IRanges>  <Rle> |
  X001.ENSG00000228327     chr1 [713791, 715578]      * |
  X001.ENSG00000237491     chr1 [713791, 715578]      * |
  X001.ENSG00000242937     chr1 [713791, 715578]      * |
  X002.ENSG00000237491     chr1 [724851, 727191]      * |
  X002.ENSG00000242937     chr1 [724851, 727191]      * |
  X002.ENSG00000197049     chr1 [724851, 727191]      * |
                                                         peakNames        peak
                                                     <CharacterList> <character>
  X001.ENSG00000228327 gr1__MACS_peak_13,gr2__region_0,gr2__region_1         001
  X001.ENSG00000237491 gr1__MACS_peak_13,gr2__region_0,gr2__region_1         001
  X001.ENSG00000242937 gr1__MACS_peak_13,gr2__region_0,gr2__region_1         001
```

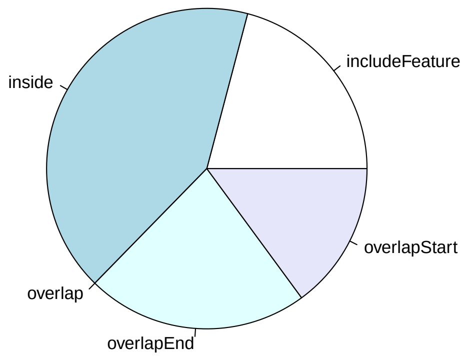

Figure 8: Pie chart of common peaks among features

```
  X002.ENSG00000237491               gr2__region_2,gr1__MACS_peak_14         002
  X002.ENSG00000242937               gr2__region_2,gr1__MACS_peak_14         002
  X002.ENSG00000197049               gr2__region_2,gr1__MACS_peak_14         002
                               feature start_position end_position feature_strand
                           <character>      <integer>    <integer>    <character>
  X001.ENSG00000228327 ENSG00000228327         700238       714006              -
  X001.ENSG00000237491 ENSG00000237491         714163       740255              +
  X001.ENSG00000242937 ENSG00000242937         717326       720070              +
  X002.ENSG00000237491 ENSG00000237491         714163       740255              +
  X002.ENSG00000242937 ENSG00000242937         717326       720070              +
  X002.ENSG00000197049 ENSG00000197049         721321       722513              +
                       insideFeature distancetoFeature shortestDistance
                            <factor>         <numeric>        <integer>
  X001.ENSG00000228327  overlapStart               215              215
  X001.ENSG00000237491  overlapStart              -372              372
  X001.ENSG00000242937      upstream             -3535             1748
  X002.ENSG00000237491        inside             10688            10688
  X002.ENSG00000242937    downstream              7525             4781
  X002.ENSG00000197049    downstream              3530             2338
                       fromOverlappingOrNearest                    symbol
                                    <character>                  <factor>
  X001.ENSG00000228327              Overlapping LOC100288069;LOC101929540
  X001.ENSG00000237491              Overlapping              LOC100287934
  X001.ENSG00000242937              Overlapping                      <NA>
  X002.ENSG00000237491              Overlapping              LOC100287934
  X002.ENSG00000242937              Overlapping                      <NA>
  X002.ENSG00000197049              Overlapping                      <NA>
  -------
  seqinfo: 1 sequence from an unspecified genome; no seqlengths
```

Plot the distribution of aggregated peak scores or peak numbers around transcript start sites (Figure 9).

```
> gr1.copy <- gr1
> gr1.copy$score <- 1
```

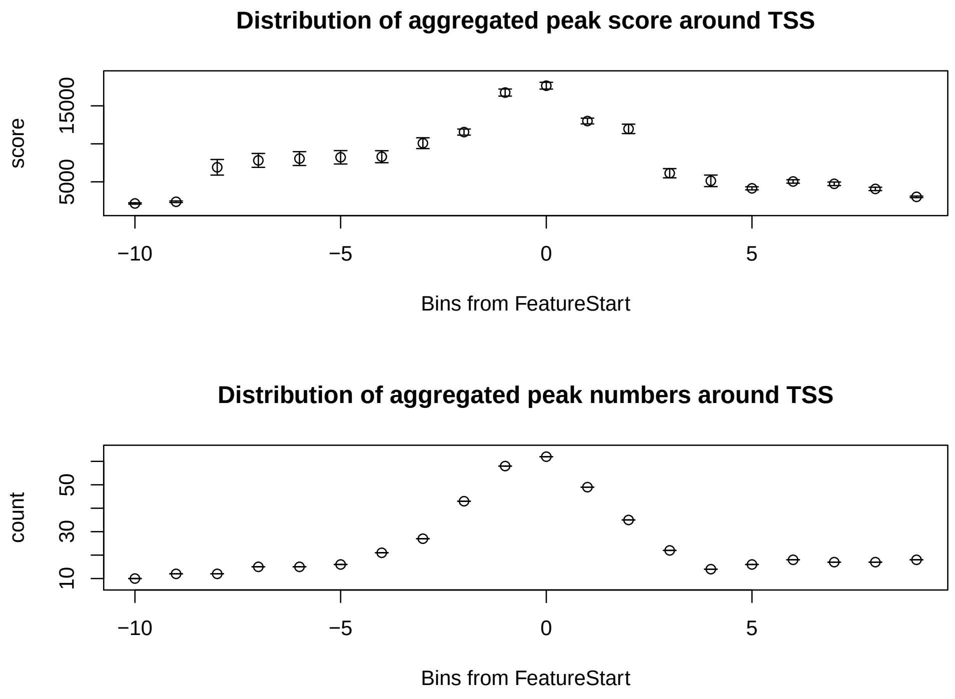

Figure 9: Distribution of aggregated peak scores or peak numbers around transcript start sites.

```
> binOverFeature(gr1, gr1.copy, annotationData=TSS.human.GRCh37,
+                radius=5000, nbins=10, FUN=c(sum, length),
+                ylab=c("score", "count"),
+                main=c("Distribution of aggregated peak score around TSS",
+                       "Distribution of aggregated peak numbers around TSS"))
```

Summarize peak distribution over exon, intron, enhancer, proximal promoter, 5 prime UTR and 3 prime UTR in peak centric and nucleotide centric view using function `assignChromosomeRegion`(Figure 10). Setting nucleotideLevel = TRUE will give a nucleotide level distribution over different features.

```
> if(require(TxDb.Hsapiens.UCSC.hg19.knownGene)){
+     aCR<-assignChromosomeRegion(gr1, nucleotideLevel=FALSE,
+                            precedence=c("Promoters", "immediateDownstream",
+                                         "fiveUTRs", "threeUTRs",
+                                         "Exons", "Introns"),
+                            TxDb=TxDb.Hsapiens.UCSC.hg19.knownGene)
+     barplot(aCR$percentage)
+ }
```

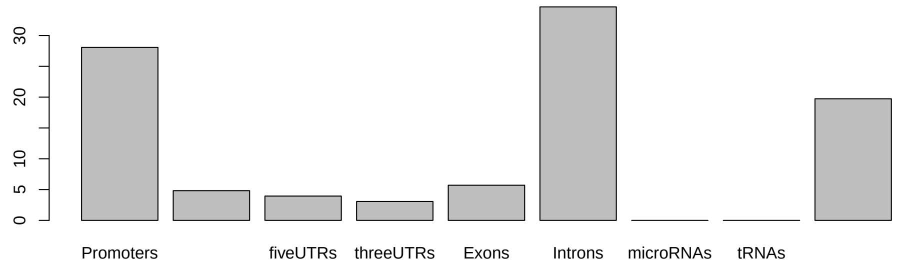

Figure 10: Peak distribution over different genomic features.

4 References

1. Y. Benjamini and Y. Hochberg (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. J. R. Statist. Soc. B. Vol. 57: 289-300.
2. Y. Benjamini and D. Yekutieli (2001). The control of the false discovery rate in multiple hypothesis testing under dependency. Annals of Statistics. Accepted.
3. S. Durinck et al. (2005) BioMart and Bioconductor: a powerful link between biological biomarts and microarray data analysis. Bioinformatics, 21, 3439-3440.
4. S. Dudoit, J. P. Shaffer, and J. C. Boldrick (Submitted). Multiple hypothesis testing in microarray experiments.
5. Y. Ge, S. Dudoit, and T. P. Speed. Resampling-based multiple testing for microarray data hypothesis, Technical Report #633 of UCB Stat. http://www.stat.berkeley.edu/ gyc
6. R. Gentleman et al. (2004) Bioconductor: open software development for computational biology and bioinformatics. Genome Biol., 5:R80
7. Y. Hochberg (1988). A sharper Bonferroni procedure for multiple tests of significance, Biometrika. Vol. 75: 800-802.
8. S. Holm (1979). A simple sequentially rejective multiple test procedure. Scand. J. Statist.. Vol. 6: 65-70.
9. N. L. Johnson,S. Kotz and A. W. Kemp (1992) Univariate Discrete Distributions, Second Edition. New York: Wiley
10. G. Robertson et al. (2007) Genome-wide profiles of STAT1 DNA association using chromatin immunoprecipitation and massively parallel sequencing. Nat Methods, 4:651-7.
11. Zhu L.J. et al. (2010) ChIPpeakAnno: a Bioconductor package to annotate ChIP-seq and ChIP-chip data. BMC Bioinformatics 2010, 11:237doi:10.1186/1471-2105-11-237.
12. Zhu L.J. (2013) Integrative analysis of ChIP-chip and ChIP-seq dataset. Methods Mol Biol. 2013;1067:105-24. doi: 10.1007/978-1-62703-607-8_8.

5 Session Info

```
> toLatex(sessionInfo())
```

- R version 3.2.0 (2015-04-16), `x86_64-unknown-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Base packages: base, datasets, grDevices, graphics, grid, methods, parallel, stats, stats4, utils
- Other packages: AnnotationDbi 1.30.1, BSgenome 1.36.0, BSgenome.Ecoli.NCBI.20080805 1.3.1000, Biobase 2.28.0, BiocGenerics 0.14.0, Biostrings 2.36.1, ChIPpeakAnno 3.2.2, DBI 0.3.1, FDb.UCSC.tRNAs 1.0.1, GenomeInfoDb 1.4.0, GenomicFeatures 1.20.1, GenomicRanges 1.20.3, IRanges 2.2.1, RSQLite 1.0.0, S4Vectors 0.6.0, TxDb.Hsapiens.UCSC.hg19.knownGene 3.1.2, VennDiagram 1.6.9, XVector 0.8.0, biomaRt 2.24.0, mirbase.db 1.2.0, org.Hs.eg.db 3.1.2, rtracklayer 1.28.2
- Loaded via a namespace (and not attached): BiocInstaller 1.18.2, BiocParallel 1.2.1, BiocStyle 1.6.0, GO.db 3.1.2, GenomicAlignments 1.4.1, MASS 7.3-40, RBGL 1.44.0, RCurl 1.95-4.6, Rsamtools 1.20.2, XML 3.98-1.1, bitops 1.0-6, futile.logger 1.4.1, futile.options 1.0.0, graph 1.46.0, lambda.r 1.1.7, limma 3.24.4, multtest 2.24.0, splines 3.2.0, survival 2.38-1, tools 3.2.0, zlibbioc 1.14.0