ALTRE: vignette

Ewy Mathe,..

2016-06-27

Introduction

These regulatory regions modulate transcription and affect cellular development using a variety of mechanism: DNA looping, the production of enhancer RNAs (eRNAs), and enhancer-promoter interactions among them. Identifying altered regulatory regions between cell lines and tissue types holds promise for a variety of applications, including identifying new mechanisms involved in disease progression, investigating how subtypes of broader diseases differ from one another, or focusing on those regulatory regions that do not change (to better understand the housekeeping functions of cells).

Assays such as ATAC-seq, DHS-seq and FAIRE-seq can identify regulatory region activity by identifying open, transcriptionally active chromatin. These assays produce files of known peak locations (called hotspot files). The hotspots are regions of open chromatin marking promoters and enhancers. While these assays are more and more frequently performed in cell lines and clinical samples, there are currently few resources available to guide the raw data to meaningful results. This package is the first to bundle the various scripts and steps of regulatory region analysis into a single, compact tool that guides users through a defined workflow. Previously, individual data analyses pipelines were strung together as needs arose, making reproducible, shareable data-analysis a challenge – an especially difficult challenge for newcomers to the field, especially those with little programming skill. This package is the solution to these problems, and provides two additional benefits:

1. The level of regulatory region accessibility across cell types is incorporated into the analysis – we suggust this is a more robust method for identifying altered regions than relying on hotspot files alone. Hotspot files mark only whether a peak is present or absent, they contain no information on the level of accessibility.
2. Altered regulatory regions are linked to genes and pathways of interest – the majority of previous studies have predominately focused only on identifying the number of altered regions, stopping short of this kind of pathway analysis.

This vignette further explains the purpose of the package and establishes the workflow.

Workflow

Before beginning

Input data

Both ATAC-seq, DHS-seq and FAIRE-seq are high-throughput sequencing techniques for identifying regions of open, transcriptionally active chromatin. As a result, short sequencing fragments will reflect open chromatin, enabling researchers to easily distinguish between open and closed chromatin. Open chromatin will generate peaks for analysis, closed chromatin will not. While ATAC-seq, DHS-seq and FAIRE-seq share many similarities, ATAC-seq has gained momentum recently, as it requires significantly fewer cells, takes less time, and is less labor-intensive.

As for all high-throughput sequencing experiments, the results of ATAC-seq, DHS-seq and FAIRE-seq experiments are delivered in FASTQ files. The initial processing of the FASTQ file must be accomplished

outside of ALTRE, as the R-package requires BAM and hotspot files as inputs. Alignment of sequencing reads produces BAM files, and peak identification produces hotspot files. There are a number of software packages available for both alignment and hotspot calling; we recommend bowtie for alignment, and dnase2hotspot for hotspot calling.

ALTRE builds a number of packages

Example data

To demonstrate the abilities of ALTRE, we have provided an example dataset (BAM and HOTSPOT files) of cancerous (A549) and normal (SAEC) lung cell types downloaded from the ENCODE project. This dataset will guide users through the workflow of the package in this vignette.

Start of pipeline

Data Preparation:

Reproducible peaks

The first step in the workflow is the function "hotspotreproduc", which processes an input of two or more hotspot files per sample and returns one GRanges object per sample. Only reproducible hotspots are kept – reproducible hotspots are hotspots present in every sample replicate. At least two replicates are required in order to differentiate sample noise from robust, reproducible peaks. Hotspot files must be formatted as "chr", "start", "stop" in the first three columns. Additional columns are allowed, but will be ignored.

The example data is provided with the R package. Here is the command to find the directory of example data:

A csv file with the names of the hotspot file replicates (and the sample each replicate corresponds to) is the only argument to the "hotspotreproduc" function. The csv file must be in the current directory, or there must be a clear path to it's location:

```
library(ALTRE)
```

```
##
```

```
##
```

```
dir <- system.file("extdata", package = "ALTRE", mustWork = TRUE)
csvfile <- file.path(dir, "lung.csv")
csvfile
```

```
## [1] "/home/rick/R/x86_64-pc-linux-gnu-library/3.3/ALTRE/extdata/lung.csv"
```

The csv also contains the names of the bam files for later use in the workflow. To see the correct format of the file:

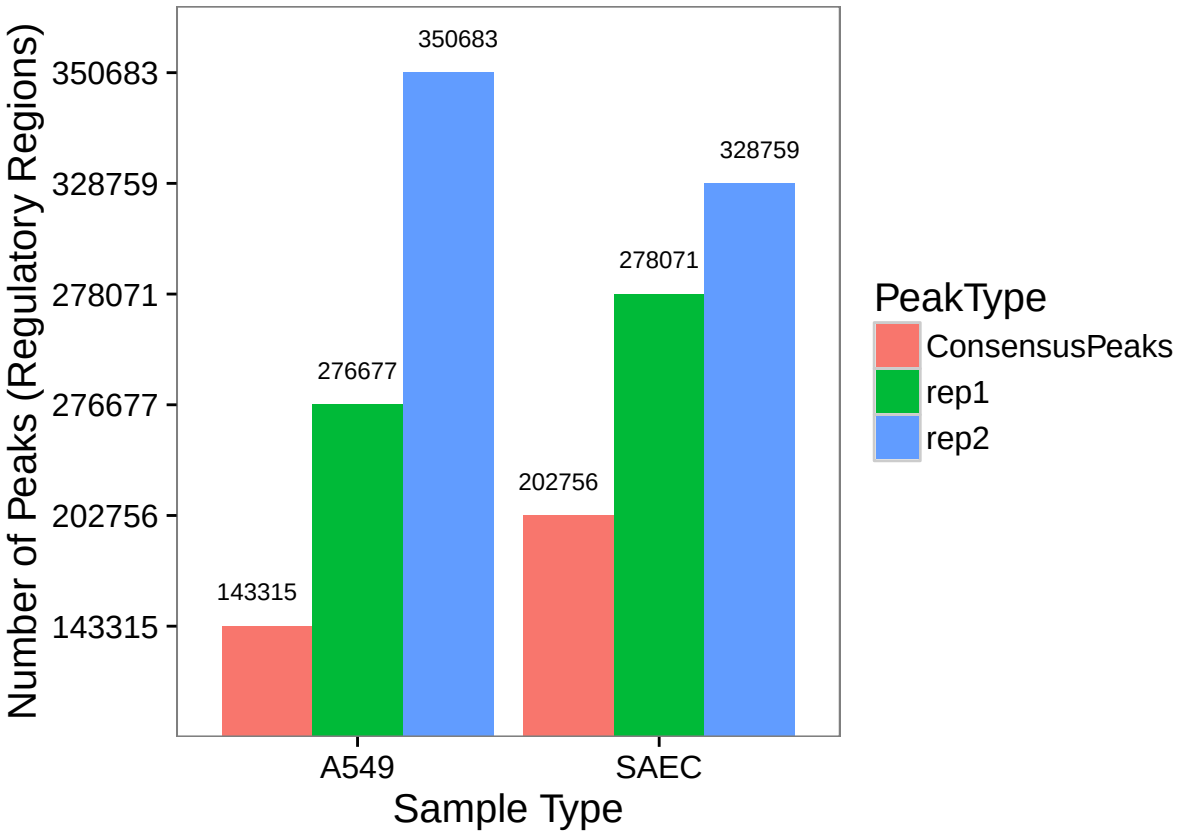
```
sampleinfo <- loadCSVFile(csvfile)
samplePeaks <- loadBedFiles(sampleinfo)
samplePeaks
```

```
## GRangesList object of length 4:
## $A549_I
## GRanges object with 276677 ranges and 2 metadata columns:
##            seqnames               ranges strand |   sample replicate
##               <Rle>            <IRanges>  <Rle> | <factor>  <factor>
##        [1]     chr1     [ 10241,  10349]      * |     A549         I
##        [2]     chr1     [237719, 237872]      * |     A549         I
##        [3]     chr1     [564496, 564831]      * |     A549         I
##        [4]     chr1     [565253, 566084]      * |     A549         I
##        [5]     chr1     [566586, 567276]      * |     A549         I
##        ...      ...                  ...    ... .      ...       ...
##   [276673]     chrY [59024237, 59024354]      * |     A549         I
##   [276674]     chrY [59027624, 59027997]      * |     A549         I
##   [276675]     chrY [59028579, 59028819]      * |     A549         I
##   [276676]     chrY [59029626, 59029819]      * |     A549         I
##   [276677]     chrY [59031344, 59031507]      * |     A549         I
##
## ...
## <3 more elements>
## -------
## seqinfo: 24 sequences from an unspecified genome; no seqlengths
```

The hotspot files must be in the same directory as the csv file.

```
consPeaks <- getConsensusPeaks(samplepeaks = samplePeaks, minreps = 2)
plotConsensusPeaks(samplepeaks = consPeaks)
```

```
## Warning: attributes are not identical across measure variables; they will
## be dropped
```



A list of GRanges objects will be returned. One GRanges object per sample. Each Granges object contains only the reproducible hotspots for that sample type. To see the list of GRanges use the "results" function:

```
#getresults(consPeaks)
```

To see the number of hotspots in each replicate and the number considered "reproducible", use the metrics function.

```
#metrics(consPeaks)
```

Regulatory region annotation

The output of "hotspotreproduc" is then further processed by the "annotateandmerge" function, which accomplishes three main tasks:

1. Combines the reproducible hotspots from each sample type into one large master list of hotspots. This enables peak height comparison across sample types.
2. Categorize the hotspots as either promoters or enhancers based on the distance of the regulatory region from a transcription start site (TSS). In this vignette we will use the default promoter distance argument: regulatory regions within 1,500 bp of a TSS are considered promoters; those not within 1,500 bp of a TSS are considered enhancers.
3. Optionally, merges regulatory regions within a certain distance of each other. Merging close-by regulatory regions loosens the stringency of the region comparison. Multiple regions of active chromatin within ~1000 bp are likely to represent only one regulatory region. In this vignette we will use the defaults of the

function, which merges regions within 1000 bp of each other and merges only within regulatory region type (i.e. only enhancers will only be merged with other enhancers, and likewise, promoters only with promoters). Both these functionalities can be changed with the "mergedistance" and "reregionspecific" arguments.

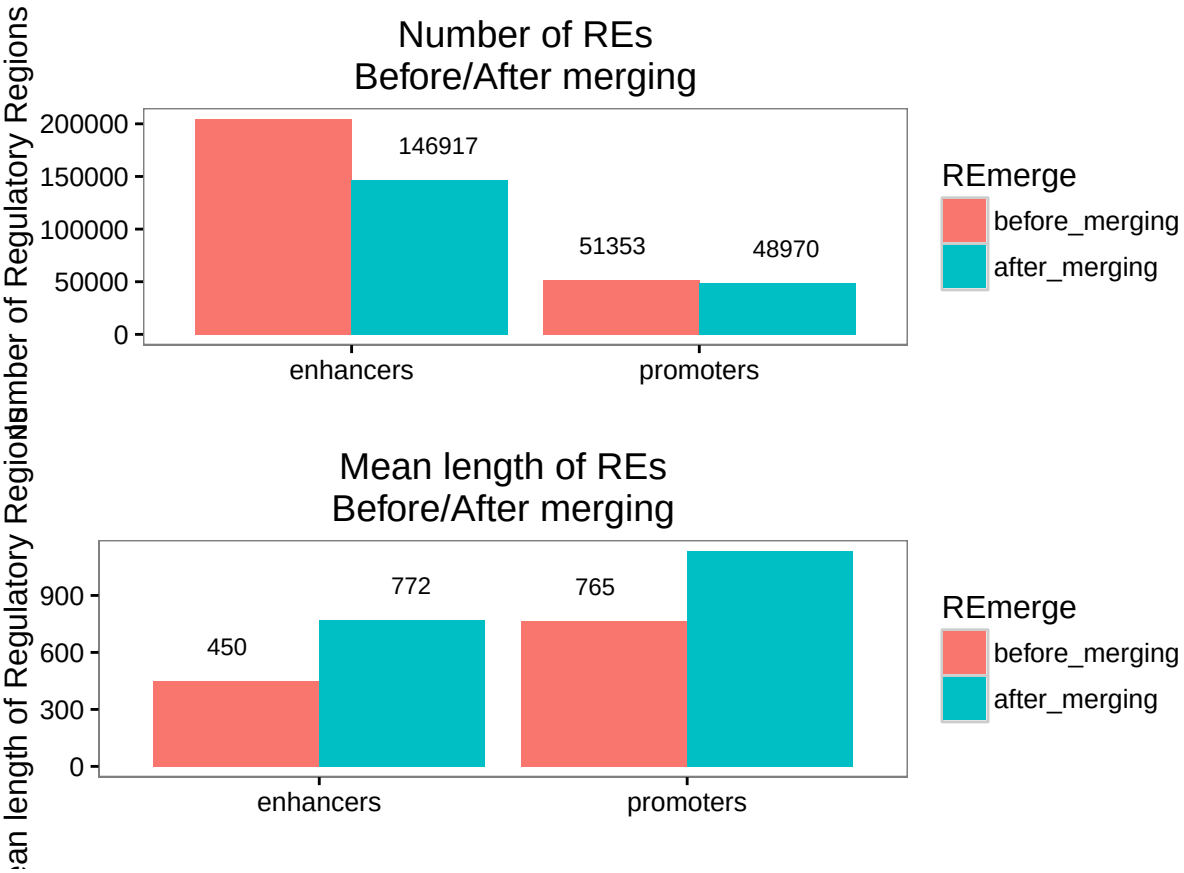
This function requires a list of TSSs for the promoter/enhancer annotation:

```
TSSannot <- getTSS()
```

The output of the previous function, "hotspotreproduc", is fed into this function:

```
consPeaksAnnotated <- combineAnnotatePeaks(conspeaks = consPeaks,
                                           TSS = TSSannot,
                                           merge = TRUE,
                                           regionspecific = TRUE,
                                           mergedistenh = 1500,
                                           mergedistprom = 1000)

plotCombineAnnotatePeaks(consPeaksAnnotated)
```



The function produces a GRanges object containing the annotated, merged hotspots – all hotspots present in cancerous or normal lung. To see this GRanges object use the function "results":

```
#getresults(consPeaksAnnotated)
```

Additionally, this function keeps track of the changes in the number and size of enhancers and promoters before and after merging close-by regulatory regions. To see this analysis of how the regulatory regions changed due to merging use the "metrics" function:

```
#getmetrics(consPeaksAnnotated)
```

The number of regulatory regions decreases and the size of the regions increases, as expected.

Quantifying regulatory region acessibility

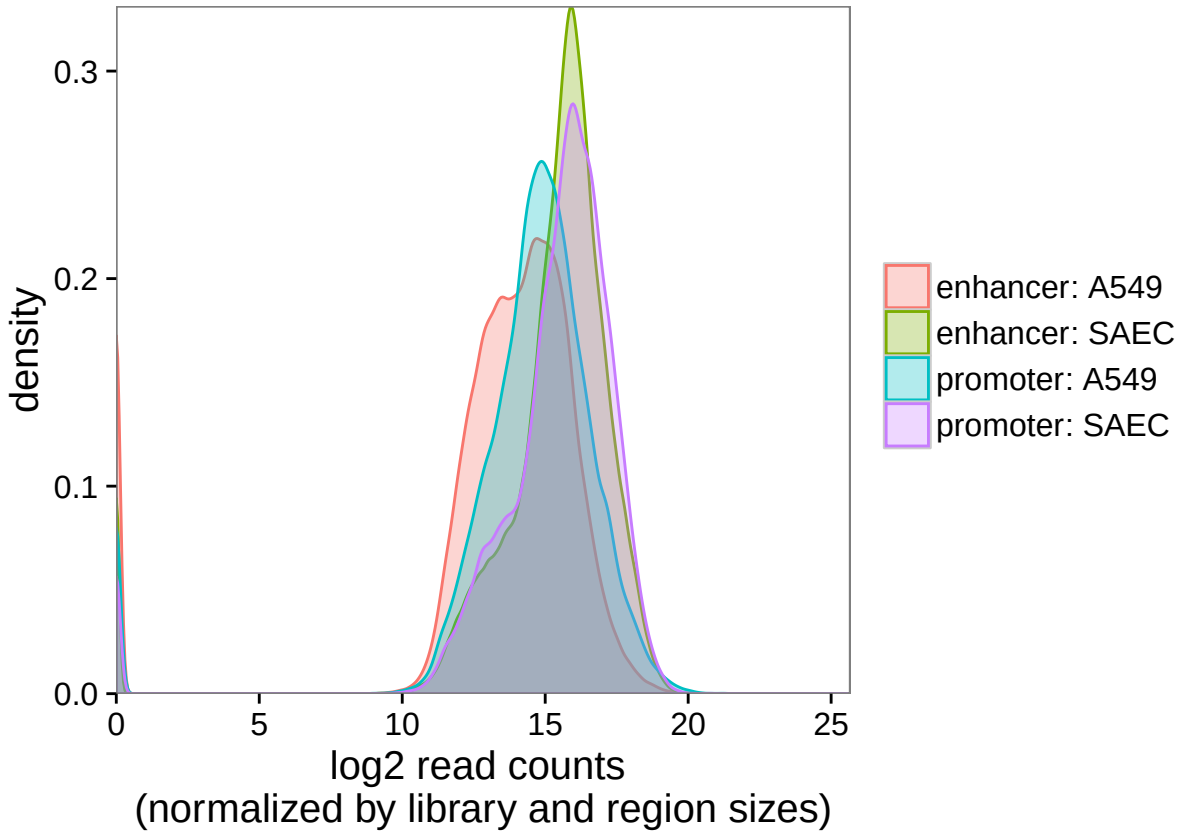
The next function, "getcounts", has two purposes:

1. Counts the number of reads in each regulatory region for each sample type – this determines the peak height/intensity and approximates the accessibility of the regulatory region in question.
2. Enables filtering of regions with overall low read counts across all sample types (those below a certain percentile of all reads). The removal of these regions will decrease "false positives" in the results: regions that are identified as significantly altered between cell and tissue types, but do not truly have enough read count information to make a definitive call either way. In this vignette, lung data regions with reads below the 50th percentile in all samples are filtered out, but any percentile can be supplied with the percentile argument.

Read count information across regions of interest is only available in the BAM file format – BAM files must be supplied to this function for each sample type. At least two replicates per sample type are required in order to distinguish reproducible peaks from sample noise. The names of the BAM files are not supplied to the function directly. Instead, a CSV file is created with the file names and sample type information. That CSV file and the directory location of the BAM files is supplied to the function.

One cell type must be designated as the "reference type", to which the other cell types will be compared. In this example, it is the normal lung cell line (SAEC).

```
 counts_consPeaks <- getcounts(annotpeaks = consPeaksAnnotated,
                               sampleinfo = sampleinfo,
                               reference = 'SAEC')
plotgetcounts(counts_consPeaks)
```



This function produces a dataframe containing count information for all replicates of all samples. To access this information use the "results" function:

```
#getresults(counts_consPeaks)
```

Additionally, this function keeps track of the number of enhancers and promoters before and after filtering. To access this information use the "metrics" function:

```
#getmetrics(counts_consPeaks)
```

At the 50th percentile, the number of enhancers and promoters decreased ~20-25%.

Finally, this function creates a density plot of counts (after standardizing samples by library size, regulatory regions by region size, and log2 transforming to create a normalized distribution from the count data). This visualization helps identify how similar or different the distributions are between sample types and regulatory region types. To see the plot, use the "seeplot" function

```
#seeplot(counts_lung)
```

Here, the samples have very similar distributions, with enhancers and promoters differing only slightly.

Identification of cell-type specific regulatory regions

The count information from "getcounts" is then processed by "countanalysis" – this is the point in the pathway where the altered enhancers and promoters are identified. The function uses an algorithm from the DESeq2 R package for identification of altered regulatory regions.

```
altre_peaks <- countanalysis(counts = counts_consPeaks,
                             pval = 0.01,
                             lfcvalue = 1)
```

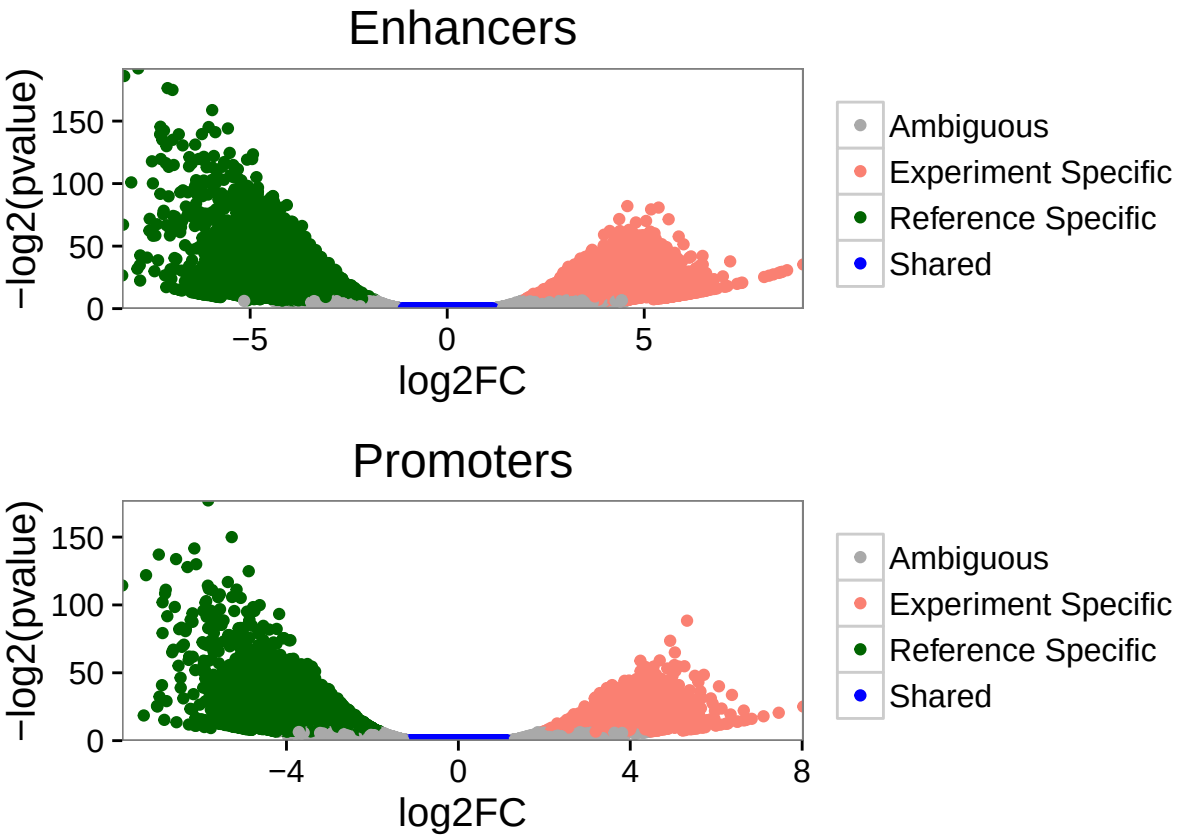
```
categaltre_peaks <- categAltrePeaks(altre_peaks,
                                    lfctypespecific = 1.5,
                                    lfcshared = 1.2,
                                    pvaltypespecific = 0.01,
                                    pvalshared = 0.05)
```

```
## [1] TRUE
##     REtype Expt_Specific Ref_Specific Shared Ambiguous
## 1 Promoter          2428         5091  17073     24378
## 2 Enhancer         11888         8524  46155     80350
```

```
plotCountAnalysis(categaltre_peaks)
```

```
## Warning: Removed 55940 rows containing missing values (geom_point).
```

```
## Warning: Removed 12425 rows containing missing values (geom_point).
```




```
## NULL
```

The results are p-values, log2fold, and other statistics for every region.

```
#getresults(altre_peaks)
```

Regulatory regions with large, positive log2fold changes and small p-values are more open in the lung cancer cell type, and therefore, their activity is associated with the cancer phenotype. Regulatory regions with large, negative log2fold changes and small p-values are more open in the normal lung cell type, and may therefore be associated with tumor suppressive properties. Regulatory regions with minimal log2fold change and high p-values are equally open in both cell types, and therefore, their activity is more likely to contribute to the house-keeping functions required in every cell.

Pathway enrichment

Finally, the results of the "countanalysis" can be used to identify whether there is an enrichment of certain pathways in the type-specific or shared regions using the "enrichment" function. This is accomplished by linking each regulatory region to the closest gene, and then linking each gene's product to the pathway the gene product participates in. Pathways that recur many times in the gene cluster ("enriched" pathways) are likely to be regulated by the regulatory regions linked to the gene cluster (whether type-specific or shared).

Pathways that are unlikely to be of interest can be filtered out of the enrichment results. ALTRE has two metrics to identify "low information" pathways.

1. Vague pathways such as "DNA binding" are removed – they encompass a number of other, more precise pathways (termed "offspring" pathways) already present in the enrichment results. The "offspring" argument limits how many offspring a pathway can contain. Generally, the more offspring, the vaguer and less informative the pathway.
2. Removal or pathways with low gene counts – these pathways are more likely to be false positives. This is because they are more likely to be elevated in the pathway rankings due to the nature of the statistical test – it is much easier to contain half of a pathway's genes in the gene cluster under analysis when the pathway contains only four genes, and not 100 genes. The "gene" argument limits how few genes a pathway can contain.

In order to link regulatory regions to genes, TSSs must be supplied for this function as well. The TSSs file was previously used in the "annotateandmerge" function, and therefore, should already exist in the R environment.

Pathways fall into three categories (Molecular Function (MF), Biological Process (BP), and Cellular Component (CC)), and each category must be analyzed separately. The category is selected by supplying an argument to "ontoltype"

```
#  MFenrich <- pathenrich(analysisresults = altre_peaks,
#                         ontoltype = 'MF',
#                         enrichpvalfilt = 0.01)
#
# enrichHeatmap(MFenrich, title='GO:MF, p<0.01')
```

```
# BPenrich <- pathenrich(analysisresults = altre_peaks,
#                         ontoltype = 'BP',
#                         enrichpvalfilt = 0.01)
# enrichHeatmap(BPenrich, title = 'GO:BP, p<0.01')
```

The result is a list of dataframes containing the enriched pathways with p-values and some additional information. The first dataframe contains regulatory regions that are cancer-specific, the second dataframe

is normal-specific regions, and the third is shared regions. Here we take a look at the first line of the cancer-specific regions. To see these dataframes, use the results function:

```
#getresults(lung_MF)
```

Post-processing summerization and visualization

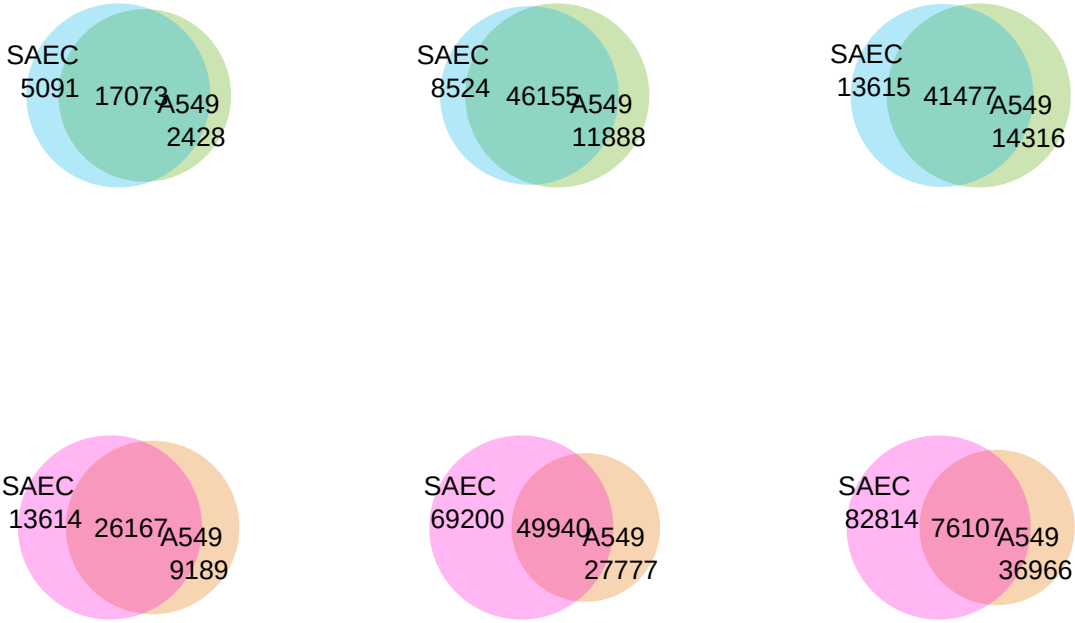
Once results have been obtained, there are a number of additional functions in ALTRE to summarize and better depict the results and raw data in tables and graphs. In this section of the vignette we will walk through these functions.

"resultscomparison" creates a table to compare the two methods of identifying altered regulatory regions – one based on peak intensity, the other on peak presence or absence as determined by hotspot calling algorithms. The intensity-based method identifies much fewer type-specific regions.

```
analysisresults <- resultsComparison(altre_peaks, reference= "SAEC")
```

"plotallvenn" takes the results from "resultscomparison" and translates the table into proportional venn diagrams.

```
plotallvenn(analysisresults)
```



"createboxplot" enables users to view the raw count data in regions identified as type-specific or shared. The log2 transformation of reads per kilobase of regulatory region per million is plotted. As expected, for regulatory regions identified as higher log-fold change than normal, the lung cancer cell line has much higher counts than he normal lung cell line. The opposite is true for the lower log-fold change regions. For regions with little log-fold change between cell types, the counts are similar. This visual depiction confirms the results of "countanalysis" and enables users to change parameters if needed.

```
#createboxplot(counts_lung, analysisresults_lung)
```

"genomebrowservis" allows raw data to be visualized as tracks in the UCSC by creating wig files as well as HOTSPOT files color-coded by type-specificity. Wig file creation involves translation of count data into a

peak width and height, which allows easier visualization of count-differences. A bed file will be created in the working directory, there is no output to the function that can be saved as an object.

```
#genomebrowvisual(analysisresults_lung)
```

"enrichmentheatmap" creates a heatmap of the analysis for all three regulatory region types (SAEC-specific, A549-specific, and shared), with the color-coding corresponding to the significance of the pathway – the lighter the blue, the lower the p-value.

```
#enrichheatmap(lung_MF, title="GO:MF, p<0.01")
```

Pathways that are identified as enriched in all three regulatory region types can be filtered out at this step.