CRISPR Screen and Gene Expression Differential Analysis

Lianbo Yu, Yue Zhao, Kevin R. Coombes, and Lang Li

2022-08-10

Contents

1 Introduction

We developed CEDA to analyze read counts of single guide RNAs (sgRNAs) from CRISPR screening experiments. Except for non-targeting sgRNAs (can be used as negative controls), each sgRNA is targeting one gene, and each gene have multiple sgRNAs targeting it. CEDA models the sgRNA counts at different levels of gene expression by multi-component normal mixtures, with the model fit by an EM algorithm. Posterior estimates at sgRNA level are then summarized for each gene.

In this document, we use data from an experiment with the MDA231 cell line to illustrate how to use CEDA to perform CRISPR screen data analysis.

2 Overview

CEDA analysis follows a workflow that is typical for most omics level experiments.

1. Put the data into an appropriate format for input to CEDA.
2. Filter out sgRNA with low read counts (optional).
3. Normalize the raw counts.
4. Fit a linear model to the data.
5. Summarize and view the results.

2.1 Data Format

In our experiment, three samples of MDA231 cells were untreated at time T=0, and another three samples of MDA231 cells were treated with DMSO at time T=0. We are interested in detecting sgRNAs that are differentially changed by a treatment.

The sgRNA read counts, along with a list of non-essential genes, are stored in the dataset `mda231` that we have included in the `CEDA` package. We read that dataset and explore its structure.

```
library(CEDA)
library(dplyr)
library(ggplot2)
```

```r
library(ggsci)
library(ggprism)
set.seed(1)
data("mda231")
class(mda231)
#> [1] "list"
length(mda231)
#> [1] 2
names(mda231)
#> [1] "sgRNA"  "neGene"
```

As you can see, this is a list containing two components

1. sgRNA, the observed count data of six samples, and
2. neGene, the set of non-essential genes.

```r
dim(mda231$sgRNA)
#> [1] 23618     9
length(mda231$neGene$Gene)
#> [1] 200
head(mda231$sgRNA)
#>                                  sgRNA  Gene DMSOa DMSOb DMSOc T0a T0b T0c
#> 64780   chr19:10655652-10655671_ATG4D_- ATG4D   126   100   132  94  82  78
#> 67381     chr5:32739109-32739128_NPR3_+  NPR3   266   452   309 557 687 587
#> 67411    chr3:45515731-45515750_LARS2_+ LARS2    28    45    36 583 660 512
#> 27053 chr12:111856039-111856058_SH2B3_+ SH2B3   509   501   661 578 824 636
#> 55806   chr9:118163517-118163536_DEC1_+  DEC1   265   489   390 718 733 655
#> 57274   chr1:228879268-228879287_RHOU_+  RHOU   144   124   137 160 164 119
#>       exp.level.log2
#> 64780    3.655866985
#> 67381    0.200236907
#> 67411    3.495375381
#> 27053    4.227348316
#> 55806    0.005794761
#> 57274    0.925993344
```

Notice that the sgRNA component includes an extra column, "exp.level.log2", that are the expression level (in log2 scale) of genes and was computed from gene expression data in the unit of FPKM.

The second component of the list mda231 is a data frame neGene, which is the gene names of the non-essential genes(Ref 1):

```r
dim(mda231$neGene)
#> [1] 200   1
head(mda231$neGene)
#>        Gene
#> 189    MMD2
#> 303    SUN5
#> 155    KRT2
#> 72    DEFA5
#> 195   MUC17
#> 70  CYP2C19
```

2.2 Filter out sgRNAs with low read counts

Due to the nature of drop-out screen, some sgRNAs targeting essential genes have low read counts at the end time point samples. Therefore, we must be cautious when applying strict filtering criteria to remove a large portion of sgRNAs before analysis. Gentle filtering criteria (i.e., removal of sgRNAs with 0 count in 75% samples) is recommended.

```
count_df <- mda231$sgRNA
mx.count = apply(count_df[,c(3:8)],1,function(x) sum(x>=1))
table(mx.count)
#> mx.count
#>     0     1     2     3     4     5     6
#>    70     8    14    30    24    41 23431
# keep sgRNA with none zero count in at least one sample
count_df2 = count_df[mx.count>=1,]
```

2.3 Normalization

The sgRNA read counts needs to be normalized across sample replicates before formal analysis. The non-essential genes are assumed to have no change after DMSO treatment. So, our recommended procedure is to perform median normalization based on the set of non-essential genes.

```
mda231.ne <- count_df2[count_df2$Gene %in% mda231$neGene$Gene,]
cols <- c(3:8)
mda231.norm <- medianNormalization(count_df2[,cols], mda231.ne[,cols])[[2]]
```

2.4 Analysis

Our primary goal is to detect essential sgRNAs that have different count levels between conditions. We rely on the R package `limma` to calculate log2 ratios (i.e., log fold changes or LFCs) between three untreated and three treated samples.

2.4.1 Calculating fold ratios

First, we have to go through the usual `limma` steps to describe the design of the study. There were two groups of replicate samples. We will call these groups "Control" and "Baseline" (although "Treated" and Untreated" would work just as well). Our main interest is determining the differences between the groups. And we have to record this information in a "contrast matrix" so limma knows what we want to compare.

```
group <- gl(2, 3, labels=c("Control","Baseline"))
design <- model.matrix(~  0 + group)
colnames(design) <- sapply(colnames(design), function(x) substr(x, 6, nchar(x)))
contrast.matrix <- makeContrasts("Control-Baseline", levels=design)
```

Finally, we can run the limma algorithm.

```
limma.fit <- runLimma(log2(mda231.norm+1),design,contrast.matrix)
```

We merge the results from our limma analysis with the post filtering sgRNA count data.

```
mda231.limma <- data.frame(count_df2, limma.fit)
head(mda231.limma)
#>                                  sgRNA  Gene DMSOa DMSOb DMSOc T0a T0b T0c
#> 64780   chr19:10655652-10655671_ATG4D_- ATG4D   126   100   132  94  82  78
#> 67381     chr5:32739109-32739128_NPR3_+  NPR3   266   452   309 557 687 587
#> 67411    chr3:45515731-45515750_LARS2_+ LARS2    28    45    36 583 660 512
#> 27053 chr12:111856039-111856058_SH2B3_+ SH2B3   509   501   661 578 824 636
```

```
#> 55806   chr9:118163517-118163536_DEC1_+  DEC1   265   489   390 718 733 655
#> 57274   chr1:228879268-228879287_RHOU_+  RHOU   144   124   137 160 164 119
#>       exp.level.log2        lfc        se           p
#> 64780    3.655866985  0.5292962 0.1092323 7.451322e-03
#> 67381    0.200236907 -0.8178899 0.3241114 1.806025e-02
#> 67411    3.495375381 -3.9445487 0.2747425 4.917981e-07
#> 27053    4.227348316 -0.2360867 0.1258984 1.612023e-01
#> 55806    0.005794761 -0.8759862 0.3301579 1.421777e-02
#> 57274    0.925993344 -0.0715033 0.1276981 6.521212e-01
```

2.4.2 Fold ratios under the null hypotheses

Under the null hypothses, all sgRNAs levels are unchanged between the two conditions. To obtain fold ratios under the null, samples were permuted between two conditions, and log ratios were obtained from limma analysis under each permutation.

```
betanull <- permuteLimma(log2(mda231.norm + 1), design, contrast.matrix, 10)
theta0 <- sd(betanull)
theta0
#> [1] 0.4588307
```

2.4.3 Fitting three-component mixture models

A three-component mixture model (unchanged, overexpresssed, and underexpressed) is assumed for log ratios at different level of gene expression. Empirical Bayes method was employed to estimate paremeters of the mixtures and posterior means were obtained for estimating actual log ratios between the two conditions. P-values of sgRNAs were then calculated by permutation method.

```
nmm.fit <- normalMM(mda231.limma, theta0, n.b=3, d=5)
```

Results from the mixture model were shown in Figure 1. False discovery rate of 0.05 was used for declaring significant changes in red color between the two conditions for sgRNAs. The vertical lines are dividing sgRNAs into bins accroding to the gene expression levels of their targetted genes.

```
scatterPlot(nmm.fit$data,fdr=0.05,xlim=c(-0.5,12),ylim=c(-8,5))
```

2.4.4 Gene level summarization

From the p-values of sgRNAs, gene level p-values were obtained by using modified robust rank aggregation method (alpha-RRA). Log ratios were also summarized at gene level.

```
mda231.nmm <- nmm.fit[[1]]
p.gene <- calculateGenePval(exp(mda231.nmm$log_p), mda231.nmm$Gene, 0.05, nperm=10)
gene_fdr <- stats::p.adjust(p.gene$pvalue, method = "fdr")
gene_lfc <- calculateGeneLFC(mda231.nmm$lfc, mda231.nmm$Gene)

gene_summary <- data.frame(gene_pval=unlist(p.gene$pvalue), gene_fdr=as.matrix(gene_fdr), gene_lfc = as
gene_summary$gene <- rownames(gene_summary)
gene_summary <- gene_summary[,c(4,1:3)]
```

2.4.5 Gene level summarization

From the p-values of sgRNAs, gene level p-values were obtained by using modified robust rank aggregation method (alpha-RRA). Log ratios were also summarized at gene level.

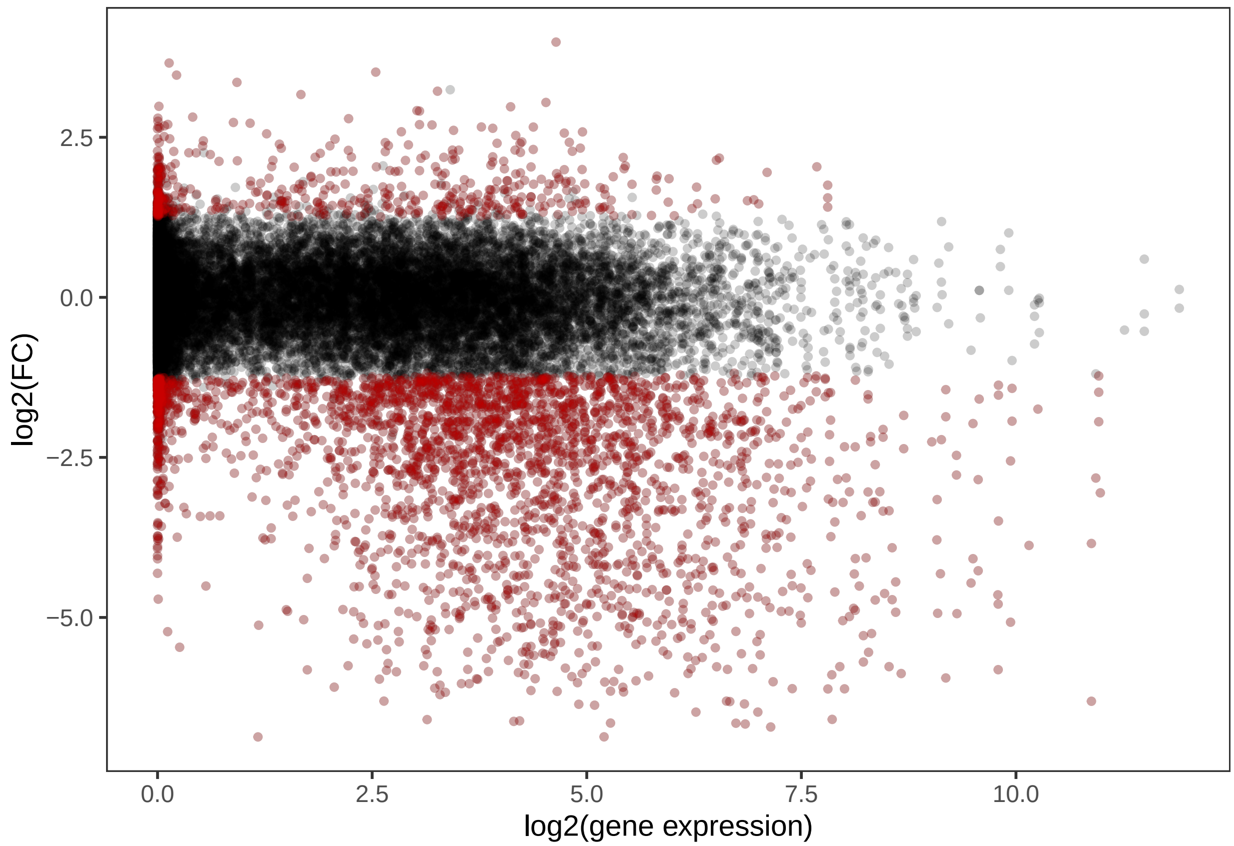

Figure 1: Log fold ratios of sgRNAs vs. gene expression level

```r
#extract gene expression data
gene.express <- mda231.nmm %>% group_by(Gene) %>% summarise_at(vars(exp.level.log2), max)
#merge gene summary with gene expression
gdata <- left_join(gene_summary, gene.express, by = c("gene" = "Gene"))
gdata <- gdata %>% filter(is.na(exp.level.log2)==FALSE)
# density plot and ridge plot
gdata$gene.fdr <- gdata$gene_fdr
data <- preparePlotData(gdata, gdata$gene.fdr)
```

Results from CEDA were shown in Figure 2. The points in the scatter plot were stratified into five color groups based on FDR. The 2D contour lines showed how the points distributed in 3D space.

```r
densityPlot(data)
```

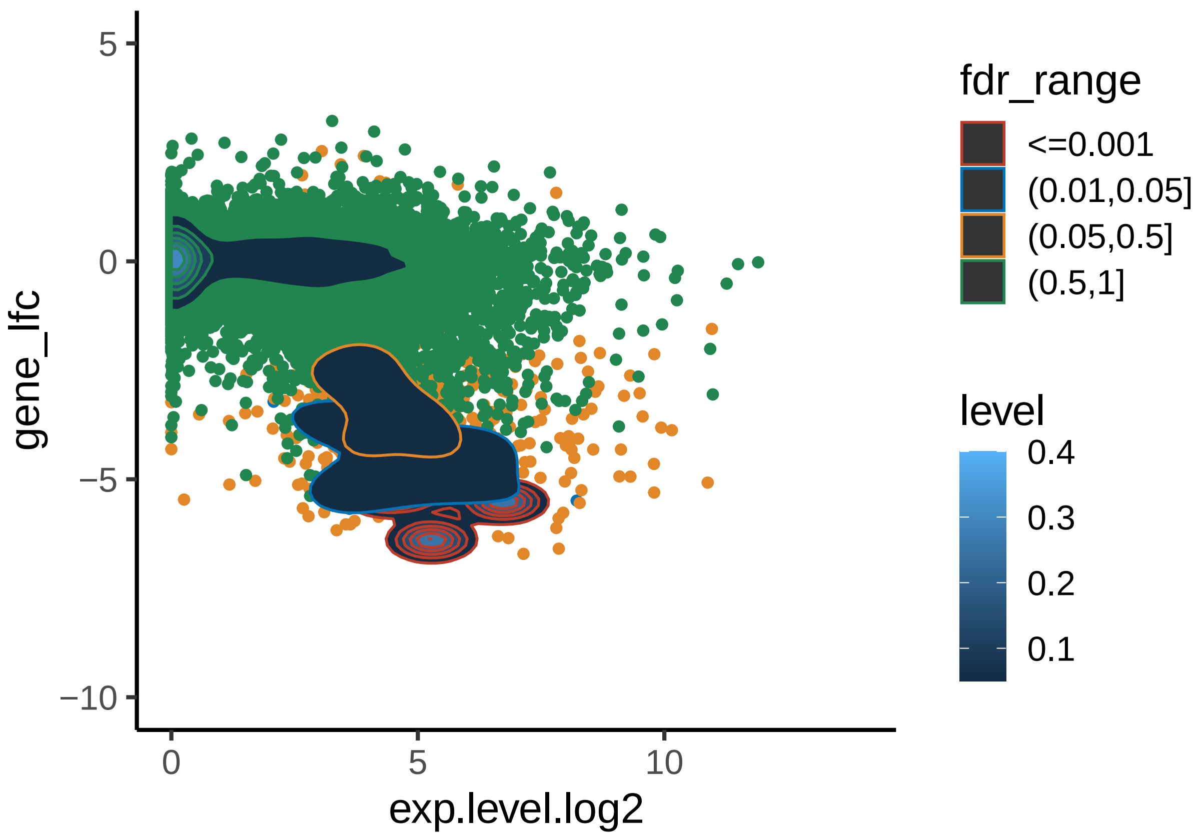

Figure 2: 2D density plot of gene log fold ratios vs. gene expression level for different FDR groups

The density ridgeline plot of CEDA results was shown in Figure 3. The groups of genes selected by CEDA (FDR<0.05, yellow, blue, and red) showed higher expression median values compared to the rest of genes (purple, green).

```r
ridgePlot(data)
```

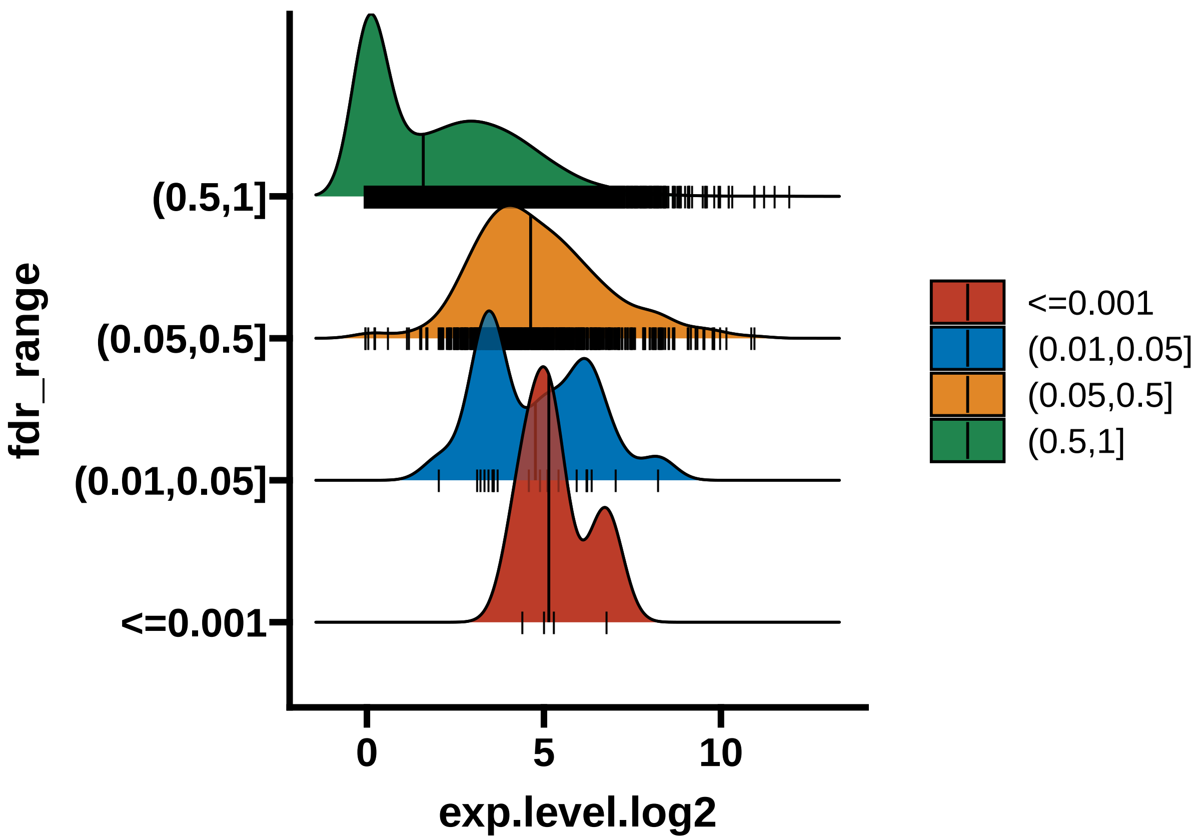

Figure 3: Ridge plot of gene expression for different FDR groups