

CRISPhieRmix Manual

Timothy Daley

8/2/2018

Contents

1	Introduction to CRISPhieRmix	2
2	Input parameters for CRISPhieRmix	3
2.1	Input	3
2.2	Return	3
3	Examples	4
3.1	A CRISPRi dropout screen	4

Contents

1 Introduction to CRISPhieRmix

CRISPhieRmix is an R package for analysing large CRISPR interference and activation (CRISPRi/a) screens. CRISPhieRmix uses a hierarchical mixture model approach to identify genes that are unlikely to follow the null distribution. CRISPhieRmix assumes that genes follow a mixture of null genes (genes with no effect) and non-null genes (genes with a significant effect). All guides from null genes follow a common null distribution. The guides for the non-null genes, on the other hand, are assumed to follow a mixture distribution, where some guides are ineffective (possibly with little or no change in the target gene expression) and follow the null distribution and some guides have an effect and follow an alternative distribution.

CRISPhieRmix can be used with or without negative control guides. We find that negative control guides help to better model the null distribution, as in our experience the null distribution tends to have long tails, and this helps to control the false discovery rate. A critical assumption is that the negative control guides accurately reflect the null distribution. This assumption can be violated in some screens, and we will discuss this later in depth.

2 Input parameters for CRISPhieRmix

2.1 Input

- **x** log2 fold changes of guides targeting genes (required)
- **geneIds** gene ids corresponding to **x** (required)
- **negCtrl** log2 fold changes of negative control guides; if **negCtrl** is not included then CRISPhieRmix uses a normal hierarchical mixture model
- **max_iter** maximum number of iterations for EM algorithm, default = 100
- **tol** tolerance for convergence of EM algorithm, default = 1e-10
- **pq** initial value of p-q, default = 0.1
- **mu** initial value of mu for the interesting genes, default = -4
- **sigma** initial value of sigma for the interesting genes, default = 1
- **nMesh** the number of points to use in numerical integration of posterior probabilities, default = 100
- **BIMODAL** boolean variable for BIMODAL mode to fit both positive and negative sides, used for cases such as Jost et al. 2017 where both sides are of interest.
- **VERBOSE** boolean variable for VERBOSE mode, default = FALSE
- **PLOT** boolean variable to produce plots, default = FALSE

2.2 Return

- **mixFit** a list containing the mixture fit for **x**
- **genes** a vector of genes from **geneIds**, in the same order as the following return values
- **locfdr** a vector of local false discovery rates, the posterior probability a gene is null in the same order as **genes**
- **FDR** a vector of global false discovery rates in the same order as **genes**
- **genePosteriors** a vector of posterior probabilities that the genes are non-null (equal to 1 - **locfdr**), in the same order as **genes**; when **BIMODAL** is set to TRUE then both **negGenePosteriors** and **posGenePosteriors** are returned that give the posterior probability that a gene is negative and positive, respectively

3 Examples

3.1 A CRISPRi dropout screen

Gilbert et al. 2014 performed genome wide screens for ricin resistance and susceptibility. In table 2, sheet 1 the authors provide the results of the CRISPRi screen at the sgRNA level. This includes the growth phenotype gamma and the ricin phenotype rho. They were interested in the ricin phenotype, but we will look at the growth phenotype to identify genes that lead to decreased growth and can thus be considered essential. In the CRISPhieRmix paper we used log2 fold changes computed by DESeq2 for a fairer comparison of algorithms (since other algorithms such as MAGeCK work directly on the counts), but here we will use the scores computed by Gilbert et al to show the flexibility of the CRISPhieRmix approach.

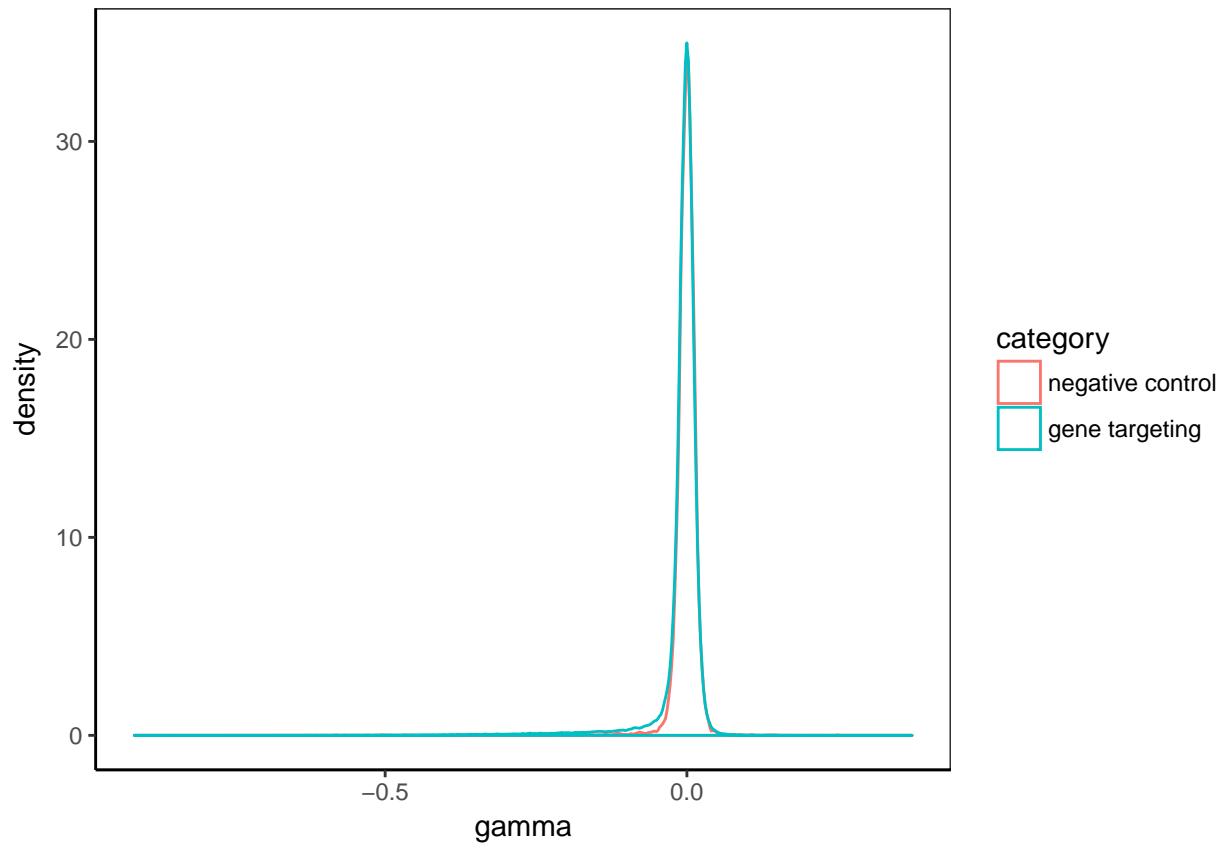
```
Gilbert2014Table2CRISPRi = read.table(file = "Gilbert2014Table2CRISPRi.txt", sep = "\t", header = TRUE)
head(Gilbert2014Table2CRISPRi)
```

```
##   gene sgRNA.ID Transcripts.targeted Protospacer.sequence
## 1 A1BG A1BG-1 all GCAAGAGAAAGACCACGAGCA
## 2 A1BG A1BG-10 all GCGGGAACAGGAGCCTTACGG
## 3 A1BG A1BG-2 all GTCTGCAGCAATGAGGCCCA
## 4 A1BG A1BG-3 all GCAGCCATATGTGAGTGCAG
## 5 A1BG A1BG-4 all GACATGATGGTCGCGCTCACTC
## 6 A1BG A1BG-5 all GAATGGTGGGCCAGGCCGGG
##   Growth.phenotype..gamma. CTx.DTA.phenotype..rho.
## 1 -0.008127700 0.011324965
## 2 -0.006738800 0.011671726
## 3 -0.009591072 -0.028942999
## 4 -0.017039814 0.046149024
## 5 -0.004868127 0.004907662
## 6 -0.012730560 -0.046096108
```

```
# identify the negative control guides
head(sort(table(Gilbert2014Table2CRISPRi$gene), decreasing = TRUE))
```

```
##
## negative_control HLA NKX2 C1QTNF9B
## 11219 193 85 45
## STON1 SSP0
## 45 44
```

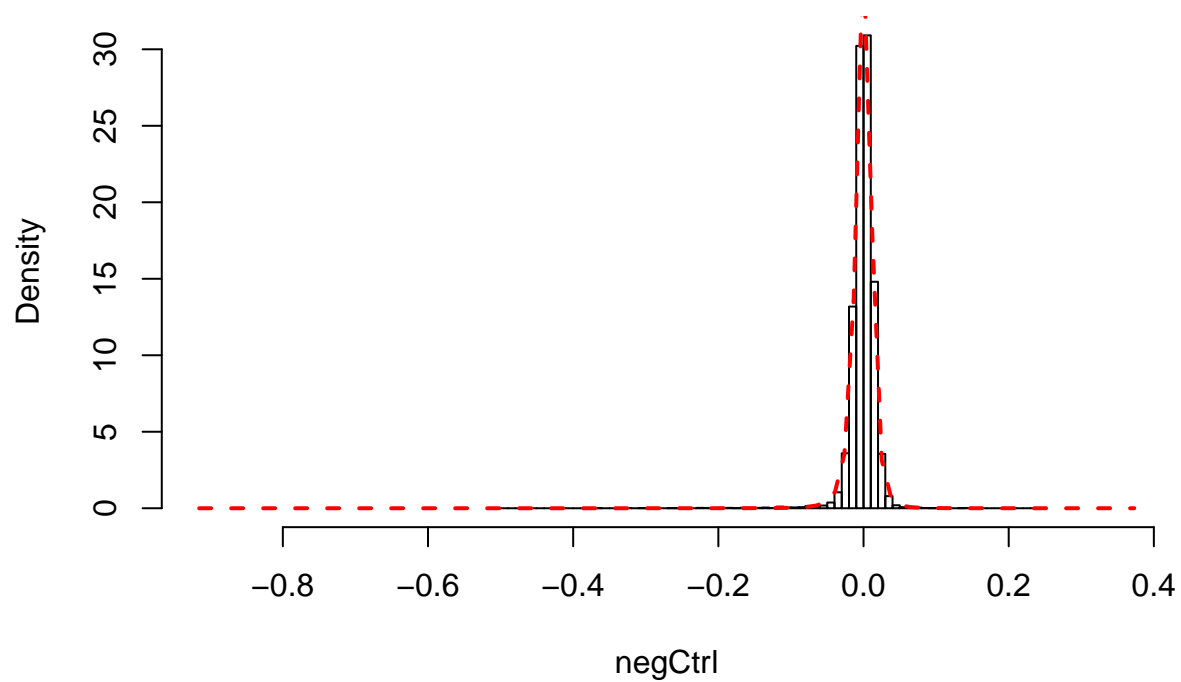
```
gamma = Gilbert2014Table2CRISPRi$Growth.phenotype..gamma.[which(Gilbert2014Table2CRISPRi$gene != "negative_control")]
negCtrl = Gilbert2014Table2CRISPRi$Growth.phenotype..gamma.[which(Gilbert2014Table2CRISPRi$gene == "negative_control")]
geneIds = Gilbert2014Table2CRISPRi$gene[which(Gilbert2014Table2CRISPRi$gene != "negative_control")]
# need to remove the negative_control factor
geneIds = factor(geneIds, levels = unique(geneIds))
x = data.frame(gamma = c(gamma, negCtrl), category = c(rep("gene targeting", times = length(gamma)), rep("negative control", times = length(negCtrl))))
x$category = factor(x$category, levels = c("negative control", "gene targeting"))
library(ggplot2)
ggplot(x, aes(x = gamma, colour = category)) + geom_density() + theme_bw() + theme(panel.grid.major = FALSE)
```



We see from the figure above that most of the gene targeting guides follow the same distribution as the negative control guides, but with a longer tail on the negative end. This is the signal due to cells dying as a result of the gene effects. We can now apply CRISPhieRmix to this data to identify genes that led to decreased growth or cell death.

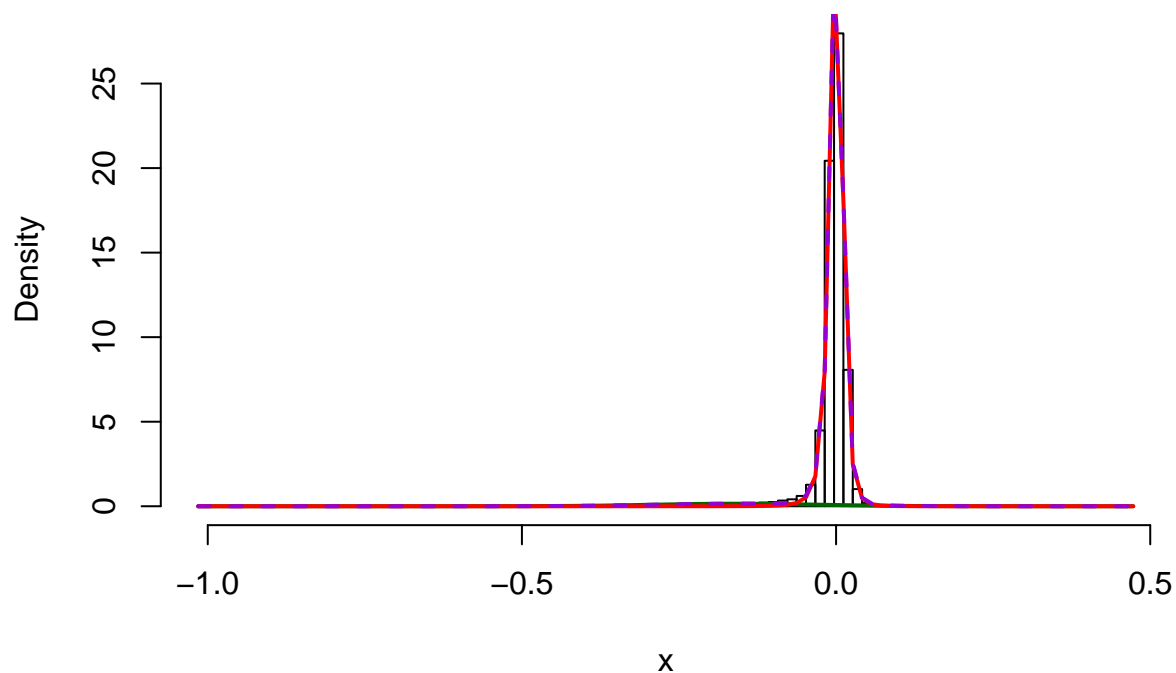
```
library(CRISPhieRmix)
gamma.CRISPhieRmix = CRISPhieRmix(x = gamma, geneIds = geneIds, negCtrl = negCtrl, mu = -0.2, sigma = 0
```

negative control fit



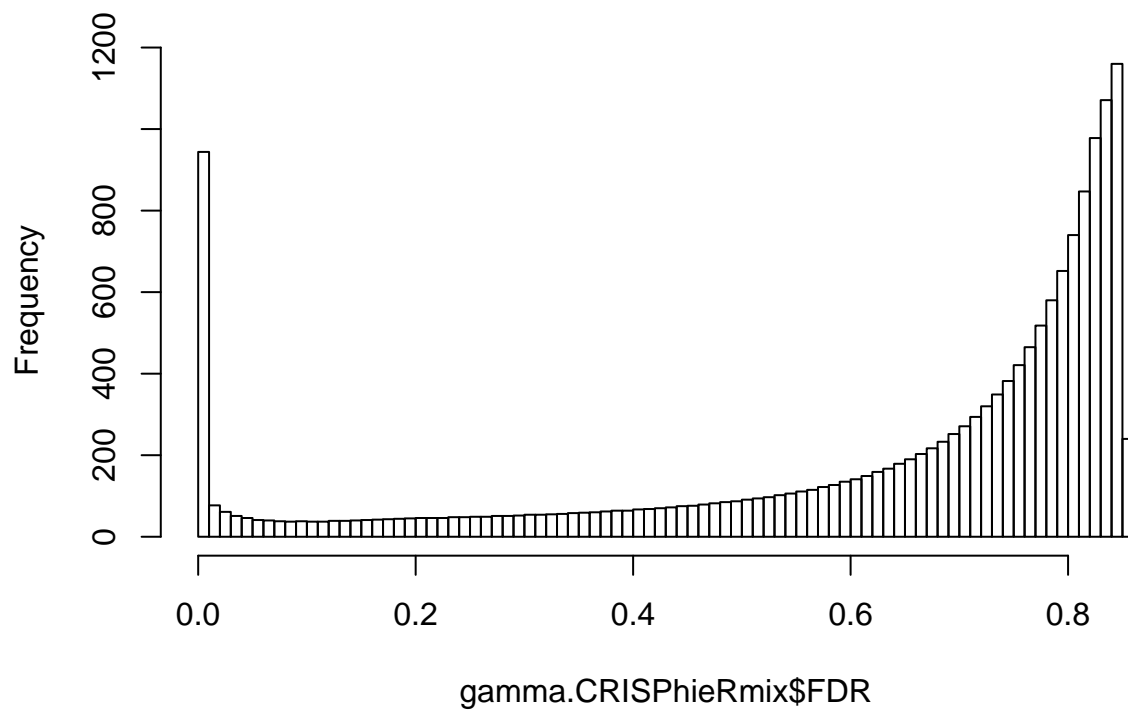
```
## fit negative control distributions
## 2 groups
## EM converged
## mu = -0.1575749
## sigma = 0.1315886
## pq = 0.04938287
```

mixture fit to observations



```
hist(gamma.CRISPhieRmix$FDR, breaks = 100)
```

Histogram of gamma.CRISPhieRmix\$FDR



```
sum(gamma.CRISPhieRmix$FDR < 0.1)
```

```
## [1] 1373
```

Let's look at how much these genes overlap the gold standard list of core constitutive genes and non-essential genes of Hart et al. 2014.

```
ConstitutiveCoreEssentialGenes = scan("ConstitutiveCoreEssentialGenes.txt", what = character())
length(ConstitutiveCoreEssentialGenes)
```

```
## [1] 217
```

```
length(intersect(ConstitutiveCoreEssentialGenes, gamma.CRISPhierMix$genes[which(gamma.CRISPhierMix$FDR < 0.1)]))
```

```
## [1] 170
```

```
NonEssentialGenes = scan("NonEssentialGenes.txt", what = character())
length(NonEssentialGenes)
```

```
## [1] 927
```

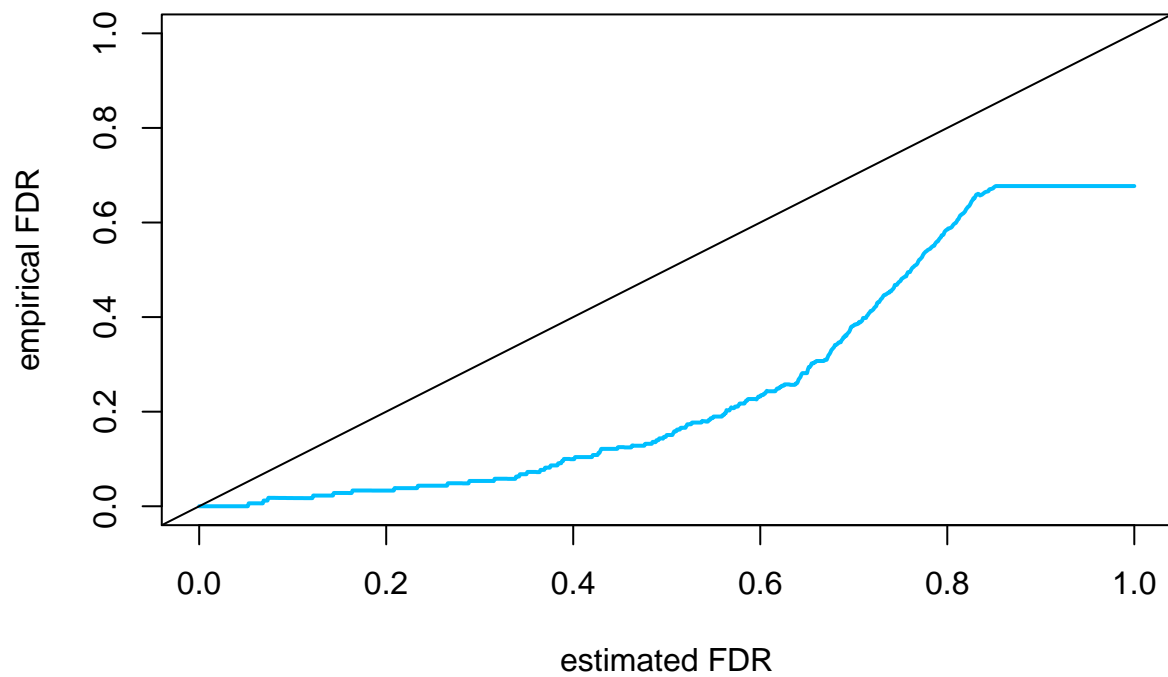
```
length(intersect(NonEssentialGenes, gamma.CRISPhierMix$genes[which(gamma.CRISPhierMix$FDR < 0.1)]))
```

```
## [1] 3
```

From this we can estimate the empirical false discovery rate at FDR level 0.1 as $3/1373 \approx 0.002$ and an empirical true positive rate as $170/217 \approx 0.78$. The empirical vs estimated FDR curve is plotted below.

```
EssentialGenes = data.frame(gene = factor(c(sapply(ConstitutiveCoreEssentialGenes, toString), sapply(NonEssentialGenes, toString))),
EssentialGenes = EssentialGenes[which(EssentialGenes$gene %in% gamma.CRISPhierMix$genes), ]
gamma.CRISPhierMixEssential = data.frame(genes = gamma.CRISPhierMix$genes, FDR = gamma.CRISPhierMix$FDR)
gamma.CRISPhierMixEssential = gamma.CRISPhierMixEssential[which(gamma.CRISPhierMixEssential$genes %in% EssentialGenes), ]
gamma.CRISPhierMixEssential = gamma.CRISPhierMixEssential[match(EssentialGenes$gene, gamma.CRISPhierMixEssential$genes), ]
fdr.curve <- function(thresh, fdrs, baseline){
  w = which(fdrs < thresh)
  if(length(w) > 0){
    return(sum(1 - baseline[w])/length(w))
  }
  else{
    return(NA)
  }
}
s = seq(from = 0, to = 1, length = 1001)
gamma.CRISPhierMixFdrCurve = sapply(s, function(t) fdr.curve(t, gamma.CRISPhierMixEssential$FDR, EssentialGenes))
plot(c(0, s[!is.na(gamma.CRISPhierMixFdrCurve)]), c(0, gamma.CRISPhierMixFdrCurve[!is.na(gamma.CRISPhierMixFdrCurve)]),
abline(0, 1))
```


Estimated vs Empirical Fdr



The receiver operator curve based on the above annotated genes is given below.

```
gamma.CRISPhieRmixROC = pROC::roc(EssentialGenes$essential, gamma.CRISPhieRmixEssential$FDR, auc = TRUE,
gamma.CRISPhieRmixROC
```

```
##
```

```
## Call:
```

```
## roc.default(response = EssentialGenes$essential, predictor = gamma.CRISPhieRmixEssential$FDR, auc = TRUE,
##
```

```
## Data: gamma.CRISPhieRmixEssential$FDR in 455 controls (EssentialGenes$essential 0) > 217 cases (EssentialGenes$essential 1)
```

```
## Area under the curve: 0.9259
```

```
plot(gamma.CRISPhieRmixROC, col = "deepskyblue", lwd = 2)
```

