Genome-wide Kozak Sequence Free Energy Analysis

Mariana Rius,Joshua Rest

Abstract

The purpose of this protocol is to evaluate the significance of delta G values, as a strategy to determine the significance of secondary structure. Sequences are shuffled randomly 100x, maintaining base composition but not base order. The percentile of the delta G vlaue of the non-scrambled (original) sequence is obtained from the distribution created by the delta G values of the 100X scrambled sequences.

Citation: Mariana Rius,Joshua Rest Genome-wide Kozak Sequence Free Energy Analysis. **protocols.io** dx.doi.org/10.17504/protocols.io.h9mb946
Published: 29 Sep 2017

Protocol

Load fasta file created previously and use mfold to obtain delta G values

Step 1.

Create object loading Sh.wg.promC.ATG26.fasta

Previously created in "Genome-wide Kozak Sequence Over-represented Motif Analysis" found here:

https://www.protocols.io/view/genome-wide-kozak-sequence-over-represented-motif-hikb4cw

cmd COMMAND (R - 3.3.2)

```
sh.seq <- read.fasta("Sh.wg.promC.ATG26.fasta")

system(paste("mfold_quik MAX=1 SEQ='Sh.wg.promC.ATG26.fasta'"))
```

Loading previously created fasta file and obtaining the delta G values for each sequence.

NOTES

Mariana Rius 01 Jun 2017

Using R version 3.3.2 and the following packages:
doBy (doBy_4.5-15)
data.table (data.table_1.10.0)
seqinr (seqinr_3.3-3)

Using mfold version 3.6

Create file of 100 additional scrambled sequences for each gene and calculate the delta G value of each

Step 2.

Create 100 sequences for each fragment randomly scrambling the base pairs to form new scrambled fragments. Use mfold (version 3.6) to calculate the delta g value of each fragment (original sequence and scrambled sequences).

cmd COMMAND (R - 3.3.2)

```
sh.seq2 <- lapply(sh.seq,function(dx) {
  sh.seq3 <-lapply(1:100,function(ddx) {
    paste(sample(dx),collapse="")})
  write.fasta(sequences=sh.seq3,file=attr(dx,"name"),as.string=TRUE,names=1:10)
  system(paste("mfold_quik MAX=1 SEQ='/scramseq100/",attr(dx,"name"),"'",sep=""))
}
)
```

Creates 100 additional 53bp fragments by scrambling the base pairs of the original fragment.

NOTES

Mariana Rius 01 Jun 2017

Using mfold version 3.6

Use grep to isolate the lowest delta g value for each sequence

Step 3.

In the shell, run grep command to create a file of only the lowest delta g values for each sequence.

cmd COMMAND

```
grep " dG" /scramseq100/*.det > sh100scrambledG.txt
```

Obtain the lowest delta G value for each fragment.

Load file in R and reorganize data table

Step 4.

Use fread to load the file. Use strsplit to organize data in a useful manner.

cmd COMMAND (R - 3.3.2)

```
sh100dGs <- fread("sh100scrambledG.txt")
sh100dGs[ , id := sapply(V1, function(v1also) strsplit(strsplit(v1also,"/")[[1]][7], "[.]")
[[1]][1])]

sh100dGs[,c("V1", "V2", "V3"):=NULL]
```

Create table of critical values representing the delta g of the original sequence fragments

Step 5.

Load file created previously by mfold ('Sh.wg.promC.ATG26.fasta.ct') and reorganize table.

cmd COMMAND (R - 3.3.2)

```
sh <- read.table("Sh.wg.promC.ATG26.fasta.ct", header= FALSE, fill = TRUE)
sh.deltag <-sh[sh$V2 == "dG",]
sh.deltag.t <- sh.deltag[!duplicated(sh.deltag$V5),]

sh.crit <- data.table( dG = sh.deltag.t$V4, id = as.character(sh.deltag.t$V5), key='id')
```

Load previously created file from mfold script in regards to Kozak sequences from Schizochytrium genome. Creates and organizes a table for use in comparison with the scrambled delta g values.

Obtain percentile for each original value when fit among distribution of delta G values from its scrambled fragments

Step 6.

Set the key for each data table and use ecdf() to create a distribution for each gene from the delta G values of the 100 scrambled fragments and obtain the percentile of the original delta G value when compared to the distribution.

cmd COMMAND (R - 3.3.2)

```
setkey(sh100dGs, id)
setkey(sh.crit, id)

sh.100dGs <- sh100dGs[sh.crit, nomatch=0]

bi <- data.table()
dG <- NULL
percentile <- c()
identity <- c()


for(i in 1:length(unique(sh.100dGs$id))){
  bi <- sh.100dGs[sh.100dGs$id == unique(sh.100dGs$id)[i],]
  values <- bi$V4
  id <- bi$id[1]
  dG <- (bi$dG)[1]
  info <- ecdf(values)
  percentile <- c(percentile, info(dG))
  identity <- c(identity, id)
}
sh.100percentiles <- data.table(percentile = percentile, id = identity)
```

Sets the key for each data table and uses ecdf() to obtain a percentile value of the original delta G value when compared to a distribution created by the delta G values of its 100 scrambled fragments.