MAR 21, 2024

OPEN ACCESS

DOI:
dx.doi.org/10.17504/protocols.io.dm6gpzeyjlzp/v1

Protocol Citation: Karina Jhingan 2024. Analyze Repeats. **protocols.io** https://dx.doi.org/10.17504/protocols.io.dm6gpzeyjlzp/v1

Protocol status: Working
We use this protocol and it's working

Created: Mar 20, 2024

Analyze Repeats

Karina Jhingan[1]

[1]Fred Hutch

Fred Hutch

Karina Jhingan
Fred Hutch

ABSTRACT

A protocol that uses Homer's analyzeRepeats, findGo, and FindMotifsGenome on the output of bowtie2 run on data using different pre-treatment conditions (LSD1i, TGFb, and LDS1i+TGFb) at 3 different timepoints (day 0, 8, 16).

Last Modified: Mar 21, 2024

PROTOCOL integer ID: 97013

Introduction

1 This linux analysis is using Homer, linked below. Here we are only focusing on the H3K9me3 histone for the human genome to compare different pre treatment conditions at different time points.
http://homer.ucsd.edu/homer/ngs/analyzeRNA.html

Load modules

2 I have found that the order that modules is loaded in can cause some errors later when using certain Homer functions and found that this order doesn't cause any errors. if your analysis is on an organism other than one of the preloaded organisms in Homer and you don't have write access, you likely will have to download Homer (the steps for doing this can be found in my CUT&tag protocol).

You can play around with loading or not loading samtools a second time as in line 4 of the below code, I found that this helped with certain Homer functions.

```
ml R
ml SAMtools
ml Homer
ml SAMtools
```

Set variables

3

```
hist="H3K9me3"
org="human"

#path to the input data
human="/fh/fast/greenberg_p/user/jwlee/ngs_processing/230225_VH00699_269_AAA
YVJYHV/alignment/bam"

#path to the genome reference file you will download in the next steo(this
can be the same as newPath)
projPath="/fh/fast/greenberg_p/user/kjhingan/CUT_TAG_Exp"

#path to store results
newPath="/fh/fast/greenberg_p/user/kjhingan/CUT_TAG_Exp/analyzeRepeats"

#different pre-treatment file names
treatments=("2-1" "3-1" "4-1")

#time points of the samples
days=("0" "8" "16")
```

4 Download genome reference file

```
wget
https://hgdownload.soe.ucsc.edu/goldenPath/hg38/bigZips/genes/hg38.refGene.g
tf.gz -O $projPath
```

Run AnalyzeRepeats

5 Here we first make a tag directory using Homer's function for each sample where the input is a mapped bam data file (from Bowtie2).
Because we are interested in entire gene bodies, the -count parameter would be "genes".

We will output the results into $projPath/analyzeRepeats/${org}_${sample}.txt

We will be running analyzeRepeats 3 times for each sample as we want to compare all time points (D0vD8, D8vD16, D0vD16

```
for treatment in ${treatments[@]}; do
for day in ${days[@]}; do
sample=D${day}_${treatment}
      #make a Tag Directory per sample
mkdir $projPath/TagDirectory/${org}_${sample}/
makeTagDirectory $projPath/TagDirectory/${org}_${sample}/
$human/${sample}_*_bowtie2_qualityScore2_mapped.bam
      #run analyzeRepeats
mkdir ${newPath}/${treatment}
analyzeRepeats.pl $projPath/data/hg38_genome.gtf hg38 -count genes -raw -d
${projPath}/TagDirectory/${org}_D0_${treatment}
${projPath}/TagDirectory/${org}_D8_${treatment}
>${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_temp.txt

analyzeRepeats.pl $projPath/data/hg38_genome.gtf hg38 -count genes -raw -d
${projPath}/TagDirectory/${org}_D8_${treatment}
${projPath}/TagDirectory/${org}_D16_${treatment}
>${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_temp.txt

analyzeRepeats.pl $projPath/data/hg38_genome.gtf hg38 -count genes -raw -d
${projPath}/TagDirectory/${org}_D0_${treatment}
${projPath}/TagDirectory/${org}_D16_${treatment}
>${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_temp.txt
done; done;
```

findGo (on top 1k differentially expressed genes)

6

Documentation: http://homer.ucsd.edu/homer/microarray/go.html

Here we are preparing the files to use as an input for findGo (gene ontology analysis) by finding the log fold change on our outputs from the previous step.

Here we calculate log fold change (lfc), sort the resulting file by lfc. Then we select relevant columns and take only the top 1000 rows.

We perform this 3 times per loop as we have the three different time comparisons as mentioned in the previous step, after that we run homer's findGo on these selected differentially expressed genes.

```
for treatment in ${treatments[@]}; do
mkdir ${newPath}/${treatment}/GO_D0.v.D8/
mkdir ${newPath}/${treatment}/GO_D8.v.D16/
mkdir ${newPath}/${treatment}/GO_D0.v.D16/

awk -F'\t' 'BEGIN {OFS=FS} {log_fold_change=log(($10 + 0.1) / ($9 + 0.1));
$11=log_fold_change; print}'
${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_temp.txt >
${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_templog.txt
sort -r -n -k 11
${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_templog.txt >
${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_templog_sorted.txt
cut -f1,2,3,4,5,6,7,8,9,10
${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_templog_sorted.txt
>
${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_templog_sorted_nor
mal.txt
head -n 1000
${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_templog_sorted_nor
mal.txt >
${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_top1k.txt

awk -F'\t' 'BEGIN {OFS=FS} {log_fold_change=log(($10 + 0.1) / ($9 + 0.1));
$11=log_fold_change; print}'
${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_temp.txt >
${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_templog.txt
sort -r -n -k 11
${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_templog.txt >
${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_templog_sorted.tx
t
cut -f1,2,3,4,5,6,7,8,9,10
${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_templog_sorted.tx
t >
${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_templog_sorted_no
rmal.txt
head -n 1000
${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_templog_sorted_no
rmal.txt >
${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_top1k.txt

awk -F'\t' 'BEGIN {OFS=FS} {log_fold_change=log(($10 + 0.1) / ($9 + 0.1));
$11=log_fold_change; print}'
${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_temp.txt >
${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_templog.txt
```

```
sort -r -n -k 11
${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_templog.txt >
${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_templog_sorted.tx
t
cut -f1,2,3,4,5,6,7,8,9,10
${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_templog_sorted.tx
t >
${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_templog_sorted_no
rmal.txt
head -n 1000
${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_templog_sorted_no
rmal.txt >
${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_top1k.txt

findGO.pl
${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_top1k.txt human
${newPath}/${treatment}/GO_D0.v.D8/
findGO.pl
${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_top1k.txt human
${newPath}/${treatment}/GO_D8.v.D16/
findGO.pl
${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_top1k.txt human
${newPath}/${treatment}/GO_D0.v.D16/
done;
```

Find Motifs

7 Documentation: http://homer.ucsd.edu/homer/ngs/peakMotifs.html

Also using the output of analyzeRepeats, we will select the columns in the correct orde in order to turn the text file into a bed file to input into findMotifsGenome from Homer.

Parameters: We just used the basic parameter for region size, although it is helpful to know the regions size specific to your experiment and use that instead. -p 4 refers to the number of cores to use, this function take a while to process, and I found that 4 was the optimal number of cores in our case, feel free to experiment going up and down in number of cores to speed up the time.

```
for treatment in ${treatments[@]}; do
mkdir ${newPath}/motifs/${treatment}

awk 'BEGIN { OFS="\t" } { print $2, $3, $4, $1, $6, $5; }'
${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_temp.txt >
${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_temp2.txt
tail -n +2
${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_temp2.txt >
${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_temp.bed
time findMotifsGenome.pl
${newPath}/${treatment}/D0_${treatment}.v.D8_${treatment}_temp.bed ${org}
${newPath}/motifs/${treatment}/D0_${treatment}.v.D8_${treatment} -size 200 -
p 4

awk 'BEGIN { OFS="\t" } { print $2, $3, $4, $1, $6, $5; }'
${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_temp.txt >
${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_temp2.txt
tail -n +2
${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_temp2.txt >
${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_temp.bed
time findMotifsGenome.pl
${newPath}/${treatment}/D8_${treatment}.v.D16_${treatment}_temp.bed ${org}
${newPath}/motifs/${treatment}/D8_${treatment}.v.D16_${treatment} -size
given -p 4

awk 'BEGIN { OFS="\t" } { print $2, $3, $4, $1, $6, $5; }'
${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_temp.txt >
${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_temp2.txt
tail -n +2
${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_temp2.txt >
${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_temp.bed
time findMotifsGenome.pl
${newPath}/${treatment}/D0_${treatment}.v.D16_${treatment}_temp.bed ${org}
${newPath}/motifs/${treatment}/D0_${treatment}.v.D16_${treatment} -size
given -p 4
done;
```

8