20151208

Scripts summary for Pengda #1

Questions to: **hyjin@scripps.edu**, Hyun Yong Jin, Xiao Lab, Scripps Research

using miR-17~92 TG, WT, TKO ribosome profiling analysis (R138)

**Ribosome footprint quantification**

**Overview**

\*Note: this methods are largely similar to RNA-seq analysis but some are different. The major differences are

1.

RNA-seq pipline: Cufflink --> Cuffmerge --> Cuffdiff

Ribosome profling pipeline: Directly to Cuffdiff

(Cuffmerge is to build de novo transcript discovery and build, but you cannot build de novo transcripts from results in ribosome profiling)

2.

After genome alignment (tophat) the perfect matched hit will be selected from the results.

Genome alignment usually allows one or a few nucleotide mismatches depending on scores of sequence quality of individual nucleotides and length of reads, but ribosome footprints are short, so we only allow perfectly matched reads for downstream analysis to increase sensitivity.

**Step01: Clip and Trim**

from fastq raw files,

clip the adaptor sequences and trim a nucleotide at the end (final nucleotide frequently introduced with a mutation)

Shell command, use FastX toolkit

$gzcat ~/TheShell/SeqResults/R138\_Jin\_RiPr/TG1\_\*.fastq.gz| \

fastx\_clipper -Q33 -a CTGTAGGCACCATCAAT -l 5 -c -n -v| \

fastx\_trimmer -Q33 -f 2 -l 80 >TG1\_cltr\_5-80.fq

\*for 9 samples, repeat 9 samples individually.

\*To conduct the multiple samples in a single script, use "&&" operator to consecutive commend.

For example to do two samples together,

$ gzcat ~/TheShell/SeqResults/R138\_Jin\_RiPr/WT2\_\*.fastq.gz| fastx\_clipper -Q33 -a CTGTAGGCACCATCAAT -l 5 -c -n -v| fastx\_trimmer -Q33 -f 2 -l 80 >WT2\_cltr\_5-80.fq && gzcat ~/TheShell/SeqResults/R138\_Jin\_RiPr/WT3\_\*.fastq.gz| fastx\_clipper -Q33 -a CTGTAGGCACCATCAAT -l 5 -c -n -v| fastx\_trimmer -Q33 -f 2 -l 80 >WT3\_cltr\_5-80.fq

This wil generate results as follow.

If your library is correctly generated, majority of your imput must contains adaptor sequences

Clipping Adapter: CTGTAGGCACCATCAAT

Min. Length: 5

Non-Clipped reads - discarded.

Input: 52027012 reads.

Output: **48119180 reads.**

discarded 33462 too-short reads.

discarded 128793 adapter-only reads.

discarded 3745577 non-clipped reads.

**Step02-1: Download reference genome.**

There are several reference genome you can use, but I will recommend iGenome from illumine, which is annotated in a way that cufflink smoothly works.

If you have generate your local environment using homebrew, use you local account (/usr/loca/) as the destined folder.

Cufflink provides direct link to illumine iGenome

<http://cole-trapnell-lab.github.io/cufflinks//igenome_table/index.html>

or you can directly go like this from Shell

wget --ftp-user=igenome --ftp-password=G3nom3s4u ftp://ussdftp.illumina.com/Homo\_sapiens/UCSC/hg19/Homo\_sapiens\_UCSC\_hg19.tar.gz

I used mm10 as reference.

**Step02-2: Build rRNA reference ebwt file from the iGenome**

$ mkdir -p /usr/local/iGenomes/contam/mm10rRNA

$ cd ~/TheShell/iGenomes/contam/mm10rRNA

$ bowtie-build /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Sequence/AbundantSequences/musRibosomal.fa mm10rRNA

# output directory is /contam/mm10rRNA and new index name is mm10rRNA

**Step02-3: Remove rRNA sequences using Bowtie**

$ bowtie -l 23 -t -p 4 --un=TG1\_norrna.fq /usr/local/iGenomes/contam/mm10rRNA/mm10rRNA ~/TheShell/SeqResults/R138\_jin\_RiPr/TG1\_cltr\_5-80.fq 2>> TG1\_stats.txt > TG1\_rrnaAlignments.aln && \

> bowtie -l 23 -t -p 4 --un=TG2\_norrna.fq /usr/local/iGenomes/contam/mm10rRNA/mm10rRNA ~/TheShell/SeqResults/R138\_jin\_RiPr/TG2\_cltr\_5-80.fq 2>> TG2\_stats.txt > TG2\_rrnaAlignments.aln && \

> bowtie -l 23 -t -p 4 --un=TG3\_norrna.fq /usr/local/iGenomes/contam/mm10rRNA/mm10rRNA ~/TheShell/SeqResults/R138\_jin\_RiPr/TG3\_cltr\_5-80.fq 2>> TG3\_stats.txt > TG3\_rrnaAlignments.aln && \

> bowtie -l 23 -t -p 4 --un=tkO1\_norrna.fq /usr/local/iGenomes/contam/mm10rRNA/mm10rRNA ~/TheShell/SeqResults/R138\_jin\_RiPr/tKO1\_cltr\_5-80.fq 2>> tKO1\_stats.txt > tKO1\_rrnaAlignments.aln && \

> bowtie -l 23 -t -p 4 --un=tkO2\_norrna.fq /usr/local/iGenomes/contam/mm10rRNA/mm10rRNA ~/TheShell/SeqResults/R138\_jin\_RiPr/tKO2\_cltr\_5-80.fq 2>> tKO2\_stats.txt > tKO2\_rrnaAlignments.aln && \

> bowtie -l 23 -t -p 4 --un=tkO3\_norrna.fq /usr/local/iGenomes/contam/mm10rRNA/mm10rRNA ~/TheShell/SeqResults/R138\_jin\_RiPr/tKO3\_cltr\_5-80.fq 2>> tKO3\_stats.txt > tKO3\_rrnaAlignments.aln && \

> bowtie -l 23 -t -p 4 --un=WT1\_norrna.fq /usr/local/iGenomes/contam/mm10rRNA/mm10rRNA ~/TheShell/SeqResults/R138\_jin\_RiPr/WT1\_cltr\_5-80.fq 2>> WT1\_stats.txt > WT1\_rrnaAlignments.aln && \

> bowtie -l 23 -t -p 4 --un=WT2\_norrna.fq /usr/local/iGenomes/contam/mm10rRNA/mm10rRNA ~/TheShell/SeqResults/R138\_jin\_RiPr/WT2\_cltr\_5-80.fq 2>> WT2\_stats.txt > WT2\_rrnaAlignments.aln && \

> bowtie -l 23 -t -p 4 --un=WT3\_norrna.fq /usr/local/iGenomes/contam/mm10rRNA/mm10rRNA ~/TheShell/SeqResults/R138\_jin\_RiPr/WT3\_cltr\_5-80.fq 2>> WT3\_stats.txt > WT3\_rrnaAlignments.aln

# bowtie -23 -t -p 4 --un=<unaligned fastq file name (results that we are interested in)> <Path to ebwt rRNA folder>/<index name> <previously trimmed and clipped  fastq file> 2>> stats.txt > <name of rRNA sequence aligned file>

# -l 23 indicates seed length is 23hr, -t is for knowing running time, -p 4 to enable multithreading (recommended by Gareth)

\*Note: You can remove rRNA at cuffdiff steps using mask.gtf file, and this is actually standard for RNA-seq analysis.

<http://onetipperday.blogspot.com/2012/08/how-to-get-trnarrnamitochondrial-gene.html>

But our current method was from Ingolia's 2012 protocol paper, and I assume the reasons are (1) you can check quality of reads using FastQC and (2) bowtie is specifically good for "short read" alignments

**Step02-4: Quality check with FastQC**

<http://www.bioinformatics.babraham.ac.uk/projects/fastqc/>

Download this and import your resulting fastq file.

It's very strait forward to use, and especially length distribution visualization is default output.

**Step 03-1 Tophat alignment**

This is the rate limiting steps, so could take more than 24h in my computer, and some RNA-seq takes even 3 days.

But your's must be faster.

no-novel-junc flag will tell tophat not to deal with novel splice discovery, which will significantly reduce running time.

Depending on the numbers of your cpu cores, you can increase thread numbers allowing multi-treaded calculation. My mac is dual core, so maximum threads I can use is 2, but I use 1 as I need to use my computer for other purpose at the same time.

You can run with n-1 threads on an n-core machine. For examples, 15 threads on a 16 core computer.

Scripts for individual sample.

$tophat --no-novel-juncs \

--GTF /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Annotation/Genes/genes.gtf \

--num-threads 1 \

--output-dir ~/TheShell/SeqResults/R138\_Jin\_RiPr/04\_topHat\_no\_rtrna/ \

 /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Sequence/BowtieIndex/genome ~/TheShell/SeqResults/R138\_Jin\_RiPr/03\_bowtie\_no\_trna/R138\_no\_trna.fq

#no-novel-juncs flag will save time if you are not interested in doing novel splice site discovery

#num-threads 1 --> if workflow is run on a machine with multiple cores, this number may be increased to reflect the number of cores present

#GTF file is for known junction used for analysis.

#this will sue Bowtie 1.0.0.0. as I don't have bowtie2 installed. maybe it is as intended.

If you have 9 sample analysis together,

$tophat --no-novel-juncs --GTF /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Annotation/Genes/genes.gtf --num-threads 1 --output-dir ~/TheShell/SeqResults/R138\_Jin\_RiPr/TG1\_topHat/ /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Sequence/BowtieIndex/genome ~/TheShell/SeqResults/R138\_Jin\_RiPr/TG1\_norrna.fq && \

tophat --no-novel-juncs --GTF /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Annotation/Genes/genes.gtf --num-threads 1 --output-dir ~/TheShell/SeqResults/R138\_Jin\_RiPr/TG2\_topHat/ /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Sequence/BowtieIndex/genome ~/TheShell/SeqResults/R138\_Jin\_RiPr/TG2\_norrna.fq && \

tophat --no-novel-juncs --GTF /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Annotation/Genes/genes.gtf --num-threads 1 --output-dir ~/TheShell/SeqResults/R138\_Jin\_RiPr/TG3\_topHat/ /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Sequence/BowtieIndex/genome ~/TheShell/SeqResults/R138\_Jin\_RiPr/TG3\_norrna.fq && \

tophat --no-novel-juncs --GTF /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Annotation/Genes/genes.gtf --num-threads 1 --output-dir ~/TheShell/SeqResults/R138\_Jin\_RiPr/tKO1\_topHat/ /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Sequence/BowtieIndex/genome ~/TheShell/SeqResults/R138\_Jin\_RiPr/tKO1\_norrna.fq && \

tophat --no-novel-juncs --GTF /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Annotation/Genes/genes.gtf --num-threads 1 --output-dir ~/TheShell/SeqResults/R138\_Jin\_RiPr/tKO2\_topHat/ /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Sequence/BowtieIndex/genome ~/TheShell/SeqResults/R138\_Jin\_RiPr/tKO2\_norrna.fq && \

tophat --no-novel-juncs --GTF /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Annotation/Genes/genes.gtf --num-threads 1 --output-dir ~/TheShell/SeqResults/R138\_Jin\_RiPr/tKO3\_topHat/ /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Sequence/BowtieIndex/genome ~/TheShell/SeqResults/R138\_Jin\_RiPr/tKO3\_norrna.fq && \

tophat --no-novel-juncs --GTF /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Annotation/Genes/genes.gtf --num-threads 1 --output-dir ~/TheShell/SeqResults/R138\_Jin\_RiPr/WT1\_topHat/ /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Sequence/BowtieIndex/genome ~/TheShell/SeqResults/R138\_Jin\_RiPr/WT1\_norrna.fq && \

tophat --no-novel-juncs --GTF /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Annotation/Genes/genes.gtf --num-threads 1 --output-dir ~/TheShell/SeqResults/R138\_Jin\_RiPr/WT2\_topHat/ /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Sequence/BowtieIndex/genome ~/TheShell/SeqResults/R138\_Jin\_RiPr/WT2\_norrna.fq && \

tophat --no-novel-juncs --GTF /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Annotation/Genes/genes.gtf --num-threads 1 --output-dir ~/TheShell/SeqResults/R138\_Jin\_RiPr/WT3\_topHat/ /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Sequence/BowtieIndex/genome ~/TheShell/SeqResults/R138\_Jin\_RiPr/WT3\_norrna.fq

**Step 03-2 Select perfect matched hits using Samtool**

This is unique to ribosome profiling that are not usually done with RNA-seq analysis (from Ingolia's 2012 paper)

$ cd <output folder>

$ samtools view -h ~/TheShell/SeqResults/R138\_Jin\_RiPr/03\_topHat/TG1\_topHat/accepted\_hits.bam| grep -E '(NM:i:0)|(^@)'| samtools view -S -b ->perfect\_hits\_TG1.bam &&\

samtools view -h ~/TheShell/SeqResults/R138\_Jin\_RiPr/03\_topHat/TG2\_topHat/accepted\_hits.bam| grep -E '(NM:i:0)|(^@)'| samtools view -S -b ->perfect\_hits\_TG2.bam &&\

samtools view -h ~/TheShell/SeqResults/R138\_Jin\_RiPr/03\_topHat/TG3\_topHat/accepted\_hits.bam| grep -E '(NM:i:0)|(^@)'| samtools view -S -b ->perfect\_hits\_TG3.bam &&\

samtools view -h ~/TheShell/SeqResults/R138\_Jin\_RiPr/03\_topHat/tKO1\_topHat/accepted\_hits.bam| grep -E '(NM:i:0)|(^@)'| samtools view -S -b ->perfect\_hits\_tKO1.bam &&\

samtools view -h ~/TheShell/SeqResults/R138\_Jin\_RiPr/03\_topHat/tKO2\_topHat/accepted\_hits.bam| grep -E '(NM:i:0)|(^@)'| samtools view -S -b ->perfect\_hits\_tKO2.bam &&\

samtools view -h ~/TheShell/SeqResults/R138\_Jin\_RiPr/03\_topHat/tKO3\_topHat/accepted\_hits.bam| grep -E '(NM:i:0)|(^@)'| samtools view -S -b ->perfect\_hits\_tKO3.bam &&\

samtools view -h ~/TheShell/SeqResults/R138\_Jin\_RiPr/03\_topHat/WT1\_topHat/accepted\_hits.bam| grep -E '(NM:i:0)|(^@)'| samtools view -S -b ->perfect\_hits\_WT1.bam &&\

samtools view -h ~/TheShell/SeqResults/R138\_Jin\_RiPr/03\_topHat/WT2\_topHat/accepted\_hits.bam| grep -E '(NM:i:0)|(^@)'| samtools view -S -b ->perfect\_hits\_WT2.bam &&\

samtools view -h ~/TheShell/SeqResults/R138\_Jin\_RiPr/03\_topHat/WT3\_topHat/accepted\_hits.bam| grep -E '(NM:i:0)|(^@)'| samtools view -S -b ->perfect\_hits\_WT3.bam

**Step04 Cuffdiff for differential gene expression analysis**

$ cuffdiff -L TG,tKO,WT \

-o cuffdiffOutput /usr/local/iGenomes/Mus\_musculus/UCSC/mm10/Annotation/Genes/genes.gtf \

accepted\_hits\_TG1.bam,accepted\_hits\_TG2.bam,accepted\_hits\_TG3.bam \

accepted\_hits\_tKO1.bam,accepted\_hits\_tKO2.bam,accepted\_hits\_tKO3.bam \

accepted\_hits\_WT1.bam,accepted\_hits\_WT2.bam,accepted\_hits\_WT3.bam

This results are now in the final folde "cuffdiffOutput"

**Step05 CummeRbund analysis for gene experession quantification**

Results are already calculated in "cuffdiffOutput" folder, you can manually extract data from it.

But bioconductor package in R called CummeRbund will make this extraction steps easy.

Now move on to R,

library(cummeRbund)

#set up defalt directory as cuffdiff output folder containing genes.fpkm\_tracking etc.

# this will generate cuffData.db file

setwd("~/TheShell/SeqResults/R138\_Jin\_RiPr/05\_cuffdiff/perfect\_hits\_analysis/cuffdiffOutput/")

cuff <-readCufflinks() #Now all your results are under "cuff"

cuff

#output

CuffSet instance with:

   3 samples

23980 genes

33295 isoforms

27067 TSS

26408 CDS

71772 promoters

81201 splicing

61419 relCDS

# You can do many analysis in this program, so please visit vignette.

#for FPKM results in csv table, do it as follow

# mean value

gene.matrix <-fpkmMatrix(genes(cuff))

head(gene.matrix)

write.csv(gene.matrix, file="fpkm\_mean.csv") # export to the base dir

#Individual replicates

gene.rep.matrix<-repFpkmMatrix(genes(cuff))

head(gene.rep.matrix)

write.csv(gene.rep.matrix, file="fpkm\_replicate.csv") # export to the base dir

Basic Idea:

Align footprints to annotated BED files contains all canonical mRNAs, and calculate how many footprints are aligned to the each position.

This will generate a huge, single text file, and downstream analysis will be conducted in R.

**01\_Generate a BED file with only unique, canonical exon regions**

If you use whole genome sequence as a align reference, this will generate ~200GB of final table after your footprint alignment. So not feasible.

Following is how I generate BED file with only exon.

I will give you the file, so you don't need to repeat this.

-------------------------------------

**Generation of BED file of exon only from unique transcript from UCSC genome browser [20140826]**

**Point is to generate transcript (5UTR/CDS/3UTR) with single transcript that we can avoid overlapping count**

Download canonical transcript - gene symbol

track UCSC genes

table :KnownCanonical

output format: select fields from primary and related tables

press get output

select fields from mm10.KnownCannonical -->term, chrimStart, chromed

transcript select fields from mm10kgXref: known gene ID, geneSymbol, RefseqID

get output

$ head -n 20 mm10\_kG\_canonical

#mm10.knownCanonical.chrom mm10.knownCanonical.chromStart mm10.knownCanonical.chromEnd mm10.kgXref.kgID mm10.kgXref.geneSymbol mm10.kgXref.refseq

chr1 3214481 3671498 uc007aeu.1 Xkr4 NM\_001011874

chr1 3648310 3658904 uc007aev.1 AK149000

chr1 4343506 4360314 uc007aex.2 Rp1 NM\_011283

chr1 4490927 4497354 uc007afc.2 Sox17 NM\_001289464

chr1 4773199 4785726 uc007aff.3 Mrpl15 NM\_001177658

chr1 4807892 4846735 uc007afh.1 Lypla1 NM\_008866

chr1 4857693 4897909 uc007afi.2 Tcea1 NM\_011541

chr1 4909575 5070285 uc007afl.2 Rgs20 NM\_001177795

chr1 5083172 5162549 uc007afn.1 Atp6v1h NM\_133826

chr1 5588492 5606133 uc007afp.2 Oprk1 NM\_001204371

chr1 5913706 5917398 uc011whw.1 Npbwr1 NM\_010342

chr1 6214661 6276104 uc007afr.2 Rb1cc1 NM\_009826

chr1 6359330 6394731 uc007afv.2 Fam150a NM\_001195732

chr1 6730050 6860940 uc007aga.1 St18 NM\_173868

chr1 6692281 6692305 uc029qmq.1 AB335791

chr1 7088919 7173628 uc007agb.1 Pcmtd1 NM\_183028

chr1 7349405 7397869 uc007age.1 AK043789

chr1 8028518 8028555 uc029qmr.1 AB351889

chr1 8179496 8179520 uc029qms.1 AB351889

# as this is canonical list, gene should not be overlapped

$ grep "Cyld" mm10\_canonical

chr8 88697027 88751946 uc009mrt.3 Cyld NM\_173369

#Thee other non-canonical Cyld transcripts are not included

Download Exon coding, exon 3UTR and exon 5UTR separately from knownGenetable

track: UCSC Genes

table:knownGene

output format BED

get output

select exon exon 3UTR and exon 5UTR separately

$ head -n 20 mm10\_kG\_exon\_coding

track name="tb\_knownGene" description="table browser query on knownGene" visibility=2 url=

chr1 3216021 3216968 uc007aeu.1\_cds\_0\_0\_chr1\_3216022\_r 0 -

chr1 3421701 3421901 uc007aeu.1\_cds\_1\_0\_chr1\_3421702\_r 0 -

chr1 3670551 3671348 uc007aeu.1\_cds\_2\_0\_chr1\_3670552\_r 0 -

$ awk '{split ($4,a,"\_"); {print $1"\t"$2"\t"$3"\t"a[1]"\t"a[2]"\t"a[3]"\t"$6}}' mm10\_kG\_exon\_coding >mm10\_kG\_exon\_coding\_mod.bed

$ awk '{split ($4,a,"\_"); {print $1"\t"$2"\t"$3"\t"a[1]"\t"a[2]"\t"a[3]"\t"$6}}' mm10\_kG\_exon\_utr3 >mm10\_kG\_exon\_utr3\_mod.bed

$ awk '{split ($4,a,"\_"); {print $1"\t"$2"\t"$3"\t"a[1]"\t"a[2]"\t"a[3]"\t"$6}}' mm10\_kG\_exon\_utr5 >mm10\_kG\_exon\_utr5\_mod.bed

#Sanity test

$ head -n 20 mm10\_kG\_exon\_coding\_mod.bed

chr1 3216021 3216968 uc007aeu.1 cds 0 -

chr1 3421701 3421901 uc007aeu.1 cds 1 -

chr1 3670551 3671348 uc007aeu.1 cds 2 -

chr1 4292980 4293012 uc007aew.1 cds 0 -

chr1 4351909 4352081 uc007aew.1 cds 1 -

chr1 4352201 4352837 uc007aew.1 cds 2 -

chr1 4409169 4409187 uc007aew.1 cds 3 -

$ grep "uc008hfr.1" mm10\_kG\_exon\_coding\_mod.bed

chr19 32758444 32758523 uc008hfr.1 cds 0 +

chr19 32776014 32776099 uc008hfr.1 cds 1 +

chr19 32792548 32792593 uc008hfr.1 cds 2 +

chr19 32798070 32798114 uc008hfr.1 cds 3 +

chr19 32799860 32800099 uc008hfr.1 cds 4 +

chr19 32811695 32811837 uc008hfr.1 cds 5 +

chr19 32815416 32815583 uc008hfr.1 cds 6 +

chr19 32817835 32818060 uc008hfr.1 cds 7 +

chr19 32819842 32820028 uc008hfr.1 cds 8 +

# this gene is pten

Add 1nt to all start site

$ awk -v s=1 '{print $1"\t"$2+s"\t"$3"\t"$4"\t"$5"\t"$6"\t"$7}' mm10\_kG\_exon\_coding\_mod.bed >mm10\_kG\_exon\_coding\_mod2.bed

$ awk -v s=1 '{print $1"\t"$2+s"\t"$3"\t"$4"\t"$5"\t"$6"\t"$7}' mm10\_kG\_exon\_utr3\_mod.bed >mm10\_kG\_exon\_utr3\_mod2.bed

$ awk -v s=1 '{print $1"\t"$2+s"\t"$3"\t"$4"\t"$5"\t"$6"\t"$7}' mm10\_kG\_exon\_utr5\_mod.bed >mm10\_kG\_exon\_utr5\_mod2.bed

Combine 3UTR, CDS and 5UTR bed file into one

$ cat mm10\_kG\_exon\_utr5\_mod2.bed mm10\_kG\_exon\_coding\_mod2.bed  mm10\_kG\_exon\_utr3\_mod2.bed >mm10\_kG\_exon\_combined.bed

Sanity test

$ grep "uc008hfr.1" mm10\_kG\_exon\_combined.bed

chr19 32757577 32758444 uc008hfr.1 utr5 0 +

chr19 32758445 32758523 uc008hfr.1 cds 0 +

chr19 32776015 32776099 uc008hfr.1 cds 1 +

chr19 32792549 32792593 uc008hfr.1 cds 2 +

chr19 32798071 32798114 uc008hfr.1 cds 3 +

chr19 32799861 32800099 uc008hfr.1 cds 4 +

chr19 32811696 32811837 uc008hfr.1 cds 5 +

chr19 32815417 32815583 uc008hfr.1 cds 6 +

chr19 32817836 32818060 uc008hfr.1 cds 7 +

chr19 32819843 32820028 uc008hfr.1 cds 8 +

chr19 32820029 32826160 uc008hfr.1 utr3 8 +

#note that there are no one nucleotide overlap and known geneID as identifier so we have only one transcript corresponding to mRNA gene

**Add gene name and NM name on top of it using knownGene name (4th column) as identifier**

$ join -1 4 -2 4 <(sort -k4 mm10\_kG\_exon\_combined.bed) <(sort -k4  mm10\_kG\_canonical) > test.bed

$ join -1 4 -2 4 <(sort -k4 mm10\_kG\_exon\_combined.bed) <(sort -k4  mm10\_kG\_canonical) | awk '{print $2"\t"$3"\t"$4"\t"$1"\t"$5"\t"$6"\t"$7"\t"$11"\t"$12"\t"}' | bedtools sort -i "-" > mm10\_kG\_exon\_all.bed

$ head -30 mm10\_kG\_exon\_all.bed

chr1 3214482 3216021 uc007aeu.1 utr3 0 - Xkr4 NM\_001011874

chr1 3216022 3216968 uc007aeu.1 cds 0 - Xkr4 NM\_001011874

chr1 3421702 3421901 uc007aeu.1 cds 1 - Xkr4 NM\_001011874

chr1 3648311 3650509 uc007aev.1 utr5 0 - AK149000

chr1 3658847 3658904 uc007aev.1 utr5 1 - AK149000

chr1 3670552 3671348 uc007aeu.1 cds 2 - Xkr4 NM\_001011874

chr1 3671349 3671498 uc007aeu.1 utr5 2 - Xkr4 NM\_001011874

chr1 4343507 4344599 uc007aex.2 utr3 0 - Rp1 NM\_011283

chr1 4344600 4350091 uc007aex.2 cds 0 - Rp1 NM\_011283

chr1 4351910 4352081 uc007aex.2 cds 1 - Rp1 NM\_011283

chr1 4352202 4352825 uc007aex.2 cds 2 - Rp1 NM\_011283

chr1 4352826 4352837 uc007aex.2 utr5 2 - Rp1 NM\_011283

chr1 4360200 4360314 uc007aex.2 utr5 3 - Rp1 NM\_011283

chr1 4490928 4491715 uc007afc.2 utr3 0 - Sox17 NM\_001289464

chr1 4491716 4492668 uc007afc.2 cds 0 - Sox17 NM\_001289464

chr1 4493100 4493406 uc007afc.2 cds 1 - Sox17 NM\_001289464

chr1 4493407 4493490 uc007afc.2 utr5 1 - Sox17 NM\_001289464

chr1 4493772 4493863 uc007afc.2 utr5 2 - Sox17 NM\_001289464

chr1 4496291 4497354 uc007afc.2 utr5 3 - Sox17 NM\_001289464

$ grep "uc008hfr.1" mm10\_kG\_exon\_all.bed

chr19 32757577 32758444 uc008hfr.1 utr5 0 + Pten NM\_008960

chr19 32758445 32758523 uc008hfr.1 cds 0 + Pten NM\_008960

chr19 32776015 32776099 uc008hfr.1 cds 1 + Pten NM\_008960

chr19 32792549 32792593 uc008hfr.1 cds 2 + Pten NM\_008960

chr19 32798071 32798114 uc008hfr.1 cds 3 + Pten NM\_008960

chr19 32799861 32800099 uc008hfr.1 cds 4 + Pten NM\_008960

chr19 32811696 32811837 uc008hfr.1 cds 5 + Pten NM\_008960

chr19 32815417 32815583 uc008hfr.1 cds 6 + Pten NM\_008960

chr19 32817836 32818060 uc008hfr.1 cds 7 + Pten NM\_008960

chr19 32819843 32820028 uc008hfr.1 cds 8 + Pten NM\_008960

chr19 32820029 32826160 uc008hfr.1 utr3 8 + Pten NM\_008960

$ grep "Cyld" mm10\_kG\_exon\_all.bed

chr8 88697028 88697099 uc009mrt.3 utr5 0 + Cyld NM\_173369

chr8 88705226 88705375 uc009mrt.3 utr5 1 + Cyld NM\_173369

chr8 88705376 88705879 uc009mrt.3 cds 1 + Cyld NM\_173369

chr8 88707089 88707391 uc009mrt.3 cds 2 + Cyld NM\_173369

chr8 88709891 88709996 uc009mrt.3 cds 3 + Cyld NM\_173369

chr8 88719298 88719396 uc009mrt.3 cds 4 + Cyld NM\_173369

chr8 88723099 88723212 uc009mrt.3 cds 5 + Cyld NM\_173369

chr8 88729459 88729838 uc009mrt.3 cds 6 + Cyld NM\_173369

chr8 88730656 88730821 uc009mrt.3 cds 7 + Cyld NM\_173369

chr8 88731675 88731816 uc009mrt.3 cds 8 + Cyld NM\_173369

chr8 88732986 88733108 uc009mrt.3 cds 9 + Cyld NM\_173369

chr8 88734883 88734974 uc009mrt.3 cds 10 + Cyld NM\_173369

chr8 88735831 88735897 uc009mrt.3 cds 11 + Cyld NM\_173369

chr8 88741291 88741423 uc009mrt.3 cds 12 + Cyld NM\_173369

chr8 88742286 88742394 uc009mrt.3 cds 13 + Cyld NM\_173369

chr8 88744828 88744946 uc009mrt.3 cds 14 + Cyld NM\_173369

chr8 88745209 88745425 uc009mrt.3 cds 15 + Cyld NM\_173369

chr8 88746829 88747013 uc009mrt.3 cds 16 + Cyld NM\_173369

chr8 88747014 88751946 uc009mrt.3 utr3 16 + Cyld NM\_173369

#note that gene name Cyld is matched to one transcript!!!

Extract only mRNAs using NM\_ as identifier

$ grep NM mm10\_kG\_exon\_all.bed > mm10\_kG\_exon\_mrna.bed

14.5 mb / 15.8mb left

**---------------------------------------**

**02\_Genrating Coverage file using CoverageBed**

From bam file with perfect hit is isolated, do as follow.

For R138 (TKO, WT and TG ribosome profiling) sample, I already did this, so I will give you the file

My version is 2.20.1 (Pengda tried 2.25, seems do not work?)

$

coverageBed -split -abam ~/TheShell/SeqResults/R138\_Jin\_RiPr/03-1\_perfect\_hits/perfect\_hits\_TG1.bam -b  ~/TheShell/SeqResults/R138\_Jin\_RiPr/06\_visualization/coverageBed/mm10\_kG\_exon\_mrna.bed -d > TG1coverage.cov && \

coverageBed -split -abam ~/TheShell/SeqResults/R138\_Jin\_RiPr/03-1\_perfect\_hits/perfect\_hits\_TG2.bam -b  ~/TheShell/SeqResults/R138\_Jin\_RiPr/06\_visualization/coverageBed/mm10\_kG\_exon\_mrna.bed -d > TG2coverage.cov && \

coverageBed -split -abam ~/TheShell/SeqResults/R138\_Jin\_RiPr/03-1\_perfect\_hits/perfect\_hits\_TG3.bam -b  ~/TheShell/SeqResults/R138\_Jin\_RiPr/06\_visualization/coverageBed/mm10\_kG\_exon\_mrna.bed -d > TG3coverage.cov && \

coverageBed -split -abam ~/TheShell/SeqResults/R138\_Jin\_RiPr/03-1\_perfect\_hits/perfect\_hits\_tKO1.bam -b  ~/TheShell/SeqResults/R138\_Jin\_RiPr/06\_visualization/coverageBed/mm10\_kG\_exon\_mrna.bed -d > tKO1coverage.cov && \

coverageBed -split -abam ~/TheShell/SeqResults/R138\_Jin\_RiPr/03-1\_perfect\_hits/perfect\_hits\_tKO2.bam -b  ~/TheShell/SeqResults/R138\_Jin\_RiPr/06\_visualization/coverageBed/mm10\_kG\_exon\_mrna.bed -d > tKO2coverage.cov && \

coverageBed -split -abam ~/TheShell/SeqResults/R138\_Jin\_RiPr/03-1\_perfect\_hits/perfect\_hits\_tKO3.bam -b  ~/TheShell/SeqResults/R138\_Jin\_RiPr/06\_visualization/coverageBed/mm10\_kG\_exon\_mrna.bed -d > tKO3coverage.cov && \

coverageBed -split -abam ~/TheShell/SeqResults/R138\_Jin\_RiPr/03-1\_perfect\_hits/perfect\_hits\_WT1.bam -b  ~/TheShell/SeqResults/R138\_Jin\_RiPr/06\_visualization/coverageBed/mm10\_kG\_exon\_mrna.bed -d > WT1coverage.cov && \

coverageBed -split -abam ~/TheShell/SeqResults/R138\_Jin\_RiPr/03-1\_perfect\_hits/perfect\_hits\_WT2.bam -b  ~/TheShell/SeqResults/R138\_Jin\_RiPr/06\_visualization/coverageBed/mm10\_kG\_exon\_mrna.bed -d > WT2coverage.cov && \

coverageBed -split -abam ~/TheShell/SeqResults/R138\_Jin\_RiPr/03-1\_perfect\_hits/perfect\_hits\_WT3.bam -b  ~/TheShell/SeqResults/R138\_Jin\_RiPr/06\_visualization/coverageBed/mm10\_kG\_exon\_mrna.bed -d > WT3coverage.cov

**They are all exactly same format, so I can generate one single Bed file to load into R**

$ mkdir tmp

#for temporarily store file with column name. in the original example, it used space instead of tab. but mine is bed file, I think I have to use tap ("\t") in this case.

# cat > file is for creating a fille and cat file is to viewing a file

# ^D is <control+D> to tell Linux system that what is typed is to be stored in to the file locataions.bed

$ cat > tmp/locations.bed

chr start stop kG utrcds exonn strd symbol refseq exonntn

^D

#Now the header is added

#From one of the coverage file, extract location and annotation informations

$ cat tKO3coverage.cov | awk '{print $1"\t"$2"\t"$3"\t"$4"\t"$5"\t"$6"\t"$7"\t"$8"\t"$9"\t"$10}' >> tmp/locations.bed

# "\t " is a space after tab. this was problem for my previous table. now is "\t"

$ head -n 20 tmp/locations.bed

$ grep "Pten"  tmp/locations.bed

#generate 1 column version of coverage from 9 samples

$mkdir cov

# first add header row with sample name. original example used echo, but in our case we did not have header so follow previous example

$ cat  > cov/tg1.bed

tg1

^D

$ cat  > cov/tg2.bed

tg2

^D

$ cat  > cov/tg3.bed

tg3

^D

$ cat  > cov/tko1.bed

tko1

^D

$ cat  > cov/tko2.bed

tko2

^D

$ cat  > cov/tko3.bed

tko3

^D

$ cat  > cov/wt1.bed

wt1

^D

$ cat  > cov/wt2.bed

wt2

^D

$ cat  > cov/wt3.bed

wt3

^D

$ cat TG1coverage.cov | awk '{print $11}' >> cov/tg1.bed && \

cat TG2coverage.cov | awk '{print $11}' >> cov/tg2.bed && \

cat TG3coverage.cov | awk '{print $11}' >> cov/tg3.bed && \

cat tKO1coverage.cov | awk '{print $11}' >> cov/tko1.bed && \

cat tKO2coverage.cov | awk '{print $11}' >> cov/tko2.bed && \

cat tKO3coverage.cov | awk '{print $11}' >> cov/tko3.bed && \

cat WT1coverage.cov | awk '{print $11}' >> cov/wt1.bed && \

cat WT2coverage.cov | awk '{print $11}' >> cov/wt2.bed && \

cat WT3coverage.cov | awk '{print $11}' >> cov/wt3.bed

# Note that this method is not stable so do not execute other functions during this process

# paste them all together into a matrix. Note that every file should contain same row numbers

$ paste tmp/locations.bed cov/\* > coverage\_matrix\_all

$ head -30 coverage\_matrix\_all

**#now I have all 9 samples combined together into one file! (4.9G)**

**To make loading speed up, generate BED file with utr/cds to number [20140829]**

**Number codes are**

utr5-->0

cds-->1

utr3-->2

#From one of the coverage file, extract location and annotation informations

$ cat TG1coverage.cov | awk '{print $5"\t"$6"\t"$10}' >> tmp/coordinate.bed

#Change UTR and CDS lable to number

#one by one

$ sed "s/[[:<:]]utr5[[:>:]]/0/g" tmp/coordinate.bed

 #three together, byte changes are in the same file, and terminal frizzed when it was done

$ sed "s/[[:<:]]utr5[[:>:]]/0/g;s/[[:<:]]cds[[:>:]]/1/g;s/[[:<:]]utr3[[:>:]]/2/" tmp/coordinate.bed

#save as new file

$ sed "s/[[:<:]]utr5[[:>:]]/0/g;s/[[:<:]]cds[[:>:]]/1/g;s/[[:<:]]utr3[[:>:]]/2/" tmp/coordinate.bed > tmp/coordinate\_new.bed

#redirect output to new file or folder. if make output file name to original file name, it will overwrite it.

$ sed "s/[[:<:]]utr5[[:>:]]/0/g;s/[[:<:]]cds[[:>:]]/1/g;s/[[:<:]]utr3[[:>:]]/2/" tmp/coordinate.bed > temporary && mv temporary tmp/coordinate\_new.bed

#Check replacement has been finished to the very end

$tail -50 tmp/coordinate\_new.bed

(If you want to combine the three columns)

$ cat  > cov/coordinate\_combine.bed

cdsutr\_exonid\_nt

^D

$ cat  tmp/coordinate\_new.bed | awk '{print $1"\_"$2"\_"$3}' >> cov/coordinate\_combine.bed

#This also makes loading slower in R. i think number should be better for fast access

(But I decided to sepatate them)

$ cat  > cov/coordinate\_combine.bed

cdsutr exonid nt

^D

$ cat  tmp/coordinate\_new.bed | awk '{print $1"\t"$2"\t"$3}' >> cov/coordinate\_combine.bed

$ cat > tmp/locations2.bed

chr start stop kG strd symbol refseq

^D

$cat TG1coverage.cov | awk '{print $1"\t"$2"\t"$3"\t"$4"\t"$7"\t"$8"\t"$9}' >> tmp/locations2.bed

#Version with no refseq to save memory

$ cat > tmp/locations3.bed

chr start stop kG strd symbol

^D

$ cat TG1coverage.cov | awk '{print $1"\t"$2"\t"$3"\t"$4"\t"$7"\t"$8}' >> tmp/locations3.bed

#Finally generate new coverage file with shortened coordination

$ paste tmp/locations3.bed cov/\* > coverage\_matrix\_small

We will use this "coverage\_matrix\_small" file for downstream analysis in R

**03\_Coverage plotting in R**

Now we have a big table with coverage information, remaining is to visualize in R.

The table is huge (>4gB)

It will take up same amount of Ram so your ram capacity should be higher than the size of your table.

To open this huge size of table requires to open it with some tricks.

If you open this normal way, it can clash computer or can take overnight to open the table.

"freed" function in R library, "data.table" deal with this issue, so we can open the table in 10 minutes in R.

To learn how to explore the table in "data.table" please visit its vignette.

Important note:

data.table changed its logic to explore dataset slightly from version 1.9.4.

My following scripts are generated based on version 1.9.2, so you can either stick to 1.9.2 or change scripts accordingly as 1.9.4

To downgrade data.table version to 1.9.2, do follow.

> remove.packages("data.table")

> require(devtools)

> install\_version("data.table", version = "1.9.2", repos = "http://cran.us.r-project.org")

**03\_1\_footprint coverage of individual gene**

setwd("~/TheShell/SeqResults/R138\_Jin\_RiPr")

library(data.table)

library(ggplot2)

cov <- fread("coverage\_matrix\_small")

setkey(cov, symbol) #setkey based on gene symbol

g <- "Cd69"

cg <- cov[g]

strand <- cg$strd[1]

if(is.na(strand) ==TRUE) {

      next

    }

    if(strand == "+") {

       cg <- cg[order(cg$cdsutr, cg$exonid, cg$nt),]

     } else {

       cg <- cg[order(cg$cdsutr, -cg$exonid, -cg$nt),]

}

#Define WT coverage

beforestart <- tail(which(cg$cdsutr ==0), n=100)

afterstart <-head(which(cg$cdsutr ==1), n=100)

cdsregion <- c(beforestart, afterstart)

wt1r <- (cov[c(g), sum(wt1)])

wt2r <- (cov[c(g), sum(wt2)])

wt3r <- (cov[c(g), sum(wt3)])

wt.nrm.strt <-cg$wt1[cdsregion[1]:tail(cdsregion, n=1)]/(wt1r[wt1r[ ,symbol] == g, V1]+1) +cg$wt2[cdsregion[1]:tail(cdsregion, n=1)]/(wt2r[wt2r[ ,symbol] == g, V1]+1) +cg$wt3[cdsregion[1]:tail(cdsregion, n=1)]/(wt3r[wt3r[ ,symbol] == g, V1]+1)

#Define TG coverage

beforestart <- tail(which(cg$cdsutr ==0), n=100)

afterstart <-head(which(cg$cdsutr ==1), n=100)

cdsregion <- c(beforestart, afterstart)

tg1r <- (cov[c(g), sum(tg1)])

tg2r <- (cov[c(g), sum(tg2)])

tg3r <- (cov[c(g), sum(tg3)])

tg.nrm.strt <-cg$tg1[cdsregion[1]:tail(cdsregion, n=1)]/(tg1r[tg1r[ ,symbol] == g, V1]+1) +cg$tg2[cdsregion[1]:tail(cdsregion, n=1)]/(tg2r[tg2r[ ,symbol] == g, V1]+1) +cg$tg3[cdsregion[1]:tail(cdsregion, n=1)]/(tg3r[tg3r[ ,symbol] == g, V1]+1)

#The last line of scripts will normalize footprint abundance of individual replicate of a given gene, so that the "area under footprints" of all genes are normalized by individual replicates.

plot <- as.data.frame (cbind(wt.nrm.strt, tg.nrm.strt))

#ggplot to generate overlay graph

ggplot(plot, aes(x=as.numeric(row.names(plot))))+

  geom\_line(aes(y=wt.nrm.strt), colour="black", size=5)+

  geom\_line(aes(y=tg.nrm.strt), colour="green", size=5)+

  geom\_vline(xintercept=100, linetype="dashed", size=5, color="gray")+

  ggtitle(colnames(g))+

  labs(x="Start Codon", y="Relative Ribosome Occupancy")+

  theme (panel.background = element\_rect(fill='white'),  axis.text.x= element\_text(color="black"), axis.text.y = element\_blank(),  plot.title = element\_text(face="italic", size=14), axis.title = element\_text(size=15))

**03\_2\_footprint coverage of group of genes**

Basic scripts is the same, but "for-loop" can repeatedly execute the same function as instructed.

In our case, the group of genes are pre-defined, and ask to put the gene one by one, and print final results in a single table.

The results can be presented various way, but I used 15% trimmed mean value to exclude outlier including genes with no 5'UTR peak.

Here's an example from 5'UTR coverage of TG responsive targets (123) in WT and TG B cells

setwd("~/TheShell/SeqResults/R138\_Jin\_RiPr")

library(data.table)

library(ggplot2)

cov <- fread("coverage\_matrix\_small")

setkey(cov, symbol) #setkey based on gene symbol

# I have generated list of targets in a csv file, so you can import if from it

genelist <- read.table("genelist.csv", header=T, sep=",")

targets\_res\_tg <-as.vector(genelist$targets\_res\_tg)

targets\_res\_tg <-targets\_res\_tg[targets\_res\_tg !="" ]

#WT -->TG resp targets

wt.start.tgres <-matrix(0, nrow=200, ncol=0)

for(i in targets\_res\_tg) {

    g <- i

    cg <- cov[g]

    strand <- cg$strd[1]

    if(is.na(strand) ==TRUE) {

      next

    }

    if(strand == "+") {

       cg <- cg[order(cg$cdsutr, cg$exonid, cg$nt),]

     } else {

       cg <- cg[order(cg$cdsutr, -cg$exonid, -cg$nt),]

    }

    beforestart <- tail(which(cg$cdsutr ==0), n=100)

    afterstart <-head(which(cg$cdsutr ==1), n=100)

    cdsregion <- c(beforestart, afterstart)

    wt1r <- (cov[c(g), sum(wt1)])

    wt2r <- (cov[c(g), sum(wt2)])

    wt3r <- (cov[c(g), sum(wt3)])

    wt.nrm.strt <-cg$wt1[cdsregion[1]:tail(cdsregion, n=1)]/(wt1r[wt1r[ ,symbol] == g, V1]+1) +cg$wt2[cdsregion[1]:tail(cdsregion, n=1)]/(wt2r[wt2r[ ,symbol] == g, V1]+1) +cg$wt3[cdsregion[1]:tail(cdsregion, n=1)]/(wt3r[wt3r[ ,symbol] == g, V1]+1)

    if(length(cdsregion) == 200) {

    } else {

    cdsregion <- append(rep(c(0), each=200-length(cdsregion)), cdsregion)

    wt.nrm.strt <- append(rep(c(0), each=200-length(wt.nrm.strt)), wt.nrm.strt)

    }

  wt.start.tgres <- cbind(wt.start.tgres, wt.nrm.strt)

}

dim(wt.start.tgres) # 123columns

wt.start.tgres.trim<- apply(wt.start.tgres, 1, mean, trim=0.15)

#TG -->TG resp targets

tg.start.tgres <-matrix(0, nrow=200, ncol=0) #empty numeric vector (inclease speed)

for(i in targets\_res\_tg) {

    g <- i

    cg <- cov[g]

    strand <- cg$strd[1]

    if(is.na(strand) ==TRUE) {   # this is for avoid non-gene name matched ones

      next

    }

    if(strand == "+") {

       cg <- cg[order(cg$cdsutr, cg$exonid, cg$nt),]

     } else {

       cg <- cg[order(cg$cdsutr, -cg$exonid, -cg$nt),]

    }

    beforestart <- tail(which(cg$cdsutr ==0), n=100)

    afterstart <-head(which(cg$cdsutr ==1), n=100)

    cdsregion <- c(beforestart, afterstart)

    tg1r <- (cov[c(g), sum(tg1)])

    tg2r <- (cov[c(g), sum(tg2)])

    tg3r <- (cov[c(g), sum(tg3)])

    tg.nrm.strt <-cg$tg1[cdsregion[1]:tail(cdsregion, n=1)]/(tg1r[tg1r[ ,symbol] == g, V1]+1) +cg$tg2[cdsregion[1]:tail(cdsregion, n=1)]/(tg2r[tg2r[ ,symbol] == g, V1]+1) +cg$tg3[cdsregion[1]:tail(cdsregion, n=1)]/(tg3r[tg3r[ ,symbol] == g, V1]+1)

    if(length(cdsregion) == 200) {

    } else {

    cdsregion <- append(rep(c(0), each=200-length(cdsregion)), cdsregion)

    tg.nrm.strt <- append(rep(c(0), each=200-length(tg.nrm.strt)), tg.nrm.strt)

    }

  tg.start.tgres <- cbind(tg.start.tgres, tg.nrm.strt)

}

dim(tg.start.tgres) #123 columnes

tg.start.tgres.trim<- apply(tg.start.tgres, 1, mean, trim=0.15)

#generate matrix for graph

plot <- as.data.frame(cbind(wt.start.tgres.trim, tg.start.tgres.trim))

#Plotting: WT vs TG --> tg\_res\_targets

ggplot(plot, aes(x=as.numeric(row.names(plot))))+

  geom\_line(aes(y=wt.start.tgres.trim), colour="black", size=2)+

  geom\_line(aes(y=tg.start.tgres.trim), colour="green", size=2)+

  geom\_vline(xintercept=100, linetype="dashed", size=2, color="gray")+

  ggtitle("Ribosome on start codon region\_TGvsWT --> TGrespTargets")+

  labs(x="Start Codon", y="Relative Ribosome Occupancy")+

  geom\_segment(mapping=aes(x=100, xend=200, y=-0.0002, yend=-0.0002), size=12, color="dark grey") +

  geom\_segment(mapping=aes(x=0, xend=200, y=-0.0002, yend=-0.0002), size=4, color="dark grey")+

  theme (panel.background = element\_rect(fill='white'),  axis.text.x= element\_text(color="black"), axis.text.y = element\_blank(),  plot.title = element\_text(face="italic", size=14), axis.title = element\_text(size=15)) +

  scale\_y\_continuous(limit=c(-0.0003, 0.003))

The results should generate figure that will be attached

**Non-target analysis**

**20151209**

```{r}

#Gene list has been generated

setwd("~/TheShell/SeqResults/R138\_Jin\_RiPr")

library(data.table)

library(ggplot2)

cov <- fread("coverage\_matrix\_small")

setkey(cov, symbol) #setkey based on gene symbol

# Import genelist, from separate csv files

genelist <- read.table("06\_visualization/Coverage\_Nontarget\_forloop\_20151209/genelist\_nontarget\_noago.csv", header=T, sep=",")

nontarget\_down\_in\_tg <- as.vector(genelist$nontarget\_down\_in\_tg)

nontarget\_down\_in\_tg  <-nontarget\_down\_in\_tg[nontarget\_down\_in\_tg !="" ]

noago\_down\_in\_tg <- as.vector(genelist$noago\_down\_in\_tg)

noago\_down\_in\_tg  <-noago\_down\_in\_tg[noago\_down\_in\_tg !="" ]

nontarget\_up\_in\_tko <- as.vector(genelist$nontarget\_up\_in\_tko)

nontarget\_up\_in\_tko  <-nontarget\_up\_in\_tko[nontarget\_up\_in\_tko !="" ]

noago\_up\_in\_tko <- as.vector(genelist$noago\_up\_in\_tko)

noago\_up\_in\_tko  <-noago\_up\_in\_tko[noago\_up\_in\_tko !="" ]

```

#WT-TG comparison first

```{r}

#WT -->notarget\_down\_in\_tg

wt.start <-matrix(0, nrow=150, ncol=0)

for(i in nontarget\_down\_in\_tg) {

    g <- i

    cg <- cov[g]

    strand <- cg$strd[1]

    if(is.na(strand) ==TRUE) {

      next

    }

    if(strand == "+") {

       cg <- cg[order(cg$cdsutr, cg$exonid, cg$nt),]

     } else {

       cg <- cg[order(cg$cdsutr, -cg$exonid, -cg$nt),]

    }

    beforestart <- tail(which(cg$cdsutr ==0), n=100)

    afterstart <-head(which(cg$cdsutr ==1), n=50)

    cdsregion <- c(beforestart, afterstart)

    wt1r <- (cov[c(g), sum(wt1)])

    wt2r <- (cov[c(g), sum(wt2)])

    wt3r <- (cov[c(g), sum(wt3)])

    wt.nrm.strt <-cg$wt1[cdsregion[1]:tail(cdsregion, n=1)]/(wt1r[wt1r[ ,symbol] == g, V1]+1) +cg$wt2[cdsregion[1]:tail(cdsregion, n=1)]/(wt2r[wt2r[ ,symbol] == g, V1]+1) +cg$wt3[cdsregion[1]:tail(cdsregion, n=1)]/(wt3r[wt3r[ ,symbol] == g, V1]+1)

    if(length(cdsregion) == 150) {

    } else {

    cdsregion <- append(rep(c(0), each=150-length(cdsregion)), cdsregion)

    wt.nrm.strt <- append(rep(c(0), each=150-length(wt.nrm.strt)), wt.nrm.strt)

    }

  wt.start <- cbind(wt.start, wt.nrm.strt)

}

dim(wt.start) # supposed to 1044 but matched ones are 976

wt.start.trim<- apply(wt.start, 1, mean, trim=0.15)

#TG -->notarget\_down\_in\_tg

tg.start <-matrix(0, nrow=150, ncol=0) #empty numeric vector (inclease speed)

for(i in nontarget\_down\_in\_tg) {

    g <- i

    cg <- cov[g]

    strand <- cg$strd[1]

    if(is.na(strand) ==TRUE) {   # this is for avoid non-gene name matched ones

      next

    }

    if(strand == "+") {

       cg <- cg[order(cg$cdsutr, cg$exonid, cg$nt),]

     } else {

       cg <- cg[order(cg$cdsutr, -cg$exonid, -cg$nt),]

    }

    beforestart <- tail(which(cg$cdsutr ==0), n=100)

    afterstart <-head(which(cg$cdsutr ==1), n=50)

    cdsregion <- c(beforestart, afterstart)

    tg1r <- (cov[c(g), sum(tg1)])

    tg2r <- (cov[c(g), sum(tg2)])

    tg3r <- (cov[c(g), sum(tg3)])

    tg.nrm.strt <-cg$tg1[cdsregion[1]:tail(cdsregion, n=1)]/(tg1r[tg1r[ ,symbol] == g, V1]+1) +cg$tg2[cdsregion[1]:tail(cdsregion, n=1)]/(tg2r[tg2r[ ,symbol] == g, V1]+1) +cg$tg3[cdsregion[1]:tail(cdsregion, n=1)]/(tg3r[tg3r[ ,symbol] == g, V1]+1)

    if(length(cdsregion) == 150) {

    } else {

    cdsregion <- append(rep(c(0), each=150-length(cdsregion)), cdsregion)

    tg.nrm.strt <- append(rep(c(0), each=150-length(tg.nrm.strt)), tg.nrm.strt)

    }

  tg.start <- cbind(tg.start, tg.nrm.strt)

}

dim(tg.start)

tg.start.trim<- apply(tg.start, 1, mean, trim=0.15)

#generate matrix for graph

plot <- as.data.frame(cbind(wt.start.trim, tg.start.trim))

#Plotting: WT vs TG --> tg\_res\_targets

ggplot(plot, aes(x=as.numeric(row.names(plot))))+

  geom\_line(aes(y=wt.start.trim), colour="black", size=2)+

  geom\_line(aes(y=tg.start.trim), colour="green", size=2)+

  geom\_vline(xintercept=100, linetype="dashed", size=2, color="gray")+

  ggtitle("Nontargets\_down\_in\_tg")+

  labs(x="Start Codon", y="Relative Ribosome Occupancy")+

  geom\_segment(mapping=aes(x=100, xend=150, y=-0.0002, yend=-0.0002), size=12, color="dark grey") +

  geom\_segment(mapping=aes(x=0, xend=150, y=-0.0002, yend=-0.0002), size=4, color="dark grey")+

  theme (panel.background = element\_rect(fill='white'),  axis.text.x= element\_text(color="black"), axis.text.y = element\_blank(),  plot.title = element\_text(face="italic", size=14), axis.title = element\_text(size=15)) +

  scale\_y\_continuous(limit=c(-0.0003, 0.0045))

Save as 6x8

Save as 4x6

#WT -->noago\_down\_in\_tg

wt.start <-matrix(0, nrow=150, ncol=0)

for(i in noago\_down\_in\_tg) {

    g <- i

    cg <- cov[g]

    strand <- cg$strd[1]

    if(is.na(strand) ==TRUE) {

      next

    }

    if(strand == "+") {

       cg <- cg[order(cg$cdsutr, cg$exonid, cg$nt),]

     } else {

       cg <- cg[order(cg$cdsutr, -cg$exonid, -cg$nt),]

    }

    beforestart <- tail(which(cg$cdsutr ==0), n=100)

    afterstart <-head(which(cg$cdsutr ==1), n=50)

    cdsregion <- c(beforestart, afterstart)

    wt1r <- (cov[c(g), sum(wt1)])

    wt2r <- (cov[c(g), sum(wt2)])

    wt3r <- (cov[c(g), sum(wt3)])

    wt.nrm.strt <-cg$wt1[cdsregion[1]:tail(cdsregion, n=1)]/(wt1r[wt1r[ ,symbol] == g, V1]+1) +cg$wt2[cdsregion[1]:tail(cdsregion, n=1)]/(wt2r[wt2r[ ,symbol] == g, V1]+1) +cg$wt3[cdsregion[1]:tail(cdsregion, n=1)]/(wt3r[wt3r[ ,symbol] == g, V1]+1)

    if(length(cdsregion) == 150) {

    } else {

    cdsregion <- append(rep(c(0), each=150-length(cdsregion)), cdsregion)

    wt.nrm.strt <- append(rep(c(0), each=150-length(wt.nrm.strt)), wt.nrm.strt)

    }

  wt.start <- cbind(wt.start, wt.nrm.strt)

}

dim(wt.start) # supposed to 755 but matched ones are 687

wt.start.trim<- apply(wt.start, 1, mean, trim=0.15)

#TG -->nogo\_down\_in\_tg

tg.start <-matrix(0, nrow=150, ncol=0) #empty numeric vector (inclease speed)

for(i in noago\_down\_in\_tg) {

    g <- i

    cg <- cov[g]

    strand <- cg$strd[1]

    if(is.na(strand) ==TRUE) {   # this is for avoid non-gene name matched ones

      next

    }

    if(strand == "+") {

       cg <- cg[order(cg$cdsutr, cg$exonid, cg$nt),]

     } else {

       cg <- cg[order(cg$cdsutr, -cg$exonid, -cg$nt),]

    }

    beforestart <- tail(which(cg$cdsutr ==0), n=100)

    afterstart <-head(which(cg$cdsutr ==1), n=50)

    cdsregion <- c(beforestart, afterstart)

    tg1r <- (cov[c(g), sum(tg1)])

    tg2r <- (cov[c(g), sum(tg2)])

    tg3r <- (cov[c(g), sum(tg3)])

    tg.nrm.strt <-cg$tg1[cdsregion[1]:tail(cdsregion, n=1)]/(tg1r[tg1r[ ,symbol] == g, V1]+1) +cg$tg2[cdsregion[1]:tail(cdsregion, n=1)]/(tg2r[tg2r[ ,symbol] == g, V1]+1) +cg$tg3[cdsregion[1]:tail(cdsregion, n=1)]/(tg3r[tg3r[ ,symbol] == g, V1]+1)

    if(length(cdsregion) == 150) {

    } else {

    cdsregion <- append(rep(c(0), each=150-length(cdsregion)), cdsregion)

    tg.nrm.strt <- append(rep(c(0), each=150-length(tg.nrm.strt)), tg.nrm.strt)

    }

  tg.start <- cbind(tg.start, tg.nrm.strt)

}

dim(tg.start)

tg.start.trim<- apply(tg.start, 1, mean, trim=0.15)

#generate matrix for graph

plot <- as.data.frame(cbind(wt.start.trim, tg.start.trim))

#Plotting: WT vs TG

ggplot(plot, aes(x=as.numeric(row.names(plot))))+

  geom\_line(aes(y=wt.start.trim), colour="black", size=2)+

  geom\_line(aes(y=tg.start.trim), colour="green", size=2)+

  geom\_vline(xintercept=100, linetype="dashed", size=2, color="gray")+

  ggtitle("No\_ago\_down\_in\_tg")+

  labs(x="Start Codon", y="Relative Ribosome Occupancy")+

  geom\_segment(mapping=aes(x=100, xend=150, y=-0.0002, yend=-0.0002), size=12, color="dark grey") +

  geom\_segment(mapping=aes(x=0, xend=150, y=-0.0002, yend=-0.0002), size=4, color="dark grey")+

  theme (panel.background = element\_rect(fill='white'),  axis.text.x= element\_text(color="black"), axis.text.y = element\_blank(),  plot.title = element\_text(face="italic", size=14), axis.title = element\_text(size=15)) +

  scale\_y\_continuous(limit=c(-0.0003, 0.0048))

Save as 6x8

Save as 4x6

```

#WT-TKO comparison next

```{r}

#WT -->notarget\_up\_in\_tko

wt.start <-matrix(0, nrow=150, ncol=0)

for(i in nontarget\_up\_in\_tko) {

    g <- i

    cg <- cov[g]

    strand <- cg$strd[1]

    if(is.na(strand) ==TRUE) {

      next

    }

    if(strand == "+") {

       cg <- cg[order(cg$cdsutr, cg$exonid, cg$nt),]

     } else {

       cg <- cg[order(cg$cdsutr, -cg$exonid, -cg$nt),]

    }

    beforestart <- tail(which(cg$cdsutr ==0), n=100)

    afterstart <-head(which(cg$cdsutr ==1), n=50)

    cdsregion <- c(beforestart, afterstart)

    wt1r <- (cov[c(g), sum(wt1)])

    wt2r <- (cov[c(g), sum(wt2)])

    wt3r <- (cov[c(g), sum(wt3)])

    wt.nrm.strt <-cg$wt1[cdsregion[1]:tail(cdsregion, n=1)]/(wt1r[wt1r[ ,symbol] == g, V1]+1) +cg$wt2[cdsregion[1]:tail(cdsregion, n=1)]/(wt2r[wt2r[ ,symbol] == g, V1]+1) +cg$wt3[cdsregion[1]:tail(cdsregion, n=1)]/(wt3r[wt3r[ ,symbol] == g, V1]+1)

    if(length(cdsregion) == 150) {

    } else {

    cdsregion <- append(rep(c(0), each=150-length(cdsregion)), cdsregion)

    wt.nrm.strt <- append(rep(c(0), each=150-length(wt.nrm.strt)), wt.nrm.strt)

    }

  wt.start <- cbind(wt.start, wt.nrm.strt)

}

dim(wt.start) # supposed to 781 but matched ones are 730

wt.start.trim<- apply(wt.start, 1, mean, trim=0.15)

#TKO -->nontarget\_up\_in\_tko

tko.start <-matrix(0, nrow=150, ncol=0) #empty numeric vector (inclease speed)

for(i in nontarget\_up\_in\_tko) {

    g <- i

    cg <- cov[g]

    strand <- cg$strd[1]

    if(is.na(strand) ==TRUE) {   # this is for avoid non-gene name matched ones

      next

    }

    if(strand == "+") {

       cg <- cg[order(cg$cdsutr, cg$exonid, cg$nt),]

     } else {

       cg <- cg[order(cg$cdsutr, -cg$exonid, -cg$nt),]

    }

    beforestart <- tail(which(cg$cdsutr ==0), n=100)

    afterstart <-head(which(cg$cdsutr ==1), n=50)

    cdsregion <- c(beforestart, afterstart)

    tko1r <- (cov[c(g), sum(tko1)])

    tko2r <- (cov[c(g), sum(tko2)])

    tko3r <- (cov[c(g), sum(tko3)])

    tko.nrm.strt <-cg$tko1[cdsregion[1]:tail(cdsregion, n=1)]/(tko1r[tko1r[ ,symbol] == g, V1]+1) +cg$tko2[cdsregion[1]:tail(cdsregion, n=1)]/(tko2r[tko2r[ ,symbol] == g, V1]+1) +cg$tko3[cdsregion[1]:tail(cdsregion, n=1)]/(tko3r[tko3r[ ,symbol] == g, V1]+1)

    if(length(cdsregion) == 150) {

    } else {

    cdsregion <- append(rep(c(0), each=150-length(cdsregion)), cdsregion)

    tko.nrm.strt <- append(rep(c(0), each=150-length(tko.nrm.strt)), tko.nrm.strt)

    }

  tko.start <- cbind(tko.start, tko.nrm.strt)

}

dim(tko.start)

tko.start.trim<- apply(tko.start, 1, mean, trim=0.15)

#generate matrix for graph

plot <- as.data.frame(cbind(wt.start.trim, tko.start.trim))

#Plotting: WT vs TKO

ggplot(plot, aes(x=as.numeric(row.names(plot))))+

  geom\_line(aes(y=wt.start.trim), colour="black", size=2)+

  geom\_line(aes(y=tko.start.trim), colour="red", size=2)+

  geom\_vline(xintercept=100, linetype="dashed", size=2, color="gray")+

  ggtitle("Nontargets\_up\_in\_ko")+

  labs(x="Start Codon", y="Relative Ribosome Occupancy")+

  geom\_segment(mapping=aes(x=100, xend=150, y=-0.0002, yend=-0.0002), size=12, color="dark grey") +

  geom\_segment(mapping=aes(x=0, xend=150, y=-0.0002, yend=-0.0002), size=4, color="dark grey")+

  theme (panel.background = element\_rect(fill='white'),  axis.text.x= element\_text(color="black"), axis.text.y = element\_blank(),  plot.title = element\_text(face="italic", size=14), axis.title = element\_text(size=15)) +

  scale\_y\_continuous(limit=c(-0.0003, 0.0045))

Save as 6x8

Save as 4x6

#WT -->noago\_up\_in\_tko

wt.start <-matrix(0, nrow=150, ncol=0)

for(i in noago\_up\_in\_tko) {

    g <- i

    cg <- cov[g]

    strand <- cg$strd[1]

    if(is.na(strand) ==TRUE) {

      next

    }

    if(strand == "+") {

       cg <- cg[order(cg$cdsutr, cg$exonid, cg$nt),]

     } else {

       cg <- cg[order(cg$cdsutr, -cg$exonid, -cg$nt),]

    }

    beforestart <- tail(which(cg$cdsutr ==0), n=100)

    afterstart <-head(which(cg$cdsutr ==1), n=50)

    cdsregion <- c(beforestart, afterstart)

    wt1r <- (cov[c(g), sum(wt1)])

    wt2r <- (cov[c(g), sum(wt2)])

    wt3r <- (cov[c(g), sum(wt3)])

    wt.nrm.strt <-cg$wt1[cdsregion[1]:tail(cdsregion, n=1)]/(wt1r[wt1r[ ,symbol] == g, V1]+1) +cg$wt2[cdsregion[1]:tail(cdsregion, n=1)]/(wt2r[wt2r[ ,symbol] == g, V1]+1) +cg$wt3[cdsregion[1]:tail(cdsregion, n=1)]/(wt3r[wt3r[ ,symbol] == g, V1]+1)

    if(length(cdsregion) == 150) {

    } else {

    cdsregion <- append(rep(c(0), each=150-length(cdsregion)), cdsregion)

    wt.nrm.strt <- append(rep(c(0), each=150-length(wt.nrm.strt)), wt.nrm.strt)

    }

  wt.start <- cbind(wt.start, wt.nrm.strt)

}

dim(wt.start) # supposed to 627 matched 578

wt.start.trim<- apply(wt.start, 1, mean, trim=0.15)

#TKO -->noago\_up\_in\_tko

tko.start <-matrix(0, nrow=150, ncol=0) #empty numeric vector (inclease speed)

for(i in noago\_up\_in\_tko) {

    g <- i

    cg <- cov[g]

    strand <- cg$strd[1]

    if(is.na(strand) ==TRUE) {   # this is for avoid non-gene name matched ones

      next

    }

    if(strand == "+") {

       cg <- cg[order(cg$cdsutr, cg$exonid, cg$nt),]

     } else {

       cg <- cg[order(cg$cdsutr, -cg$exonid, -cg$nt),]

    }

    beforestart <- tail(which(cg$cdsutr ==0), n=100)

    afterstart <-head(which(cg$cdsutr ==1), n=50)

    cdsregion <- c(beforestart, afterstart)

    tko1r <- (cov[c(g), sum(tko1)])

    tko2r <- (cov[c(g), sum(tko2)])

    tko3r <- (cov[c(g), sum(tko3)])

    tko.nrm.strt <-cg$tko1[cdsregion[1]:tail(cdsregion, n=1)]/(tko1r[tko1r[ ,symbol] == g, V1]+1) +cg$tko2[cdsregion[1]:tail(cdsregion, n=1)]/(tko2r[tko2r[ ,symbol] == g, V1]+1) +cg$tko3[cdsregion[1]:tail(cdsregion, n=1)]/(tko3r[tko3r[ ,symbol] == g, V1]+1)

    if(length(cdsregion) == 150) {

    } else {

    cdsregion <- append(rep(c(0), each=150-length(cdsregion)), cdsregion)

    tko.nrm.strt <- append(rep(c(0), each=150-length(tko.nrm.strt)), tko.nrm.strt)

    }

  tko.start <- cbind(tko.start, tko.nrm.strt)

}

dim(tko.start)

tko.start.trim<- apply(tko.start, 1, mean, trim=0.15)

#generate matrix for graph

plot <- as.data.frame(cbind(wt.start.trim, tko.start.trim))

#Plotting: WT vs TKO

ggplot(plot, aes(x=as.numeric(row.names(plot))))+

  geom\_line(aes(y=wt.start.trim), colour="black", size=2)+

  geom\_line(aes(y=tko.start.trim), colour="red", size=2)+

  geom\_vline(xintercept=100, linetype="dashed", size=2, color="gray")+

  ggtitle("Noago\_up\_in\_ko")+

  labs(x="Start Codon", y="Relative Ribosome Occupancy")+

  geom\_segment(mapping=aes(x=100, xend=150, y=-0.0002, yend=-0.0002), size=12, color="dark grey") +

  geom\_segment(mapping=aes(x=0, xend=150, y=-0.0002, yend=-0.0002), size=4, color="dark grey")+

  theme (panel.background = element\_rect(fill='white'),  axis.text.x= element\_text(color="black"), axis.text.y = element\_blank(),  plot.title = element\_text(face="italic", size=14), axis.title = element\_text(size=15)) +

  scale\_y\_continuous(limit=c(-0.0003, 0.0045))

Save as 6x8

Save as 4x6

```