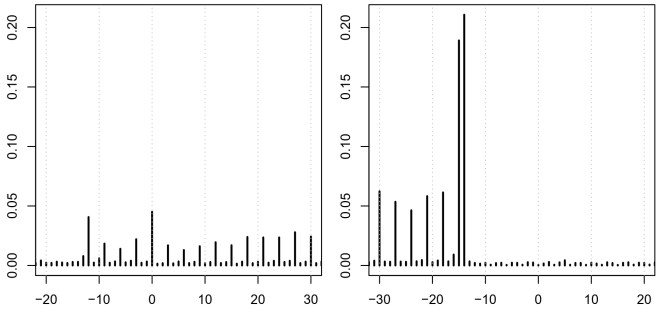
Rotation in Murre lab
Building Riboprofiling Pipeline

Pin-Chung (Tony) Cheng

Workflow

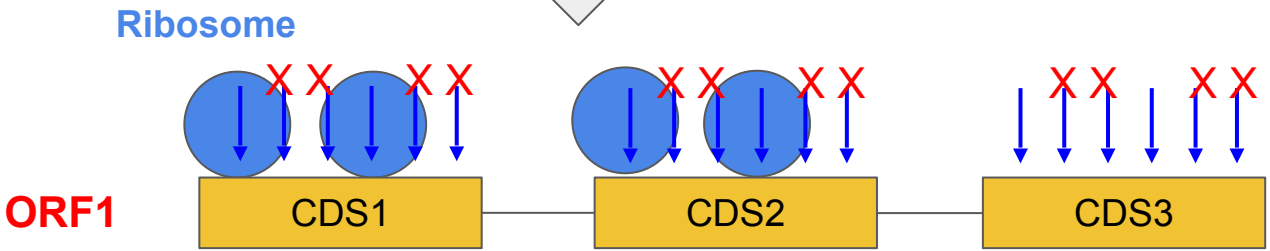
Arthur's Pipeline

-Demultiplex
-Read alignment
-Filter rRNA reads
-Filter read length
…
…



offsetCorrect.sam -> filtered reads files

repre*.gtf -> predicted ORF file



Pre-filter reads on terminal

```
# Step 0: Run on terminal

# Create read bed file from sam file
sh_name = paste0(script.basename, "/Codes/sam_to_bed_strand_v2.sh")
read_basename = sub(".sam", "", basename(opt$file))
pre_bed_name = paste0(script.basename, "/Temp/", read_basename, "_pre_bedtools_intersect.tsv")

system(paste(sh_name, opt$file, pre_bed_name))

# Create gtf bed file and filter exon for pre bedtools intersect in linux
repre_gtf_bed = paste0(script.basename, "/Temp/", read_basename, "_repre_gtf2bed.bed")

# need a lot of \ to escape quotation mark
# uniq again, becasue orfID can have many same exon coordinate > falsely increase read count
system(paste("grep \'exon\\s\'", opt$gtf, "| awk \'{print $1 \"\t\" $4 \"\t\" $5 \"\t\" $7}\' | uniq >", repre_gtf_bed))


# Run bedtools intersect, -s: match strand, -wa: if A intersect B, keep A
post_bed_name = paste0(script.basename, "/Temp/", read_basename, "_post_bedtools_intersect.tsv")

system(paste("bedtools intersect -a", pre_bed_name, "-b", repre_gtf_bed, ">", post_bed_name))
```

```r
# Step 1: Read in "repre.valid.ORF.*gtf" and prepare for intersect

gtf = read.table(opt$gtf, sep="\t", stringsAsFactors=FALSE)

# Remove duplicated rows, if input is from multiple gtf file concatenated from multiple samples
gtf = distinct(gtf)

# Select the columns we need and filter only "CDS"
gtf_selected = gtf %>%
    select(V1, V3, V4, V5, V8, V9, V7) %>%
    filter(V3 == "CDS")


# extract orf_id
gtf_selected$V9 = str_match(gtf_selected$V9,"\\s(ENST[0-9]+.*:chr.+);\\stranscript_id")[,2]

colnames(gtf_selected) = c("chr", "CDS", "str", "end", "frame", "orf_id", "strand_col")

# remove CDS column
gtf_bed = gtf_selected[,-2]

# change to appropiate column data type
gtf_bed$frame = as.numeric(gtf_bed$frame)
gtf_bed$orf_id = as.character(gtf_bed$orf_id)

gtf_bed = arrange(gtf_bed, orf_id)



# -------------------------
# Step 1.2: Create a merged ORF table for annotation and a dataframe for intersect

# find identical elements in a vector, return a list of positions(vector), need to sort the vector first
find_identical_element = function(x){

    listv = list()

    # first index
    first_idx_v = which(!duplicated(x))
    last_idx_v = c((first_idx_v - 1)[-1], length(x))
```

Read in gtf file,
manipulate dataframe

```r
# Step 1.3: Create Annotation table for merged_ORF to geneID

# Create mORF_table by adding column "geneID" "tx_id" "merged_orfID" > for annotation
tx_id_vec = str_extract(gtf_bed$orf_id, "ENST[0-9]+")

idx = match(tx_id_vec, anno$target_id)

mORF_table = gtf_bed %>%
            mutate(tx_id = tx_id_vec, ens_gene = anno$ens_gene[idx], ext_gene = anno$ext_gene[idx],
                   merged_orfID = as.character(mORF_list[gtf_bed$orf_id]))


# Use distinct() to merge same merged_ORF (CDSs string)
mORF_table = mORF_table %>%
    select(tx_id, ens_gene, ext_gene, merged_orfID) %>%  # Note: merged_ORF can match many tx_id, here just show one
    distinct(merged_orfID, .keep_all = TRUE) %>%
    arrange(ext_gene)


# create a vector of number to add to geneID, eg. A2M.1, A2M.2, A2M.3, AAAS.1 ...
gene_count = mORF_table %>% count(ext_gene)

gene_nvec = c()
for (i in 1:nrow(gene_count)){

    # create a vector for each geneID, starting from 1
    v = 1:gene_count$n[i]

    gene_nvec = append(gene_nvec, v)
}


# Add number to geneID
mORF_table = mORF_table %>%
    mutate(ens_gene = paste0(ens_gene, ".", gene_nvec), ext_gene = paste0(ext_gene, ".", gene_nvec))

# Replace output NA.$number with NA
NA_idx = str_which(mORF_table$ext_gene, "NA.*")
mORF_table$ens_gene[NA_idx] = NA


```

Different ORFs have same set of CDSs > create merge ORF

```r
# ------------------------
# Step 3: Intersect reads and mergeORF
# query (reads) is the first argument, subejct(ORF) the second

list = bedtools_intersect(reads, mORF_df, ignore_strand=FALSE)

overlap = list$overlap


# Use dplyr package, do all steps by piping:
    # Add read_position, cds_start, cds_frame by index;
    # Add remainder column by calculating (read_pos - cds_start - cds frame)/3 and output remainder
    # Add tx_id column by index

result = overlap %>%
    mutate(read_pos = reads[overlap$bed1_idx, 2], cds_start = mORF_df[overlap$bed2_idx, 2],
           cds_frame = mORF_df[overlap$bed2_idx, 4], remainder = (read_pos - cds_start - cds_frame) %% 3,
           orfID = mORF_df[overlap$bed2_idx, 5])

# Select result that are in frame
result_inframe = result %>%
    filter(remainder == 0)


# Number of in-frame reads
paste0(" in_frame reads: ", nrow(result_inframe))
paste0(" total reads: ", nrow(result))


result_f1 = result %>%
    filter(remainder==0) %>%
    count(orfID)

colnames(result_f1) = c("merged_orfID", "f1Num")

result_f2 = result %>%
    filter(remainder==1) %>%
    count(orfID)

colnames(result_f2) = c("merged_orfID", "f2Num")
```

Intersect,
filtering read by frame f1, f2, f3,
count

Run Rscript in one step

Run steps on server terminal

Copy package “Ribo_Tony” to your directory:

```
cp -r /home/tonycheng/Tools/Ribo_Tony PATH/TO/YOUR_DIRECTORY
```

Create environment:

```
conda env create -f tony_r_env.yml
```

Activate environment:

```
conda activate tony_r
```

Run Rscript in one step

Input: offset corrected sam file and re.pred.valid.gtf file (can be a concatenated gtf from multiple samples)

Output: a table with Gene ID and read counts on codon frame 1, 2, 3 position

Rscript PATH/TO/Ribo_Tony/Intersect_mergeORF.R -f ${PATH/TO/*offsetCorrect.sam} -g ${PATH/TO/repre*.gtf} -o ${PATH/TO/OUTPUT_FILENAME.tsv}

Example code:

```
Rscript /home/tonycheng/Tools/Ribo_Tony/Intersect_mergeORF.R -f /home/tonycheng/Data/
Fleur_data/sam/GSC2_t2.sam -g /home/tonycheng/Data/Fleur_data/gtf/
GSC_total_uniq.repre.valid.ORF.gtf -o /home/tonycheng/Data/Fleur_data/GSC2_t2_mergeORF.tsv
```

Prepare a concatenated gtf file

Concatenate, sort, and remove duplicate row

cat ${gtf file1} ${gtf file2} ${gtf file3} | sort | uniq > ${output_filename}

Create a directory with all the sample gtf files

mkdir ${folder_name}

Example code:

```
cat /PATH/TO/DIRECTORY/*.gtf | sort | uniq > GSC_total_uniq.repre.valid.ORF.gtf
```

f1Num	f2Num	f3Num	ens_gene	ext_gene	merged_orfID
23718	1739	2894	ENSG00000026025.2	VIM.2	17229423:17229985:0|17230650:17230710:1
:17236379:2|17237230:17237268:0|chr10;+					
21846	1520	2609	ENSG00000026025.11	VIM.11	17229423:17229985:0|17230650:17230710:1
13003	778	1421	ENSG00000026025.10	VIM.10	17229423:17229985:0|17230650:17230785:1
9816	791	1373	ENSG00000026025.8	VIM.8	17230633:17230710:0|17233587:17233682:0
9556	791	1338	ENSG00000026025.9	VIM.9	17230663:17230710:0|17233587:17233682:0
8355	2446	6811	ENSG00000222293.1	RNU2-36P.1	86423020:86423034:0|chr9;-
6464	485	929	ENSG00000026025.4	VIM.4	17230633:17230710:0|17233587:17233682:0
6204	485	894	ENSG00000026025.5	VIM.5	17230663:17230710:0|17233587:17233682:0
4947	411	704	ENSG00000026025.12	VIM.12	17233605:17233682:0|17233770:17233931:0
4896	6	774	ENSG00000062716.1	VMP1.1	59841265:59841285:0|chr17;+
4765	325	568	ENSG00000026508.7	CD44.7	35139304:35139370:0|35176575:35176740:2
:35201195:2|35201671:35201787:2|35204512:35204640:2|35206112:35206243:2|35208105:35208206:2|352					
29330:1|chr11;+					
4764	325	567	ENSG00000026508.3	CD44.3	35139304:35139370:0|35176575:35176740:2
:35201787:2|35204512:35204640:2|35206112:35206243:2|35208105:35208206:2|35209965:35210054:2|352					
4743	325	566	ENSG00000026508.8	CD44.8	35139304:35139370:0|35176575:35176740:2
:35211449:2|35214852:35214914:2|35219316:35219387:2|35221654:35221732:2|35229129:35229330:1|chr					
4741	325	566	ENSG00000026508.9	CD44.9	35139304:35139370:0|35176575:35176740:2
:35219387:2|35221654:35221732:2|35229129:35229330:1|chr11;+					
4723	325	566	ENSG00000026508.1	CD44.1	35139304:35139370:0|35176575:35176740:2
:35221732:2|35229129:35229330:1|chr11;+					
4376	342	523	ENSG00000135046.2	ANXA1.2	73158536:73158601:0|73158695:73158803:0
:73165209:0|73166097:73166192:2|73167497:73167555:2|73169032:73169154:0|73170051:73170104:0|chr					
4075	273	452	ENSG00000026508.17	CD44.17	35139304:35139370:0|35176575:35176740:2
4060	521	263	ENSG00000150991.7	UBC.7	124911717:124912180:2|124913321:1249137
3890	347	440	ENSG00000134531.2	EMP1.2	13211511:13211588:0|13213479:13213575:0
3789	284	606	ENSG00000026025.13	VIM.13	17230663:17230710:0|17233587:17233682:0
3715	222	468	ENSG00000111640.3	GAPDH.3	6534833:6534861:0|6536494:6536593:1|653
2;+					
3673	300	445	ENSG00000135046.1	ANXA1.1	73158794:73158803:0|73159329:73159423:2
:73166192:2|73167497:73167555:2|73169032:73169154:0|73170051:73170104:0|chr9;+					
3628	693	485	ENSG00000242265.15	PFC10.15	24656587:24656591:0|24663334:24

Differential Expression Analysis

Fluer's Data

Two patient sample: GSC2, GSC20

Three conditions: t=0, t=1, t=2

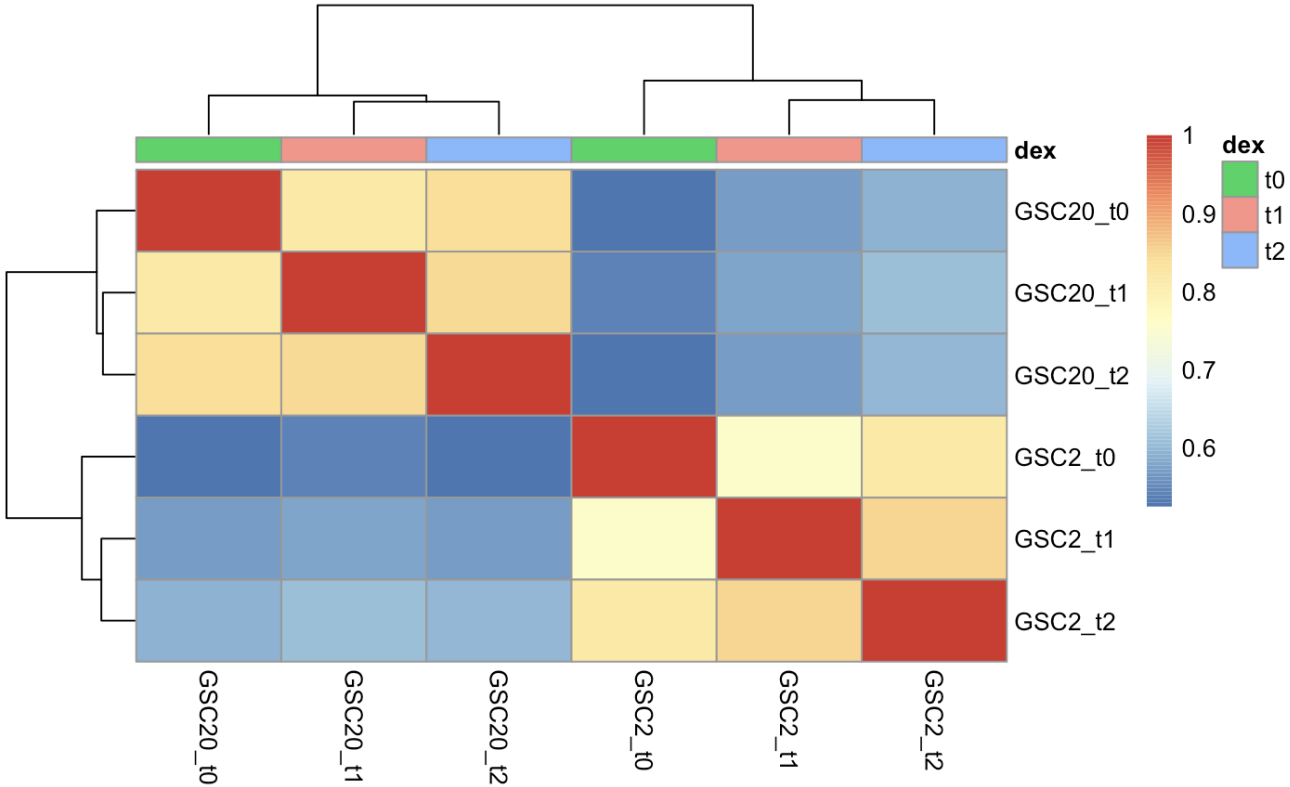
Gene <chr>	GSC2_t0 <int>	GSC2_t1 <int>	GSC2_t2 <int>	GSC20_t0 <int>	GSC20_t1 <int>	GSC20_t2 <int>
VIM.2	3611	5900	23718	2071	3315	4103
VIM.11	3418	5561	21846	2014	3133	4006
VIM.10	2192	4006	13003	1471	2242	2986
RNU2-57P.1	1466	1570	2486	*NA*	*NA*	*NA*
RNU2-26P.1	1415	1620	2349	*NA*	0	*NA*
VIM.8	1270	1775	9816	592	898	1165
VIM.9	1245	1745	9556	557	883	1120
RNU2-36P.1	883	770	8355	1	32	2
VIM.4	772	1129	6464	400	610	813
VIM.5	747	1099	6204	365	595	768

1–10 of 38,707 rows

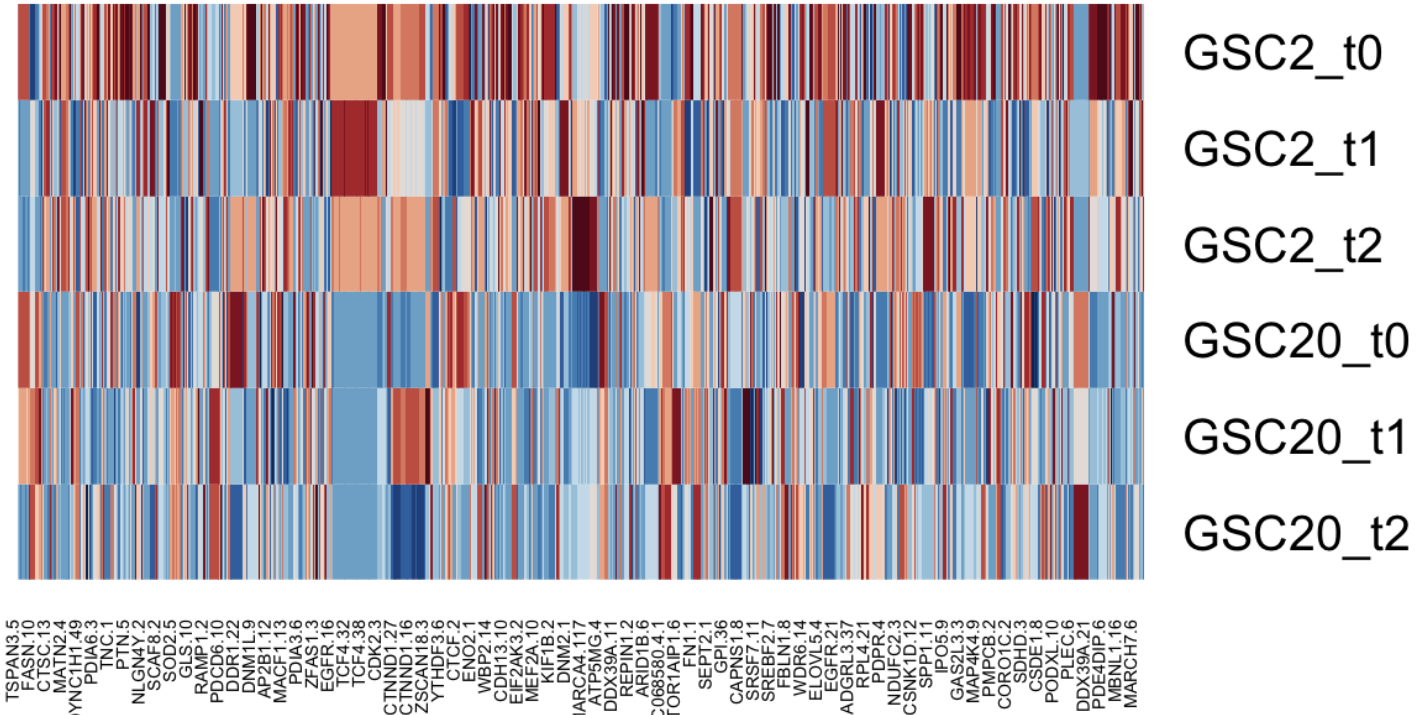
Previous 1 2 3 4 5 6 … 100 Next

Heatmap

Samples are more correlated between patients rather than conditions



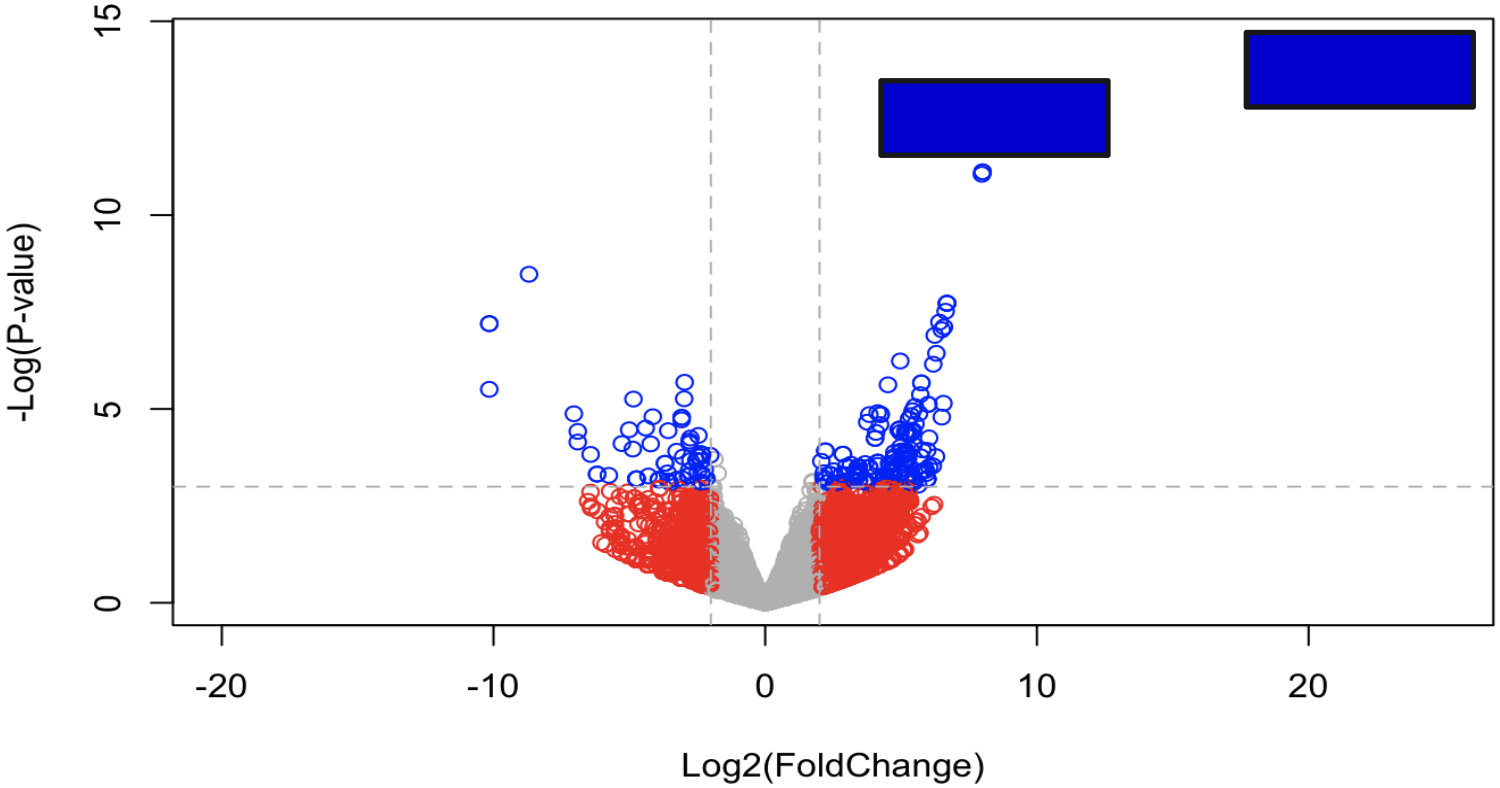
Heatmap

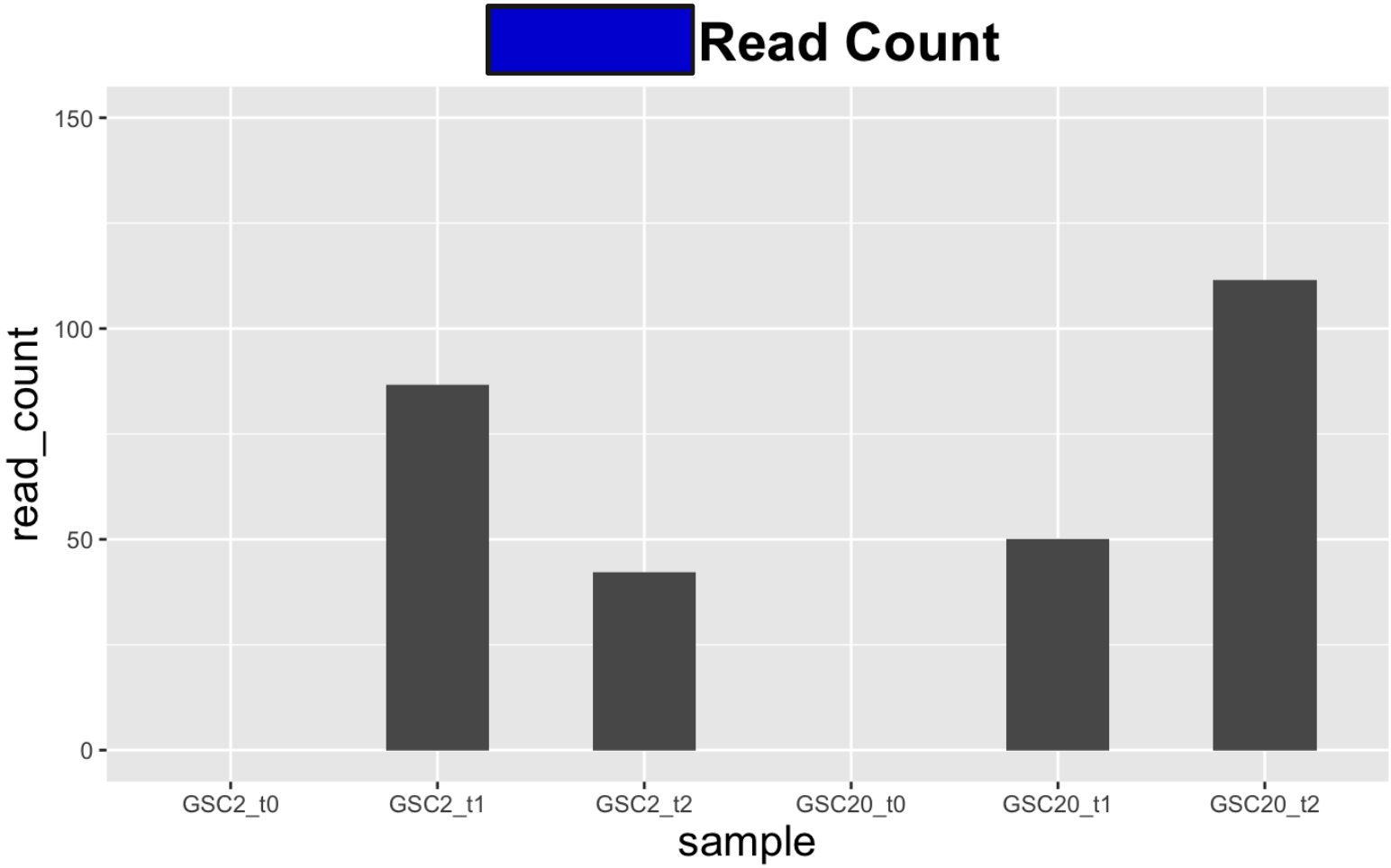

GSC2_t0
GSC2_t1
GSC2_t2
GSC20_t0
GSC20_t1
GSC20_t2
TSPAN3.5
FASN.10
CTSC.13
MATN2.4
DYNC1H1.49
PDIA6.3
TNC.1
PTN.5
NLGN4Y.2
SCAF8.2
SOD2.5
GLS.10
RAMP1.2
PDCD6.10
DDR1.22
DNM1L.9
AP2B1.12
MACF1.13
PDIA3.6
ZFAS1.3
EGFR.16
TCF4.32
TCF4.38
CDK2.3
CTNND1.27
CTNND1.16
ZSCAN18.3
YTHDF3.6
CTCF.2
ENO2.1
WBP2.14
CDH13.10
EIF2AK3.2
MEF2A.10
KIF1B.2
DNM2.1
SMARCA4.117
ATP5MG.4
DDX39A.11
REPIN1.2
ARID1B.6
AC068580.4.1
TOR1AIP1.6
FN1.1
SEPT2.1
GPI.36
CAPNS1.8
SRSF7.11
SREBF2.7
FBLN1.8
WDR6.14
ELOVL5.4
EGFR.21
ADGRL3.37
RPL4.21
PDPR.4
NDUFC2.3
CSNK1D.12
SPP1.11
IPO5.9
GAS2L3.3
MAP4K4.9
PMPCB.2
CORO1C.2
SDHD.3
CSDE1.8
PODXL.10
PLEC.6
DDX39A.21
PDE4DIP.6
MBNL1.16
MARCH7.6

t2 vs t0: order by log2FoldChange

	baseMean <dbl>	log2FoldChange <dbl>	lfcSE <dbl>	stat <dbl>	pvalue <dbl>	padj <dbl>
	275.19264	24.027155	4.784769	5.021591	5.124517e-07	0.01978679
	48.43936	7.999540	1.847291	4.330417	1.488272e-05	0.15295725
	48.43936	7.999540	1.847291	4.330417	1.488272e-05	0.15295725
	46.13576	7.977077	1.848003	4.316593	1.584557e-05	0.15295725
	14.37565	6.691227	1.904044	3.514218	4.410506e-04	0.99999629
	14.37565	6.691227	1.904044	3.514218	4.410506e-04	0.99999629
	14.37565	6.691227	1.904044	3.514218	4.410506e-04	0.99999629
	15.65169	6.640647	1.919939	3.458780	5.426281e-04	0.99999629
	16.89030	6.578488	1.965008	3.347818	8.145057e-04	0.99999629
	10.20756	6.559565	2.378785	2.757527	5.824039e-03	0.99999629

Volcano Plot




Read Count
150
100
50
0
read_count
GSC2_t0
GSC2_t1
GSC2_t2
GSC20_t0
GSC20_t1
GSC20_t2
sample

rep("Thank you!", 100)

😀