# Ribo-seq-analysis

**How to generate annotation files used in the analysis:**

(For this analysis, we used gtf downloaded from UCSC, CDS fasta downloaded from Ensembl, and transcriptome fasta downloaded from NCBI Refseq)

First, prepare the annotation file for the generation of bed files:

gffread annotation.gtf -T -o annotation.gtf

gtf2bed --gtf annotation.gtf --bed annotation.bed

#generation of CDS annotation

awk '$3 == "CDS" {

match($0, /gene\_name "([^"]+)"/, gname);

if (gname[1] != "") print $1"\t"$4-1"\t"$5"\t"gname[1]"\t.\t"$7

}' annotation.gtf > annotation\_CDS.bed

#generation for CDS annotation with exclusion of the first 18 and last 15 nucleotides:

awk '$3 == "CDS" {

print $1 "\t" ($4-1) "\t" ($4+17) "\t.\t.\t" $7; # First 18 nt

print $1 "\t" ($5-15) "\t" $5 "\t.\t.\t" $7; # Last 15 nt

}' annotation.gtf > exclude\_regions.bed

bedtools subtract -a annotation\_CDS.bed -b exclude\_regions.bed > filtered\_CDS.bed

#generation of 3'UTR annotation non-overlaping with CDS:

awk -F'\t' '

$3 == "CDS" {

match($9, /gene\_id "([^"]+)"/, gid);

match($9, /transcript\_id "([^"]+)"/, tid);

gene\_id = (gid[1] != "") ? gid[1] : "NA";

transcript\_id = (tid[1] != "") ? tid[1] : "NA";

transcript\_key = gene\_id ";" transcript\_id;

if (!last\_CDS\_end[transcript\_key] || $5 > last\_CDS\_end[transcript\_key]) {

last\_CDS\_end[transcript\_key] = $5;

last\_CDS\_chr[transcript\_key] = $1;

last\_CDS\_strand[transcript\_key] = $7;

}

}

END {

for (transcript\_key in last\_CDS\_end) {

chr = last\_CDS\_chr[transcript\_key];

start = last\_CDS\_end[transcript\_key];

end = (last\_CDS\_strand[transcript\_key] == "+") ? start + 100 : start - 100;

if (last\_CDS\_strand[transcript\_key] == "+") {

print chr "\t" start "\t" end "\t" transcript\_key "\t.\t+";

} else {

print chr "\t" end "\t" start "\t" transcript\_key "\t.\t-";

}

}

}' annotation.gtf > initial\_3UTR.bed

bedtools subtract -a initial\_3UTR.bed -b annotation\_CDS.bed > 3UTR.bed

bedtools intersect -a 3UTR.bed -b filtered\_CDS.bed -wa -wb > overlaps\_check.txt #check for overlapping

#generation of stop codon annotation:

awk -F'\t' '

$3 == "CDS" {

match($9, /gene\_id "([^"]+)"/, gid);

match($9, /transcript\_id "([^"]+)"/, tid);

gene\_id = (gid[1] != "") ? gid[1] : "NA";

transcript\_id = (tid[1] != "") ? tid[1] : "NA";

transcript\_key = gene\_id ";" transcript\_id;

if (!last\_CDS\_end[transcript\_key] || $5 > last\_CDS\_end[transcript\_key]) {

last\_CDS\_end[transcript\_key] = $5;

last\_CDS\_chr[transcript\_key] = $1;

last\_CDS\_strand[transcript\_key] = $7;

}

}

END {

for (transcript\_key in last\_CDS\_end) {

print last\_CDS\_chr[transcript\_key] "\t" (last\_CDS\_end[transcript\_key]-3) "\t" last\_CDS\_end[transcript\_key] "\t" transcript\_key "\t.\t" last\_CDS\_strand[transcript\_key];

}

}' annotation.gtf > stop\_codons.bed

#cleaning of Ensembl CDS fasta file:

grep -v "^>" Mus\_musculus.GRCm39.cds.all.fa | grep -E "[^ACGTacgt]" #check for strange characters

#If NN appears then there is an issue

#How to clean:

awk '/^>/{print (NR==1 ? "" : "\n") $0; next}{printf "%s", $0}END{print "\n"}' Mus\_musculus.GRCm39.cds.all.fa > out.fasta

grep ">" out.fasta

sed 's/N/A/g' out.fasta > out\_clean.fasta #use this file.

\*\*How to run scripts:\*\*

\*\*SCRT:\*\*

python RRS.py

\*\*Stop codon context:\*\*

python stop\_codon\_context\_CDS.py -i “SCRT statistics file from SCRT analysis” -f “CDS fasta” -t “Number of threads” -o “output directory

\*\*Stop codon context statistical analysis:\*\*

python stop\_codon\_context\_stats.py

\*\*Metagene around 3’UTR:\*\*

python Metagene-3UTR.py

\*\*Elongation analysis:\*\*

python elongation\_analysis.py config config.yaml

\*\*Metagene for specific genes CDS (linked to elongation analysis):\*\*

python metagene.py

\*\*Protein prediction for SCRT events:\*\*

python extract\_cds.py transcriptome.fasta gene\_list.txt

\*\*Active ORF analysis (using Ribotoolkit input):\*\*

python ORF.py

python analysis.py -s limma.csv -o outdir/name #for analyzing amino acids sequences and types of ORFs

\*\*Gene-level frameshift analysis:\*\*

python frameshift.py Config.yaml