Data Processing for Blood RNA-seq from DO Mice

- Libraries were prepared by Qiagen in 96-well format, barcoding each sample (including UMI).
- Output fastq files from each of 3 sequencing runs were processed to allow alignment against an 8-way haplotype-specific transcriptome index derived for the Collaborative Cross parents.
- Steps in the data processing pipeline were:
    - Whitelisting cell barcodes allowing distance 1 correction, retaining only known sample barcodes included in each run.
    - Removal of globin transcripts in silico (> 90% total RNA).
    - Genewise deduplication — one read per barcode per UMI per gene (during library prep, reverse-transcribed RNA's are fragmented prior to 5' adapter addition).
    - Splitting deduplicated reads by barcode into individual sample BAM files.
    - bam2fastq conversion and haplotype-specific realignment.

Details are provided in the following slides, using run 2 as the example.
Runs 1 and 3 were processed identically.

UMI-tools

https://github.com/CGATOxford/UMI-tools

GBRS

https://gbrs.readthedocs.io/en/latest/readme.html

Whitelist and Extract Reads

101900-002.R2.fastq.gz

↓ umi-tools whitelist

```
whitelist --bc-pattern=CCCCCCCCCNNNNNNNNNNNNXXX -L 101900-002_whitelist96.log
    --error-correct-threshold=1 --set-cell-number=96
    -I /Volumes/UMass-Projects/DO_RNA-seq/For_Fred/101900-002.R2.fastq.gz -S 101900-002_whitelist96.out
```

↓ Edit whitelist to contain only expected barcodes

whitelist_run2.tsv

↓ umi-tools extract

```
extract --bc-pattern=CCCCCCCCCNNNNNNNNNNNNXXX -L 101900-002_whitelist2_extract.log--extract-method=string
    --read2-in=/Volumes/UMass-Projects/DO_RNA-seq/For_Fred/101900-002.R1.fastq.gz
    --read2-out=101900-002_R1_whitelisted.fastq.gz --white=whitelist_run2.tsv --filter-cell-barcode --error-correct-cell
    -I /Volumes/UMass-Projects/DO_RNA-seq/For_Fred/101900-002.R2.fastq.gz -S 101900-002_R2_whitelisted.fastq.gz
```

↓

101900-002_R1_whitelisted.fastq.gz

101900-002_R2_whitelisted.fastq.gz

Trim Reads (adapters & polyA)

101900-002_R1_whitelisted.fastq.gz

$\downarrow$ cutadapt_plus_pA.sh

```
cutadapt -j 16 -a GATCGGAAGAGCGGTTCAGCAGGAATGCCGAGACCGATCTCGTATGCCGT
    -q 20 -m 25 -o 101900-002_R1_whitelisted.temp.fastq.gz 101900-002_R1_whitelisted.fastq.gz

cutadapt -j 16 -a "A{50}" -m 25 -o 101900-002_R1_pA-trimmed.fastq.gz 101900-002_R1_whitelisted.temp.fastq.gz
```

$\downarrow$

101900-002_R1_pA-trimmed.fastq.gz

Alignment to 8-way Transcriptome of Collaborative Cross Parents

101900-002_R1_pA-trimmed.fastq.gz

bowtie v1.0.0

```
bowtie -n2  --best -S ./bwt_index_gbrs/transcripts
```

101900-002_R1_pA-trimmed_bwt.bam

Preprocess BAM File for Deduplication (preprocess_and_sort.sh)

1. Keep only aligned reads
2. Filter globin reads
3. Replace transcript ID with gene ID in both alignments and header
4. Use transcript maxLength for each gene in header (LN:)
5. Extract barcode tag from read ID and add as SAM tag (BC)
6. Sort by position and index

Preprocess BAM File for Deduplication (preprocess_and_sort.sh)

101900-002_R1_pA-trimmed_bwt.bam

$\downarrow$ samtools v0.1.19

```
samtools view -h -F4 101900-002_R1_pA-trimmed_bwt.bam | perl dedup_preprocess_v2.pl - | \
    samtools view -Sb - > 101900-002_R1_pA-trimmed_bwt_preprocessed.bam

samtools sort -@ 4 101900-002_R1_pA-trimmed_bwt_preprocessed.bam \
    101900-002_R1_pA-trimmed_bwt_preprocessed_sorted

samtools index 101900-002_R1_pA-trimmed_bwt_preprocessed_sorted.bam
```

$\downarrow$

101900-002_R1_pA-trimmed_bwt_preprocessed_sorted.bam

dedup_preprocess_v2.pl

```
# Strip globin reads from BAM
# Replace transcript ID with gene ID in both alignments and header
# Use transcript maxLength in header (LN:)
# Add BC tag
#
# Pipe input and output
# e.g., samtools view -h -F4 file.bam | perl dedup_preprocess_v2.pl - | samtools view -Sb - > file_stripped.bam
#
# Transcripts to be removed:
#       ENSMUST00000093207      Hba-a2
#       ENSMUST00000093209      Hba-a1
#       ENSMUST00000142555      Hba-a1
#       ENSMUST00000023934      Hbb-bs
#       ENSMUST00000131960      Hbb-bs
#       ENSMUST00000147010      Hba-a2
#       ENSMUST00000098192      Hbb-bt
#
# @SQ       SN:ENSMUST00000005218_A      LN:5613 gets changed to
# @SQ       SN:ENSMUSG00000005087 LN:5613 where 5613 is max length

# NB551406:124:HK7YFBGXB:1:13211:11577:7060_ATGTCTTACG_CGCCCAGATGAT 0       ENSMUST00000005218_A   3973 255   97M  * ...  changed to
# NB551406:124:HK7YFBGXB:1:13211:11577:7060_ATGTCTTACG_CGCCCAGATGAT 0       ENSMUSG00000005087     3973 255   97M  * …
#
# Annotation info in GBRS_SQ_transcript_gene_lengths.txt
#
# maxLength_transcript        parent       length       gene
# ENSMUST00000000001    F       3264 ENSMUSG00000000001
# ENSMUST00000000003    A       902  ENSMUSG00000000003
# ENSMUST00000000010    A       2576 ENSMUSG00000020875
[code]
```

Deduplicate Reads

101900-002_R1_pA-trimmed_bwt_preprocessed_sorted.bam

$$\Big\downarrow \text{ umi-tools dedup}$$

```
dedup -L run2_by_sample_dedup_contig.log --per-cell --per-gene --per-contig
    --stdin=101900-002_R1_pA-trimmed_bwt_preprocessed_sorted.bam
    --stdout=101900-002_R1_trimmed_bwt_preprocessed_deduped.bam
```

$$\Big\downarrow$$

101900-002_R1_trimmed_bwt_preprocessed_deduped.bam

Split Reads by Cell Barcode & Convert to fastq

101900-002_R1_trimmed_bwt_preprocessed_deduped.bam

$\downarrow$ split_and_convert_bam-2.sh

```
# sort bam file
samtools sort -@4 -t BC -o 101900-002_R1_trimmed_bwt_preprocess_deduped_BCtag_sorted.bam \
    101900-002_R1_trimmed_bwt_preprocessed_deduped.bam
# split
mkdir -p run2_splits
python split_run2.py > run2_split_info.tsv
# rename
perl rename_bam.pl run2_split_info.tsv  BC_xref-2.tsv
# convert to fastq for gbrs alignment
cd run2_splits
bams=$(ls S*.bam | sort) # S78.bam
for k in ${bams[*]}
do  base=$(basename "$k" .bam) # S78
        bam2fastq -o $base.fastq $k
        gzip $base.fastq
done
```

$\rightarrow$

```
run2_splits/S10.fastq.gz
run2_splits/S109.fastq.gz
.
.
run2_splits/S98.fastq.gz
```

Realign with Bowtie (GBRS Parameters)

```
run2_splits/S10.fastq.gz
run2_splits/S109.fastq.gz
.
.
run2_splits/S98.fastq.gz
```

8-parent transcriptome index (GBRS)

```
cp -r /nl/umw_richard_baker/umi-tools_cluster/run2/bwt_index_gbrs ./
files=$(ls *.fastq.gz) # sample.fastq.gz
for k in ${files[*]}
do  base=$(basename "$k" .fastq.gz)
        zcat $k | bowtie -n2 -a --best --strata -S bwt_index_gbrs/transcripts - \
                | samtools view -Sb - > $base\_bwt.bam
        rm $k
done
rm -r bwt_index_gbrs
```

```
run2_splits/S10_bwt.bam
run2_splits/S109_bwt.bam
.
..
run2_splits/S98_bwt.bam
```

Count Reads per Gene (GBRS)

```
run2_splits/S10_bwt.bam
run2_splits/S109_bwt.bam
.
..
run2_splits/S98_bwt.bam
```

gbrs_quantify.sh

```
bams=$(ls *bwt.bam) # S321_bwt.bam
for k in ${bams[*]}
do  base=$(basename "$k" .bam) # S321_bwt
        name=$(echo $base | cut -d '_' -f1) # S321
        gbrs bam2emase -i $k -m ${GBRS_DATA}/ref.transcripts.info \
                -s A,B,C,D,E,F,G,H -o $name.emase
        gbrs compress -i $name.emase -o $name.cemase
        rm $k
        rm $name.emase
        gbrs quantify -i $name.cemase -g ${GBRS_DATA}/ref.gene2transcripts.tsv \
                -L ${GBRS_DATA}/gbrs.hybridized.targets.info -M 4 -o $name
        rm *multiway.isoforms*
done
```

```
run2_splits/S10.multiway.genes.expected_read_counts
run2_splits/S109.multiway.genes.expected_read_counts
.
.
run2_splits/S98.multiway.genes.expected_read_counts
```

Collect Results from All Samples

```
run2_splits/S10.multiway.genes.expected_read_counts
run2_splits/S109.multiway.genes.expected_read_counts
.
run2_splits/S98.multiway.genes.expected_read_counts
```

$\downarrow$ make_count_matrix2.R

```
names <- unlist(read.delim("run2_samples.txt", stringsAsFactors=F, header=F))
counts <- list()
for (i in names) {
  this.filename <- paste0(i, ".multiway.genes.expected_read_counts")
  counts[[i]] <- read.table(this.filename, row.names=1, stringsAsFactors=F)[, 9]
}
count.matrix <- as.data.frame(do.call(cbind, counts))
colnames(count.matrix) <- names
# get row names
df <- read.table(this.filename, row.names=1, stringsAsFactors=F)[, 1:2]
row.names(count.matrix) <- row.names(df)
# remove 'total' line
count.matrix <- count.matrix[-1, ]
write.csv(count.matrix, "run2_byGene_count_matrix.csv")
```

$\downarrow$

run2_byGene_count_matrix.csv