

PHS 675

Special Topics: Analyzing RNA Sequencing Data

Homework 4

Model Solution

1. Create a variable that hold the samples

```
samples=$( ls raw_fastqs | cut -d '_' -f 1 | sort -u )
```

2. Iterate through the list of samples and run hisat2 on each sample.

Pipe the results through samtools view to create a bam file.

Submit these calls to the cluster using srun.

Note: the cpus-per-task will ensure no more than six jobs on each node,

which can help prevent out-of-memory crashes.

```
for s in ${samples} ; do
```

```
    srun -p normal --cpus-per-task=8 \  
        hisat2 \  
        --no-mixed \  
        --no-discordant \  
        -x Genome/Mus_musculus.GRCm39 \  
        -1 trimmed/${s}_R1.fastq.gz \  
        -2 trimmed/${s}_R2.fastq.gz \  
        2>logs/${s}_align.log \  
        | samtools view -o aligned/${s}.bam &
```

```
done
```

```
wait
```

3. Iterate through all the samples and call samtools sort on each sample,
sorting by name and storing the results in a folder called name_sorted.

```
for s in ${samples}; do
    srun -p normal --cpus-per-task=8 \
        samtools sort -n -o name_sorted/${s}.bam aligned/${s}.bam \
        2>logs/${s}_sort.bam &
done

wait
```

4. Iterate through all the samples and call htseq-count on each of the
sorted bam files, and put the output in a folder called counts.

```
for s in ${samples}; do
    srun -p normal \
        htseq-count \
        --stranded=no \
        --order=name \
        --mode=union \
        --nonunique=none \
        name_sorted/${s}.bam \
        Genome/Mus_musculus.GRCm39.108.gtf \
        >counts/${s}.txt \
        2>logs/${s}_count.log &
done
```

Step 1 uses ls to get a list of the raw fastq files, pipes that to cut to select the part before the first underscore, and then sorts them and gets unique results (this is necessary because there are two fastq files per sample, and we want each sample name only once).

Step 2 iterates through the list of samples obtained from step 1, putting each sample name in turn into a variable called s. For each sample, it invokes hisat2 with the no-mixed and no-

discordant options, which only count paired reads if the reads align to the same region (no-mixed) and in the expected directions (no-discordant). The -x option specifies the location of the hisat2 index of the genome, the -1 and -2 options specify the forward and reverse read files, respectively, which are the results of running trimmomatic on the raw reads. The 2> option sends the standard error output (which has logging information) into a file in the logs file. The standard output (which contains the output in SAM format) is piped to samtools view, which in turn outputs to a bam file. This has the effect of converting the SAM output to BAM output, which conserves disk space.

Step 3 again iterates through the samples, and for each sample passes the output file from step 2 to samtools sort. The -n option specifies to sort by name, and the -o option specifies to place the output in a file in the name_sorted directory.

Step 4 iterates through the samples one more time, calling htseq-count on each sample. It specifies that the libraries were not built using a stranded protocol, that the reads are ordered by name, and then explicitly sets the mode to the default ("union") and that no non-unique reads should be counted. The input is the sorted bam file from step 3. Next, the GTF file containing the locations of exons, transcripts, and genes on the mouse genome is specified. The resulting table is sent by htseq-count to standard output, which is redirected to a file in the counts folder, and the standard error (which contains logging and diagnostic information) is sent to a file in the logs folder.