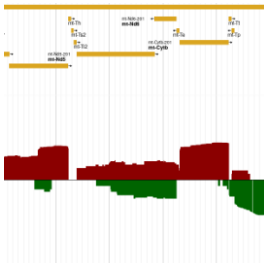
Apr 28, 2019 Version 4

Stranded Mapping from Oriented Long Reads V.4



DOI
dx.doi.org/10.17504/protocols.io.z9zf976

David A Eccles[1]

[1]Malaghan Institute of Medical Research (NZ)

David A Eccles
Malaghan Institute of Medical Research (NZ)

OPEN ACCESS

DOI: dx.doi.org/10.17504/protocols.io.z9zf976

External link: https://bioinformatics.stackexchange.com/a/3922/73

Protocol Citation: David A Eccles 2019. Stranded Mapping from Oriented Long Reads. **protocols.io** https://dx.doi.org/10.17504/protocols.io.z9zf976

Protocol status: In development
We are still developing and optimizing this protocol

Created: April 23, 2019

Last Modified: April 28, 2019

Protocol Integer ID: 22553

Abstract

This protocol demonstrates how to map strand-oriented long reads to a genome, and visualise them in a genome browser.

The general idea is to use minimap2 to create stranded BAM files, which are split for forward/reverse orientation then converted into BigWig format for display in a genome browser.

Input(s):
- stranded fastq files (see protocol **Preparing Reads for Stranded Mapping**)
- a FASTA file containing the genome / sequence of interest.

Output(s):
- Genome-mapped stranded BAM files
- Genome-mapped stranded BigWig files

Before start

You will need access to the following free and open-source software program(s):
- **minimap2**
- **samtools**

And the following additional data file(s):
- a FASTA file containing the genome / sequence of interest.

Orient Reads

1 Orient reads as per protocol **Preparing Reads for Stranded Mapping**.

If this has been done, then the following command should produce output without errors:

```
for bc in $(awk '{print $2}' barcode_counts.txt);
  do ls oriented/${bc}_reads_dirAdjusted.fastq.gz;
done
```

Example output:

```
oriented/BC03_reads_dirAdjusted.fastq.gz
oriented/BC04_reads_dirAdjusted.fastq.gz
oriented/BC05_reads_dirAdjusted.fastq.gz
oriented/BC06_reads_dirAdjusted.fastq.gz
oriented/BC07_reads_dirAdjusted.fastq.gz
oriented/BC08_reads_dirAdjusted.fastq.gz
```

Index Preparation

2 Prepare genome index for spliced alignment

Software	
minimap2	NAME
Linux	OS
Heng Li	DEVELOPER
https://github.com/lh3/minimap2	SOURCE LINK

```
minimap2 -d mmus_ucsc_all-splice.idx -Q -t 10 -x splice
mmus_ucsc_all.fa
```

Read Mapping

3 Map the long reads to the genome using minimap2, using samtools to covert to a sorted BAM format. This is where the reverse complementing done during demultiplexing gives a big saving of effort.

Software

SAMtools	NAME
Linux	OS
Wellcome Trust Sanger Institute	DEVELOPER
https://github.com/samtools/samtools	SOURCE LINK

```
mkdir -p mapped;
for bc in $(awk '{print $2}' barcode_counts.txt);
  do echo ${bc};
  minimap2 -t 10 -a -x splice mmus_ucsc_all-splice.idx
oriented/${bc}_reads_dirAdjusted.fastq.gz | \
    samtools view -b | samtools sort >
mapped/mm2_called_${bc}_vs_MmusG.bam;
done
```

Creating BigWig Coverage Files

4 mpileupDC.pl

A bedGraph of coverage is created using samtools mpileup and **mpileupDC.pl**, excluding any skipped intronic sequence. When 'mpileupDC.pl' is provided with a single file, it will output a bedGraph file with a header line starting with '##'; this header line is removed. The particular JBrowse plugin that I use for stranded display requires that the reverse strand have *negative* coverage values, so that file needs to be changed:

```
for bc in $(awk '{print $2}' barcode_counts.txt);
  do echo ${bc};
  samtools view -b -F 0x10 mapped/mm2_called_${bc}_vs_MmusG.bam |
\
    samtools mpileup -A -B -Q 0 -q 0 -I -q 0 -Q 0 - | \
    mpileupDC.pl | tail -n +2 >
mapped/mm2_called_${bc}_vs_MmusG.bg.plus
  samtools view -b -f 0x10 mapped/mm2_called_${bc}_vs_MmusG.bam |
\
    samtools mpileup -A -B -Q 0 -q 0 -I -q 0 -Q 0 - | \
    mpileupDC.pl | tail -n +2 >
mapped/mm2_called_${bc}_vs_MmusG.bg.minus
  perl -i -pe 's/([0-9]+)$/-$1/'
mapped/mm2_called_${bc}_vs_MmusG.bg.minus
done;
```

5 Stranded bedgraph files are converted to bigwig. This requires BEDTools and a genome information file containing chromosome lengths (one for Mmus/mm10 is attached to this step).

Software

NAME	DEVELOPER	SOURCE LINK
BEDTools	Quinlan laboratory, University of Utah	https://github.com/arq5x/bedtools2/

```
for bc in $(awk '{print $2}' barcode_counts.txt);
  do echo ${bc};
  basename="mapped/mm2_called_${bc}_vs_MmusG"
  bedGraphToBigWig ${basename}.bg.plus Mmus_genome.chrInfo.txt
${basename}.bw.plus
  bedGraphToBigWig ${basename}.bg.minus Mmus_genome.chrInfo.txt
${basename}.bw.minus
done
```

Mmus_genome.chrInfo.txt

JBrowse Configuration

6 Each track should have its own JBrowse configuration section using the *StrangedBigWig* class and *StrandedXYPlot* type. An example is shown here:

```
[tracks.BWCG004-4T1-BC04-both-track ]
 storeClass     =
StrandedPlotPlugin/Store/SeqFeature/StrandedBigWig
 urlTemplate    = bw/mm2_called_CG004_BC04_vs_MmusG.bw
 category       = MinION - Coverage
 type           =
StrandedPlotPlugin/View/Track/Wiggle/StrandedXYPlot
 key            = MinION minimap2 coverage from CG004-4T1-WT
(combined strands)
 scale          = log
 scoreType      = maxScore
 autoscale      = global
 style.pos_color = darkred
 style.neg_color = darkgreen
```

Sanity Check

7 If this has worked properly, then mapping human or mouse to the mitochondrial genome should show most expression appearing on the positive strand, with a small scattering of negative-strand expression, a bit like the *Expected Results* shown here.

If not, check for the following issues:

- Tracks not displaying at all in JBrowse -- make sure track IDs inside square brackets are of the form [*tracks.<unique-id-without-dots>-track*]
- JBrowse track is reflected in the X axis -- make sure that the reverse bedgraph file is orientated the correct way; it should be created with the '-f 0x10' flag (no capitalisation).
- JBrowse track only shows one direction -- make sure that the reverse bedgraph file has *negative* values, and re-generate the bigwig file

Expected result