# isoform: A set of tools for RNA isoform study using RNA-seq data.

Wei Sun

February 22, 2013

## 1 Overview

```
> library(isoform)
```

This vignette describes how to use `R/isoform` to estimate isoform abundance and assess differential isoform usage between cases and controls using an example. We are working on applying this method to assess differential isoform usage with respect to continuous covariate, e.g., isoform-specific eQTL mapping. Such examples will be added soon.

## 2 A brief Introduction

RNA-seq is replacing gene expression microarrays to be the major experimental technique for genome-wide assessment of transcript abundance. One of the most important novelties of RNA-seq is that it provides much more informative data to study RNA isoform expression. This vignette presents a working example to estimate isoform abundance and to assess differential isoform usage of a gene by R package `isoform`, which can be downloaded from `http://www.bios.unc.edu/~weisun/software/isoform_files/`. The details of the statistical methods used in this R package are beyond the scope of this paper and they are presented elsewhere.

## 3 Download simulated data

We used a set of simulated paired-end RNA-seq reads (76bp+76bp) for this working example. The reads were simulated by Flux Simulator (`http://flux.sammeth.net/simulator.html`) based on the transcriptome annotation of chromosome 1 and chromosome 18 of mouse genome (`Mus_musculus.NCBIM37.67.gtf` from Ensembl database). We specifically set up the simulation such that all the genes from chromosome 1 are equivalently expressed between the two samples (`mm9_simu_set1` vs. `mm9_simu_set2`, or case vs. control), and all the genes from chromosome 18 are differentially expressed between the two samples.

The bam files and some intermediate files for two samples (`mm9_simu_set1` and `mm9_simu_set2`) can be downloaded from the following link: `http://www.bios.unc.edu/~weisun/software/isoform_files/mm9_simulated_data.zip`. Each sample has a similar set of files. Here we only list the files for sample `mm9_simu_set1`. Similar explanations apply to the files for sample `mm9_simu_set2`.

- `mm9_simu_set1_paired_reads_sorted_by_name.bam`: a bam file that includes about 1 million mapped paired-end reads from mouse chromosome 1 or 18. All the reads in this file are sorted by read name, such that the two ends of a paired-end read is next to each other, if they are both mapped. This particular order of reads is required for read counting in the next step.

The following four files will be created in later steps of this working example.

- `mm9_simu_set1_insertSize_dist.txt`: A text file of the insert size distribution. Insert size is the same as fragment length. For example, an RNA-seq fragment of 300bp may be sequenced 76bp in both ends, then the insert size is 300bp, instead of 300-76$\times$2 = 148bp.

- `mm9_simu_set1_counts_paired_reads.txt`: read counts per exon set.
- `mm9_simu_set1_geneModel_knownIsoforms.RData`: selected isoforms and their expression abundance for each gene (or transcript cluster) using known isoform information.
- `mm9_simu_set1_geneModel.RData` selected isoforms and their expression abundance for each gene (or transcript cluster) when no existing isoform information is available.

# 4 Start from .bam file – count the number of reads

Our method does not require known isoform annotation. However we do require exon annotation. We count the number of reads per exon set, which includes one or more exons. A single or paired-end read overlaps with an exon set if it overlaps with all the exons in this exon set and does not overlap with any other exon. In the existing annotations, the exons are often overlapping. We produce a set of non-overlapping exons for mouse (`Mus_musculus.NCBIM37.67_data.zip`) and human (`Homo_sapiens.GRCh37.66_data.zip`), which can be downloaded from `http://www.bios.unc.edu/~weisun/software/isoform_files/`. The scripts used to generate (and check) these non-overlapping exons can be also downloaded in the above link.

All the reads the input bam file must be sorted by read name before counting. The following code perform sorting and some basic QC using function prepareBAM from R package `asSeq`.

```
cmd2 = sprintf("samtools sort -n %s %s_sorted_by_name", bami, sami)
system(cmd2)
bamF = sprintf("%s_sorted_by_name.bam", sami)

prepareBAM(bamF, sprintf("%s_sorted_by_name", sami), sortIt=FALSE)
```

The following set of codes can be used to count the number of reads per exon set.

```
library(isoform)

bedFile = "Mus_musculus.NCBIM37.67.nonoverlap.exon.bed"
bamFile = "mm9_simu_set1_paired_reads_sorted_by_name.bam"
ctFile  = "mm9_simu_set1_counts_paired_reads.txt"

countReads(bamFile, bedFile, ctFile)

bamFile = "mm9_simu_set2_paired_reads_sorted_by_name.bam"
ctFile  = "mm9_simu_set2_counts_paired_reads.txt"

countReads(bamFile, bedFile, ctFile)
```

A few lines of file `mm9_simu_set1_counts_paired_reads.txt` are shown below.

```
37 chr18_109|ENSMUSG00000024491|4;chr18_109|ENSMUSG00000024491|5;
17 chr18_109|ENSMUSG00000024491|5;
88 chr18_109|ENSMUSG00000024491|5;chr18_109|ENSMUSG00000024491|6;
```

The first column corresponds to read count and the second column lists exon sets, where "chr18_109" indicates a transcript cluster, "ENSMUSG00000024491" is ensemble gene ID, and the numbers at the end is exon ID. We organize the annotation into transcript clusters because the genomic positions of two genes may overlap, then we need to consider these two genes together in isoform studies.

# 5 Select RNA-isoforms and estimate isoform level expression

Not all the RNA-isoforms of a gene are expressed. Even if RNA isoform annotations are available, it is very likely that only some of the annotated RNA-isoforms are expressed. Therefore, select the expressed RNA isoforms and estimate their expression abundance are two connected problems. We achieve these two goals using one function `isoDetector`:

```
countFile    = "mm9_simu_set1_counts_paired_reads.txt"
fragSizeFile = "mm9_simu_set1_insertSize_dist.txt"
output       = "mm9_simu_set1_geneModel_knownIsoforms.RData"
bedFile      = "Mus_musculus.NCBIM37.67.nonoverlap.exon.bed"
iso          = "Mus_musculus.NCBIM37.67.nonoverlap.exon.knownIsoforms.RData"
readLen      = 76

# only select isoforms from known isoforms
isoDetector(countFile, bedFile, fragSizeFile, readLen, output, knownIsoforms=iso)

# select isoforms from all possible isoforms
isoDetector(countFile, bedFile, fragSizeFile, readLen=76, output)
```

The fragSizeFile ”mm9_simu_set1_insertSize_dist.txt” can be created using the following command. A separate fragSizeFile should be created for each sample.

```
samtools view -f 73 mm9_simu_set1_paired_reads_sorted_by_name.bam \
| awk '{ print ($8 >= $4) ? $8-$4+76 : $4-$8+76 }' \
| sort -n | uniq -c > mm9_simu_set1_insertSize_dist.txt
```

where 76 in the code (8−4+76 : 4−8+76) is the read length. The output of each transcript cluster is a list. While a complete description of the elements of this list can be found in the help file of function `isoDetector`, two most important elements are `w2kp`, which indices the isoforms selected by penalized regression, and `abundance`, which are the estimated number of fragments of each selected isoform. In this testing example, the function `isoDetector` took about 2 hours to finish.

# 6 Test differential isoform usage

After running function `isoDetector` for samples `mm9_simu_set1` and `mm9_simu_set2`, we are ready to test for differential isoform usage. This step is time consuming because we use resampling method to calculate p-values. Parallel computation is needed to run a small number of transcript clusters for each CPU. For example, the following code tests for differential expression and/or differential isoform usage for 10 transcript clusters.

```
idx1  = 1
idx2  = 10
tags  = c("set1", "set2")
Routs = paste("mm9_simu_",tags,"_geneModel_knownIsoforms.RData", sep="")
fragSizeFiles = paste("mm9_simu_",tags,"_insertSize_dist.txt", sep="")

# xData is a case/control indicator of the two samples
xData = c(0, 1)

# ------------------------------------------------------------------------
# select the transcript clusters to be used
# ------------------------------------------------------------------------

load(Routs[1])
```

```
ct1 = sapply(geneMod, function(x){ sum(x$y) } )
length(ct1)
summary(ct1)

load(Routs[2])
ct2 = sapply(geneMod, function(x){ sum(x$y) } )
length(ct2)
summary(ct2)

g2test = intersect(names(ct1), names(ct2))
length(g2test)

# ---------------------------------------------------------------------------
# the cases where there are reads from one sample but not the other sample
# are dropped from testing, though they may be very intersting results.
# ---------------------------------------------------------------------------

g2drop = setdiff(union(names(ct1), names(ct2)), g2test)
c2drop = data.frame(tcluster=g2drop, ct1=ct1[match(g2drop,names(ct1))],
  ct2=ct2[match(g2drop,names(ct2))])

c2drop

# ---------------------------------------------------------------------------
# test differential expression and differential isoform usage
# ---------------------------------------------------------------------------

outputFileName = "set1_vs_set2_knownIsoforms"
outputFileName = sprintf("%s_%d_%d.txt", outputFileName, idx1, idx2)

isoDu(tags, Routs, xData, outputFileName, fragSizeFiles, g2test=g2test,
readLen=76, method="bootstrap",  lmax=500, idxs=idx1:idx2, maxTime=43200)

# ---------------------------------------------------------------------------
# test differential isoform usage only
# ---------------------------------------------------------------------------

outputFileName = "set1_vs_set2_knownIsoforms_duOnly"
outputFileName = sprintf("%s_%d_%d.txt", outputFileName, idx1, idx2)

isoDu(tags, Routs, xData, outputFileName, fragSizeFiles, g2test=g2test,
readLen=76, method="bootstrap",  lmax=500, idxs=idx1:idx2, duOnly=TRUE,
maxTime=43200)
```

The above code tests differential expression and/or differential isoform usage using the results when isoform annotation is provided. Similar test can be applied to the results when isoform annotation is not used (i.e., *de novo* isoform identification). Users should pay special attention to the parameter `method` in R function `isoDu`. When sample size is small, for example, here we compare one case vs. one control, `method` should be set as "bootsrap", and the testing p-value only indicates the difference of the studied samples, but cannot be generalized to other samples. When sample size is relatively large, for example, 5 cases vs. 5 controls, the user should set the the parameter `method` to be "permutation" and the obtained p-values imply the difference between the case and control population, beyond the particular samples used in the study.