Reciprocal Cross in RNA-seq

Vasyl Zhabotynsky [*] Wei Sun Fei Zou

December 1, 2016

1 Overview

This vignette describes how to use `R/rxSeq` to perform an analysis on RNA-seq data from F1 reciprocal crosses.

```
> library(rxSeq)
```

2 Introduction

RNA sequencing (RNA-seq) not only measures total gene expression but may also measure allele-specific gene expression in diploid individuals. RNA-seq data collected from F1 reciprocal crosses (and respective inbred) in mouse can powerfully dissect strain and parent-of-origin effects on allelic imbalance of gene expression. This R package, rxSeq, implements a novel statistical approach for RNA-seq data from F1 and inbred strains. Zou *et al.* (2014) [[5]]
The package allows to fit the joint model of the total read counts for each mouse (assuming Negatvie-Binomial model to allow for an overdispersion) and allele specific counts (Beta-Binomial model). In the provided data example these counts are aggregated on gene level, though as long as counts are not too small, any level of generalization can be used: transcript level, exon level, etc.

3 Citing `R/rxSeq`

When using the results from the `R/rxSeq` package, please cite:

> Zou, Fei, Wei Sun, James J. Crowley, Vasyl Zhabotynsky, Patrick F. Sullivan, and Fernando Pardo-Manuel de Villena (2014), A novel statistical approach for jointly analyzing RNA-Seq data from F1 reciprocal crosses and inbred lines. *Genetics* **197**(1):389-399

The article describes the methodological framework behind the `R/rxSeq` package.

[*] vasyl@unc.edu

4 rxSeq implementation and output

4.1 Fitting the data

4.1.1 Joint model (TReCASE model) for total read counts (TReC) and allele specific expression (ASE) counts

The package `R/rxSeq` was developed for analyzing RNA-seq data from F1 reciprocal mouse crosses (2013)[[3]] based on development of Collaborative Cross mouse model [[2]]. The model aimes to combine the total read counts (TReC) and allele specific expression (ASE) counts, estimate simultaneously additive strain effect, parent of origin effect, as well as taking in account dominance effect, sex effect and adjust for individual total level of expression of a mouse. At the same time, model allows to reduce type II error by estimating overdispersion of the count data. One of the packages allowing to produce TReC as well as ASE data is the R package `R/asSeq`[[1]] developed by our group or using Genomic Alignment [[4]]. A detailed pipeline of producing gene-level (or transcript-level) allele specific counts can be found in the asSeq document, the general idea is to find the reads which have SNP and indel information using which reads can be classified as allele specific (ASE), as well as count number of reads that overlapped a particular part of a genome - gene level TReC counts.

First lets create an input data object for autosomal and X-chromosome genes

```
> rcA = readCounts(index=data.A$index, y=data.A$y[1:2,], n=data.A$n[1:2,],
+         nOB=data.A$nOB[1:2,], kappas=data.A$kappas, geneids=data.A$geneids[1:2])
> rcX = readCounts(index=data.X$index, y=data.X$y[1:2,], n=data.X$n[1:2,],
+         nOB=data.X$nOB[1:2,], kappas=data.X$kappas, geneids=data.X$geneids[1:2],
+         tausB=data.X$tausB, chrom="X")
```

For autosomal genes, the full TReCASE model can be fitted as:

```
> #fit trecase autosome genes:
> trecase.A.out = process(rcA)
```

Note, that it requires both TReC and ASE counts, and assumes that mice and genes match in the data matrices.

Since the X chromosome is different, in that it has *Xce* effect - proportion of one of the allele expression is not 0.5, but is scewed for the whole chromosome, we need to adjust for it in order to estimate effects correctly. The following command runs the TReCASE model for chromsome X genes, which requires two additional parameters for dealing with X-chromosome inactivations: *Xce* effect - proportion of the reads coming from the allele B for the whole X chromosome and which genes have this proportion switched (the well known example is *Xist* gene (it's Ensembl id is provided by default - ENSMUSG00000086503)

```
> #fit trecase X chromosome genes:
> trecase.X.out = process(rcX)
```

These functions return the following outputs: parameter estimates from the full models and associated p-values, and all reduced short models, followed by the list of errors:

```
> names(trecase.A.out)

 [1] "pvals"        "coef.full"    "coef.add"     "coef.poo"     "coef.dom"
 [6] "coef.same"    "coef.ase.add" "coef.sex"     "coef.sex.add" "coef.sex.poo"
[11] "coef.sex.dom" "errorlist"

> trecase.A.out$pval[,1:2]

                       pval_add  pval_poo
ENSMUSG00000055725 7.224770e-02 0.5868794
ENSMUSG00000015568 1.515202e-25 0.5460404

> names(trecase.X.out)

 [1] "pvals"        "coef.full"    "coef.add"     "coef.poo"     "coef.dom"
 [6] "coef.same"    "coef.ase.add" "coef.sex"     "coef.sex.add" "coef.dev.dom"
[11] "errorlist"

> trecase.X.out$pval[,1:2]

                       pval_add  pval_poo
ENSMUSG00000086503 1.009569e-02 0.5095113
ENSMUSG00000049775 6.992049e-05 0.9503648
```

We can recalculate negative log likelihood with optional estimation of hessian:

```
> nLogLik(res=trecase.A.out, rc=rcA, genei=1)$nll

[1] 261.349

> nLogLik(res=trecase.X.out, rc=rcX, genei=1)$nll

[1] 247.1559
```

4.1.2 TReC model for TReC only

The package also allows to fit the data with only TReC when for a given gene, if there is no enough SNP or indel information for estimating ASE.

The following function fits the TReC model for autosomal genes:

```
> #fit trec autosome genes
> rcA$model = "short"
> trec.A.out = process(rcA)
> names(trec.A.out)

 [1] "pvals"        "coef.full"    "coef.add"     "coef.poo"     "coef.dom"
 [6] "coef.sex"     "coef.sex.add" "coef.sex.poo" "coef.sex.dom" "errorlist"

> trec.A.out$pval[,1:2]
```

```
                       pval_add  pval_poo
ENSMUSG00000055725 2.332545e-02 0.5853869
ENSMUSG00000015568 5.106522e-09 0.8125544
```

Again, the separate treatment of X chromosome is implemented. The following function fits the TReC model for chromosome X genes:

```
> #fit trec X chromosome genes
> rcX$model = "short"
> trec.X.out = process(rcX)
> names(trec.X.out)
```

```
[1] "pvals"        "coef.full"    "coef.add"     "coef.poo"     "coef.dom"
[6] "coef.sex"     "coef.sex.add" "coef.dev.dom" "errorlist"
```

```
> trec.X.out$pval[,1:2]
```

```
                     pval_add  pval_poo
ENSMUSG00000086503 0.29390171 0.3133116
ENSMUSG00000049775 0.06834156 0.3018140
```

Similarly for short model we also can recalculate negative log likelihood with optional estimation of hessian:

```
> nLogLik(res=trec.A.out, rc=rcA, genei=1)$nll
```

```
[1] 183.4093
```

```
> nLogLik(res=trec.X.out, rc=rcX, genei=1)$nll
```

```
[1] 204.7115
```

4.2 Estimating *Xce* effect for X chromosome

Both proc.trecase.X and proc.trec.X require an estimate of the *Xce* effect. In the above examples, we used an estimated value from the data in Crowley at al. (2013) [[3]].

The following function estimates the *Xce* effect for any given data:

```
> get.tausB(n=data.X$n, nOB=data.X$nOB, geneids=data.X$geneids,
+           Xist.ID="ENSMUSG00000086503")
```

```
           FG_0125_F_hapG FG_0162_F_hapG FG_0163_F_hapG FG_0164_F_hapG
med.tauB        0.2266945      0.2512354      0.2888816      0.2984825
ave.tauB        0.2338611      0.2511211      0.2864014      0.2961086
all.genes       8.0000000      8.0000000      8.0000000      8.0000000
used.genes      8.0000000      8.0000000      8.0000000      8.0000000
           FG_0167_F_hapG FG_0168_F_hapG GF_0164_F_hapG GF_0165_F_hapG
med.tauB        0.2381954      0.2433292      0.3331889      0.2372911
ave.tauB        0.2354252      0.2525122      0.3477522      0.2317843
```

```
all.genes        8.0000000      8.0000000      8.0000000      8.0000000
used.genes       8.0000000      8.0000000      8.0000000      8.0000000
           GF_0166_F_hapG GF_0168_F_hapG GF_0238_F_hapG
med.tauB         0.2395825      0.3396480      0.3413311
ave.tauB         0.2529066      0.3592291      0.3367434
all.genes        8.0000000      8.0000000      8.0000000
used.genes       8.0000000      8.0000000      8.0000000
```

For genes that are known to escape X inactivation or have different *Xce* control effects, adjusted analysis can be done provided their ids are given. A default gene - *Xist* which is known to have an opposite inactivation pattern with the other X chromosome genes, we set its estimate to $1 - Xce$. We may also exclude genes with too low ASE (which is set to 50 by default) and/or with too low proportion of one of the alleles. The default value for the latter is set to 0.05 to avoid fully imprinted genes.
In order to reliently estimate Xce effect we need to have allele specific data, so if you don't have such information, you may use literature average, and if those are not available, you still can fit the model assuming proportion is 0.5, understanding that it may bias the inference.

```
> data.X$tausB

FG_0125_F_hapG FG_0162_F_hapG FG_0163_F_hapG FG_0164_F_hapG FG_0167_F_hapG
     0.2346327      0.2520325      0.3043478      0.3000000      0.2555848
FG_0168_F_hapG GF_0164_F_hapG GF_0165_F_hapG GF_0166_F_hapG GF_0168_F_hapG
     0.2645804      0.3488372      0.2486188      0.2712934      0.3781178
GF_0238_F_hapG
     0.3611111
```

The first row of the **get.tausB** output provides a medain estimate of the *Xce* effect and the second row provides an average estimate of *Xce* effect. The two estimates are expected to be close, though median would be more stable.

```
> get.tausB(n=data.X$n,nOB=data.X$nOB,geneids=data.X$geneids,Xist.ID = "")

           FG_0125_F_hapG FG_0162_F_hapG FG_0163_F_hapG FG_0164_F_hapG
med.tauB         0.2303523      0.2534435      0.2897196      0.2986111
ave.tauB         0.3733453      0.3372797      0.3296875      0.3400289
all.genes        9.0000000      9.0000000      9.0000000      9.0000000
used.genes       9.0000000      9.0000000      9.0000000      9.0000000
           FG_0167_F_hapG FG_0168_F_hapG GF_0164_F_hapG GF_0165_F_hapG
med.tauB         0.2466844      0.2501718      0.3350168      0.2475884
ave.tauB         0.3291692      0.3398780      0.4076566      0.3148905
all.genes        9.0000000      9.0000000      9.0000000      9.0000000
used.genes       9.0000000      9.0000000      9.0000000      9.0000000
           GF_0166_F_hapG GF_0168_F_hapG GF_0238_F_hapG
med.tauB         0.2437753      0.3507246      0.3437500
ave.tauB         0.3357056      0.4229374      0.3998674
all.genes        9.0000000      9.0000000      9.0000000
used.genes       9.0000000      9.0000000      9.0000000
```

5 Simulations

The TReC and ASE counts can be simulated using function **simRX** which requires the following input variables:

```
> dat.A = simRX(b0f=.5, b0m=.6, b1f=.3, b1m=.4, beta_sex=.1, beta_dom=.1, n.simu=1E1)
> names(dat.A)

[1] "index"   "y"       "n"       "n0B"     "kappas"  "geneids"

> dat.X = simRX(b0f=.5, b0m=.6, b1f=.3, b1m=.4, beta_sex=.1, beta_dom=.1, n.simu=1E1,
+ is.X=TRUE, tauB=.3)
> names(dat.X)

[1] "index"   "y"       "n"       "n0B"     "kappas"  "tausB"   "geneids"
```

It produces three data matrices: TReC - **y**, all ASE counts - **n** and ASE counts for allele B - **n0B**, as well as specifying which mouse belongs to which cross, and what is the overall expression level for each mouse. These simulations provide all the required input for the fitting functions.

6 Function name specification

The essential functions for the package have names reiterating their purpuse:
The first part of the name is:

1. (proc) - process the data, fit the likehood and test the hypotheses of interest and (optionally calculate Hessian matrix at the MLE)
2. (nLogLik) - calculate negative log-likelihood at a given point (and, optionally, calculate Hessian matrix)

the second part is specifies if it is a full or a short model:

1. (trecase) - a full model (using TReC and ASE counts)
2. (trec) - a short model (using TReC only counts)

and the last part is specifying gene from which chromosome the model will fit:

1. (A) - an autosome
2. (X) - X chromosome

7 References

References

[1] Wei Sun, Vasyl Zhabotynsky (2013) asSeq: A set of tools for the study of allele-specific RNA-seq data. *http://www.bios.unc.edu/ weisun/software/asSeq.pdf.*

[2] Collaborative Cross Consortium (2012), The Genome artichecture of the Collaborative Cross Mouse Genetic Reference Population., *Genetics.* **190**(2):389-401

[3] Crowley, J. J., Zhabotynsky, V., Sun, W., Huang, S., Pakatci, I. K., Kim, Y., Wang, J. R., Morgan, A., P., Calaway, J. D., Aylor, D. L., Yun, Z., Bell, T. A., Buus, R. J., Calaway, M. E., Didion, J. P., Gooch, T. J., Hansen, S. D., Robinson, N. N., Shaw, G. D., Spence, J. S., Quackenbush, C. R., Barrick, C. J., Xie, Y., Valdar, W., Lenarcic, A. B., Wang, W., Welsh, C. E., Fu, C. P., Zhang, Z., Holt, J., Guo, Z., Threadgill, D. W., Tarantino, L. M., Miller, D., R., Zou, F., McMillan, L., Sullivan, P. F., and Pardo-Manuel de Villena, F., (2015), Analyses of allele-specific gene expression in highly divergent mouse crosses identifies pervasive allelic imbalance., *Nature genetics*, **47**(4):353-360

[4] Lawrence, M., Huber, W., Pages, H., Aboyoun, P., Carlson, M., Gentleman, R., Morgan, M.T. and Carey, V.J. (2013), Software for computing and annotating genomic ranges. *PLoS Comput Biol*, **9**(8):p.e1003118

[5] Zou, Fei, Wei Sun, James J. Crowley, Vasyl Zhabotynsky, Patrick F. Sullivan, and Fernando Pardo-Manuel de Villena (2014), A novel statistical approach for jointly analyzing RNA-Seq data from F1 reciprocal crosses and inbred lines. *Genetics* **197**(1):389-399