ExplorATE vignette

Martin Femenias

29/12/2021

#ExporATE vignette

This vignette implements the ExplorATE package on different data sets, both for model and non-model organisms. Further, this vignette helps users to obtain files necessary for the execution of ExplorATE, such as gene models and reference genomes for model organisms. User can find detailed information on each function in the users' guide.

1 Installation

To install the ExplorATE shell script, make sure you meet the requirements detailed on the program page and type the following command at the console:

```
git clone https://github.com/FemeniasM/ExplorATE_shell_script
```

The script can be run from the folder created in your directory:

```
cd ExplorATE_shell_script
```

To run the program, the user must define the mode and the flags corresponding to each mode:

```
bash ExplorATE [mode][flags]
```

To install the R package ExplorATE, make sure you meet the requirements detailed on the program page and type the following command at the R console:

```
devtools::install_github("FemeniasM/ExplorATEproject")
```

More information about the installation can be found in the user guide.

2 Obtaining the data set

ExplorATE includes a test data set for model and non-model organisms. Users can create a folder `ExplorATE_data_test` containing the data sets by typing in the console:

```
git clone https://github.com/FemeniasM/ExplorATE_data_test
```

Inside the `ExplorATE_data_test` folder three subdirectories are created: `inputs_hs`, `inputs_dm` and `inputs_lp` that contain the data for humans, *Drosophila melanogaster* and *Lioleamus parthenos* respectively. Each of the data sets are described throughout the vignette.

3. Analysis with model organisms

This vignette proposes to first analyze a set of "toy data" based on the human genome and then to analyze real data on Drosophila melanogaster. The data set based on the human genome allows users to become familiar with the environment while performing a quick execution of the program. Also, it allows users to verify if the installation of ExplorATE was successful. We start with this dataset below. ### 3.1 Obtaining additional data for humans {#t3.1}

ExplorATE analysis consists of identification of decoy and target TE sequences, and quantification in Salmon applying Selective Alignment to handle multimapper reads derived from co-transcribed elements. Target TEs can be based on intergenic regions of the genome, or on a *de novo* transcriptome and its RepeatMasker file. To perform analysis with target TEs based on intergenic regions, ExplorATE requires a folder containing the `.fastq` files, a reference genome, a genetic model, and RepeatMasker annotations for the genome version. The `ExplorATE_data_test/inputs_hs` directory contains a reads folder with the `.fastq` files. The remaining files can be downloaded with the following commands. First set the working directory:

```
cd ExplorATE_data_test/inputs_hs
```

Downloading the files:

```
wget https://hgdownload.soe.ucsc.edu/goldenPath/hg38/bigZips/p13/hg38.p13.fa.gz
wget https://hgdownload.soe.ucsc.edu/goldenPath/hg38/bigZips/p13/hg38.p13.fa.out.gz
wget https://hgdownload.soe.ucsc.edu/goldenPath/hg38/bigZips/genes/hg38.refGene.gtf.gz
gzip -d *.gz
```

After downloading and uncompressing the files, you should have three additional files in your directory: `hg38.p13.fa` (genome), `hg38.p13.fa.out` (RepeatMasker output), `hg38.refGene.gtf` (gene model). Next we will run the mode `mo` from the Shell script ExplorATE to perform the analysis. The user must assign the corresponding paths, and run:

```
bash path/to/ExplorATE_shell_script/ExplorATE mo -p12 -b path/to/bedtools \\
-s path/to/salmon -f hg38.p13.fa -g hg38.refGene.gtf -r hg38.p13.fa.out \\
-e pe -l reads/ -o out_mo
```

The above command creates an `out_mo` folder with the output files. Within this folder, users will find the references file `references.csv` and a `quant_out` folder containing the Salmon estimates. Both files are used to import to environment R. From R console, user can use this code below to import estimates:

```
setwd("path/to/out_mo/folder")
hs.references <- read.csv("references.csv", sep = ";", header = F)
y <- ExplorATE::import.RTEs(
                      path.sal = "quant_out",
                      ref.sal = hs.references,
                      import_to = "edgeR",
                      conditions = rep("Ovary", each = 3)
                      )
```

For this reduced data set, only three replicates are included for a sample ("Ovary") that only has repeats of a single family of transposons (AluY). A more realistic analysis with Drosophila melanogaster data is included at the end of the section (section 3.2). If the target TEs are based on a *de novo* transcriptome (rather than using intergenic regions as shown above), ExplorATE also requires the *de novo* transcriptome and a transcriptome-derived RepeatMasker file. The `ExplorATE_data_test/inputs_hs` directory contains a *de novo* transcriptome and a RepeatMasker file for this reduced data set. Users can run the analysis with target TEs based on the *de novo* transcriptome with the following command:

```
bash path/to/ExplorATE_shell_script/ExplorATE mo -p12 -b path/to/bedtools \\
-s path/to/salmon -f hg38.p13.fa -g hg38.refGene.gtf -r hg38.p13.fa.out \\
-e pe -l reads/ -t trme_hs.fa -u RM_trme_hs.out -o out_mo_tr
```

3.2 Analyzing *Drosophila melanogaster* data

3.2.1 Obtaining additional data for *Drosophila melanogaster* data

We will analyze the *Drosophila melanogaster* ovarian cells data set from Ohtani *et al.* (2013). Ohtani *et al.* observed negative regulation of transposable elements after altering a *Drosophila* homologous of gametocyte-specific factor 1 (DmGTSF1). DmGTSF1 mutations caused the derepression of transposons

suggesting that DmGTSF1 is an integral factor in Piwi-piRISC-mediated transcriptional silencing. The FASTQ files are available from Gene Expression Omnibus (accession no. GSE47006). For this vignette, we will use the control samples (SRR851837) and Piwi (SRR851838). First, create a working directory (here dm_ExplorATE_wd) in the desired location and navigate to it:

```
mkdir dm_ExplorATE_wd
cd dm_ExplorATE_wd
```

Users can use the SRA Toolkit to download the files. To install SRA Toolkit follow the detailed instructions here. If you have an older SRA Toolkit version (releases <2.9.1) you can download and format the headers to be used with Trinity as shown below:

```
fastq-dump --defline-seq '@$sn[_$rn]/$ri' --split-files --accession SRR851837 > SRR851837.log
fastq-dump --defline-seq '@$sn[_$rn]/$ri' --split-files --accession SRR851838 > SRR851838.log
```

If the files downloaded successfully you should get something like this:

```
tail *.log

==> SRR851837.fastq-dump.log <==
Read 42134407 spots for SRR851837
Written 42134407 spots for SRR851837

==> SRR851838.fastq-dump.log <==
Read 48277060 spots for SRR851838
Written 48277060 spots for SRR851838
```

Once the files are downloaded, you have to rename them:

```
mv SRR851837.fastq control.fastq
mv SRR851838.fastq piwi.fastq
```

If you are using a version of SRA Toolkit 2.9.1 or higher you can use the fasterq-dump tool (much faster) as detailed in the program's web page, and later format the headers as follows. Replace <threads>argument appropriately and run:

```
fasterq-dump SRR851837 -e <threads> -t /dev/shm -p > SRR851837.log
fasterq-dump SRR851838 -e <threads> -t /dev/shm -p > SRR851838.log
```

If the files downloaded successfully you should get something like this:

```
tail *.log
spots read      : 42,134,407
reads read      : 42,134,407
reads written   : 42,134,407

spots read      : 48,277,060
reads read      : 48,277,060
reads written   : 48,277,060
```

Now, you have to modify the headers:

```
for i in SRR*;
do cat $i | sed 's/\sl:\w*\sm:\w*\sh:\w*//' | sed -r '/(^[\@]SRR\S+ )/s/\slength=\w*$'/\\/1/ |
sed -r 's/(^[\@]SRR\S+ )/@/' | sed -r 's/(^[\+]SRR\S+ )/+/' > headerok_$i;
done
```

This code above formats the headers to be used with Trinity and creates a new copy with the headerok_ tag. Verify that the file was modified correctly:

```
for i in headerok_*;
do grep -c '^@HW' $i;
done
```

If the files are correct, rename them:

```
mv headerok_SRR851837.fastq control.fastq
mv headerok_SRR851838.fastq piwi.fastq
```

Compress the .fastq files and move them to a new "reads" folder.

```
mkdir reads
gzip *.fastq
mv *.gz reads
```

Next, we will download other necessary files. All of these files are in the ExplorATE_data_test/drosophila directory. The user can clone the directory and continue with step #3.2.2 and #3.2.3, or download them as shown below: Downloading the reference genome for *D. melanogaster*

```
wget https://hgdownload.soe.ucsc.edu/goldenPath/dm3/bigZips/dm3.fa.gz
```

Downloading the GFF annotation file

```
wget http://hgdownload.soe.ucsc.edu/goldenPath/dm3/bigZips/genes/dm3.refGene.gtf.gz
```

Downloading the transposable elements annotation file from RepeatMasker

```
wget https://www.repeatmasker.org/genomes/dm3/RepeatMasker-rm405-db20140131/dm3.fa.out.gz
```

Renaming files

```
mv dm3.fa.gz genome_dm.fa.gz
mv dm3.refGene.gtf.gz geneModel_dm.gtf.gz
mv dm3.fa.out.gz RM_gen_dm.out.gz
```

uncompress .gz files

```
gzip -d *.gz
```

3.2.2 Target TEs based on de novo transcriptome

If the target TEs are based on a *de novo* transcriptome, this must be assembled from the reads and then masked with RepeatMasker. ExplorATE provides a *de novo* transcriptome and a transcriptome-derived RepeatMasker file for this vignette found in the `ExplorATE_data_test/inputs_dm` directory. Copy these files to the current working directory as shown below:

```
cp path/to/ExplorATE_data_test/inputs_dm/trme_dm.fa .
cp path/to/ExplorATE_data_test/inputs_dm/RM_trme_dm.out .
```

Alternatively, users can assemble their own transcriptomes and mask them with RepeatMasker from the command line or from the program's website. For example, for the *Drosophila melanogaster* dataset the transcriptome can be masked by assigning as "DNA source" `Drosophila`, and the code for the command line would be:

```
RepeatMasker -species Drosophila -a -e rmblast -pa <threads> -nolow -xm -u -gff
-dir <output directory> dm_transcriptome.fa
```

Replace `<threads>` and `<output directory>` arguments appropriately.

To mask the transcriptome form the RepeatMasker web service, users must select the `Fruit fly (Drosophila melanogaster)` in "DNA source" and specify "tar file" as "Return Format". And then select `"Do not mask`

satellites and simple repeats" in the "Lineage Specific Masking" item. Once the run is finished, RepeatMasker returns a link "Masked sequence and matches in compressed format" where files can be downloaded. Extract the files, place them in your working directory, and rename them as follows:

```
mv dm_transcriptome.fa_*.out dm_RMtr.out
```

The users should have the files genome_dm.fa, geneModel_dm.gtf, RM_gen_dm.out, trme_dm.fa and RM_trme_dm.fa in your desired working directory (here dm_ExplorATE_wd) and inside this directory a folder "reads" with the fastq files control.fastq.gz, and piwi.fastq.gz.

3.2.3 Run ExplorATE with *Drosophila melanogaster* dataset

Similar to the analysis with human data, we can base target TEs from intrgenic regions of the genome, or from the de novo transcriptome. Both examples are given below.

Genome-based target TEs

```
bash path/to/ExplorATE_shell_script/ExplorATE mo -p12 -b path/to/bedtools \\
-s path/to/salmon -f genome_dm.fa -g geneModel_dm.gtf -r RM_gen_dm.out \\
-e pe -l reads/ -o out_dm
```

Transcriptome-based target TEs

```
bash path/to/ExplorATE_shell_script/ExplorATE mo -p12 -b path/to/bedtools \\
-s path/to/salmon -f genome_dm.fa -g geneModel_dm.gtf -r RM_gen_dm.out \\
-e pe -l reads/ -t trme_dm.fa -u RM_trme_dm.fa -o out_dm_tr
```

Next, the user must import the Salmon estimates into the R environment and will be able to perform the differential expression analyzes as shown in section 4.4

4. Analysis with non-model organisms

4.1 Obtaining the data set

We will analyze the data set for *Liolaemus parthenos*, the only parthenogenetic lizard of the entire Pleurodonta (Iguanidae) clade. The FASTQ files are available from Gene Expression Omnibus (accession no.GSE173261). However in this vignette, we will use a data set with FASTQ files randomly sampled from the original file in order to reduce library sizes and generate replicates for differential expression analysis. Similarly, we subset the RepeatMasker output file and reduced the transcriptome size to run the pipeline in a reasonable time for the vignette. First clone the repository to your local directory and descompress all files.

The inputs_lp folder contains the output files for BLAST, TransDecoder, RepeatMasker, and the *de novo* transcriptome. You can use these files directly (continuing with step #4.2) or generate them as shown below. To generate the input files with the nmo_in mode of the ExplorATE shell script, users can modify the following code:

```
bash path/to/ExplorATE_shell_script/ExplorATE nmo_in -p 12 -n <blastp binary path> \\
-m <hmmerscan binary path> -r <RepeatMasker binary path> -d <TransDecoder directory path> \\
-u <SwissProt database> -f <Pfam database> -i <user-defined TE library> -t trme_lp.fa \\
-o inputs_lp
```

The script above generates the input files to run the ExplorATE pipeline. For the next steps in the vignette it is assumed that all input files are in a folder named inputs_lp

The script will generate the corresponding input files, except for the transcriptome which must be *de novo* assembled by the user or use the provided transcriptome (Lp_trme.fasta). For the execution of the next steps in the vignette we assume that all the input files are in the inputs_lp folder.

4.2 Running the ExplorATE pipeline for non-model organisms from the shell script

The ExplorATE shell script can run the pipeline in `nmo` mode. Below is the corresponding code for the *L. parthenos* input files.

```
bash path/to/ExplorATE_shell_script/ExplorATE nmo -p 12 -b path/to/bedtools \\
-s path/to/salmon -e pe -l reads/ -o out_lp -t trme_lp.fa -r RM_lp.out \\
-n blastAnot_lp.outfmt6 -d geneModel_lp.gff3 -w 80,80,80 -v 'higher_score' \\
-x subclass -q transcripts
```

In the output directory a folder `quant_out` is created with the Salmon estimates and the reference file`references.csv`. These files are used to import estimates into R as shown in section 4.4 . Further, ExplorATE writes a tab separated file (`repeats_in_UTRs.txt`) with the transcripts that contain repeats in the UTR regions. The first column indicates the name of the transcript, the second and third columns indicate the position of the UTR region in the transcript (start and end), the fourth column indicates the type of feature (5'-UTR or 3'-UTR) , the fifth and sixth columns indicate the position of the repeat in the transcript (start and end), and finally the last column indicates the name, class and family of the repeat separated by colon symbols (":", for example, "Plat_L3: LINE: CR1").

Users can generate the input files and run the pipeline simultaneously with the shell script's `nmo_all` mode.

4.3 Running the ExplorATE pipeline for non-model organisms with functions from the R package

To run the pipeline from the R ExplorATE package, assign the `inputs_lp` folder as the working directory and create the reference files for the salmon run.

```
setwd("path/to/folder/inputs_lp")
Lp.references <- ExplorATE::mk.reference(
                                        RepMask = "RM_lp.out",
                                        gff3 = "geneModel_lp.gff3",
                                        anot = "blastAnot_lp.outfmt6",
                                        cleanTEsProt = T,
                                        featureSum = T,
                                        outdir = "out_lp",
                                        rm.cotrans = T,
                                        overlapping = T,
                                        trme = "trme_lp.fa",
                                        stranded = F,
                                        by = "classRep",
                                        threads = 12,
                                        rule = c(80,80,80),
                                        over.res= "HS",
                                        annot_by = "transcripts"
                                        )
```

Next, the message `"please press 'y' if the names are correct"` is printed on the console to verify that the names assigned to each repetition do not contain ambiguities (consult the users' guide. Press `'y'+ENTER` and ExplorATE continues with the exclusion of elements co-transcribed with genes and the resolution of overlaps by the highest score. Finally, the transcripts with repeats are annotated by class , and the files `decoys.txt` and `trmeSalmon.fasta` are written to the output directory. These files will be used in the next step for abundance estimation in Salmon. With the `featureSum = T` argument, a file `features.summary.csv` is created with the transcripts containing co-transcribed TEs.

To run Salmon, users can run the following code

```
ExplorATE::run.salmon(index = "Lparthenos_index",
                      decoys = "out_lp/decoys.txt",
```

```
                        salmon_path = "~/programs/salmon-latest_linux_x86_64/bin/salmon",
                        pe_se = "pe",
                        kmer = 31,
                        trme = "out_lp/trmeSalmon.fasta",
                        lib_dir = "reads/",
                        threads = 12
)
```

Similar to the shell script, this function creates a folder `quant_out` in the working directory that contains the Salmon estimates and a reference file `references.csv` that are used to import the estimates as shown in the next section.

4.4 Importing estimates to R and estimating differential expression

The last step is to import the estimates into the R environment for subsequent analyzes such as differential expression analysis. The `import.RTEs()` function allows to import the estimates and create ready-to-use objects `DGEList` or `DESeqDataSet` for edgeR or DESeq respectively. Below is an example of how to import the estimates and perform the expression analysis in edgeR.

```
Lp.references <- read.csv("out_lp/references.csv", sep = ";", header = F)

y <- ExplorATE::import.RTEs(
                        path.sal = "quant_out",
                        ref.sal = Lp.references,
                        import_to = "edgeR",
                        conditions = rep(c("Brain", "Liver", "Ovary"), each = 3)
                        )
```

Now "y" is a `DGEList` object that is ready to perform the dispersion estimation.Only estimates for the ERV3 and CR1 families are included in the reduced data set. Users can download the complete set and perform differential expression analysis as follows.

Users can explore the dispersion of the data through an MDS plot from the limma package:

```
limma::plotMDS(y)
abline(h=0, v=0)
```

The GLM approach require build a design matrix to compare tissues and then the dispersion is estimated with the `estimateDisp()` function:

```
group.0 <- factor(rep(c("Brain", "Liver","Ovary"), each = 3))
des <- model.matrix( ~ 0 + group.0, data = y$samples)
colnames(des) <- c("Brain", "Liver","Ovary")
y <- edgeR::estimateDisp(y,des)
```

Finally the quasi-likelihood (QL) F-test is used to determine the differential expression:

```
fit_qlf <- edgeR::glmQLFit(y, des)
my.contrasts <- limma::makeContrasts(OvsB=Ovary-Brain,
                                     OvsL=Ovary-Liver,
                                     BvsL=Brain-Liver,
                                     levels=des)

qlf_OvsB <- edgeR::glmQLFTest(fit_qlf, contrast=my.contrasts[,"OvsB"])
qlf_OvsL <- edgeR::glmQLFTest(fit_qlf, contrast=my.contrasts[,"OvsL"])
qlf_BvsL <- edgeR::glmQLFTest(fit_qlf, contrast=my.contrasts[,"BvsL"])
```

The TEs differentially expressed in each contrast with FDR $<.05$ can be explored with the following code:

```
topTags_OvsB <- edgeR::topTags(qlf_OvsB, n = nrow(qlf_OvsB$table),sort.by = "none")
topTags_OvsB[topTags_OvsB$table$FDR<.05,]

topTags_OvsL <- edgeR::topTags(qlf_OvsL, n = nrow(qlf_OvsL$table),sort.by = "none")
topTags_OvsL[topTags_OvsL$table$FDR<.05,]

topTags_BvsL <- edgeR::topTags(qlf_BvsL, n = nrow(qlf_BvsL$table),sort.by = "none")
topTags_BvsL[topTags_BvsL$table$FDR<.05,]
```