Arkas: Repetitive Elements Quantification In Much Less Time

Timothy J. Triche, Jr, Anthony R. Colombo, Harold Pimentel

21 April, 2016

Contents

1 Introduction

Kallisto is software developed by Nicolas Bray, Harold Pimentel, Pall Melsted, and Lior Pachter (UC Berkeley) that analyzes 30 million unaligned paired-end RNA-Seq reads in less than 5 minutes on a standard laptop computer. Kallisto quantifies transcript abundance from input RNA-Seq reads by using a process, known as pseudoalignment, which identifies the read-transcript compatibility matrix. arkas is a BioConductor package that extends functions and utilities for RNA-Seq analysis from raw reads to results in minutes.

2 Reads to Quantification to Annotation

Arkas was designed to reduce the programmative steps required to quantify and annotate multitudes of sample directories. Arkas calls Kallisto to perform on- the-fly transcriptome indexing and quantification recursively for numerous sample directories. For RNA- Seq projects with numerous sequenced samples, Arkas encapsulates expensive preparatory routines. Arkas programmatically orders FASTQ files output from DNA sequencers and inputs a list required by Kallisto for processing multitudes of demultiplexed reads. The Arkas function 'runKallisto' recursively indexes transcriptomes and quantifies abundances for any number of samples. The function 'mergeKallisto' merges quantified output into an object of ofsubclass a KallistoExperiment-class, SummarizedExperiment-class. Standard mutators and accessor methods from SummarizedExperiment-methods are preserved in KallistoExperiment-methods. Gene annotation is performed from user-selected

bundled transcriptomes (ERCC, Ensembl, and/or RepBase) simultaneously merging annotated samples into one R object: KallistoExperiment. Arkas annotates genes for Homo-Sapiens GrCh38 and Mouse GrCm38 (NCBI). Routines such as 'annotateBundles' yields annotated genes from transcriptomes such as External RNA Control Consortium (ERCC), Ensembl release 81 of non-coding RNA, coding RNA, and a hg38 repeatome for both species.

2.1 Kallisto Installation

For linux systems, after installing the dependencies, kallisto is installed via:

```
mkdir /KallistoSource
cd /KallistoSource
git clone https://github.com/pachterlab/kallisto.git
cd ./kallisto
mkdir ./build
cd ./build
cmake ..
make
make install
```

3 Reference File Preparation

Arkas imports methods for identifying duplicated RepBase sequences, and imports these methods from TxDbLite. It is not uncommon for RepBase files with fixed species to have duplicated sequences and duplicated sequence names across fasta files. The package arkasData stores supplementary data for the package arkas; we've included raw RepBase files with duplicated repeats, and a set of fasta with manually excised duplicated RepBase sequences across fasta files.

3.1 Identifying Duplicated Sequences in RepBase References

Kallisto source must be built locally by indexing the reference files. There can not be duplicated sequences across reference fasta; Arkas will identify where in there reference duplicates occur.

```
suppressWarnings(suppressPackageStartupMessages(library(arkas)))
suppressPackageStartupMessages(library(arkasData))
suppressPackageStartupMessages(library(TxDbLite))
pathBase<-system.file("extdata",package="arkasData")
FastaPath <- paste0(pathBase, "/fasta")
fastaFilesDuplicates <- c( "ERCC.fa", ## spike-in controls
                "Homo_sapiens.RepBase.20_05.humrep.fa", ## repeats
                "Homo_sapiens.RepBase.20_05.humsub.fa")  ## ALUs and such
findDupes(paste0(FastaPath,"/",fastaFilesDuplicates)) #return a df with column of 0
```

```
## Warning in .Call2("fasta_index", filexp_list, nrec, skip, seek.first.rec, :
## reading FASTA file /usr/local/lib/R/site-library/arkasData/extdata/fasta/
## Homo_sapiens.RepBase.20_05.humrep.fa: ignored 163 invalid one-letter
## sequence codes

## Warning in .Call2("fasta_index", filexp_list, nrec, skip, seek.first.rec, :
## reading FASTA file /usr/local/lib/R/site-library/arkasData/extdata/
```

```
## fasta/Homo_sapiens.RepBase.20_05.humsub.fa: ignored 16 invalid one-letter
## sequence codes
```

```
##                                                                                                    duplicates
## /usr/local/lib/R/site-library/arkasData/extdata/fasta/ERCC.fa                                               0
## /usr/local/lib/R/site-library/arkasData/extdata/fasta/Homo_sapiens.RepBase.20_05.humrep.fa                  0
## /usr/local/lib/R/site-library/arkasData/extdata/fasta/Homo_sapiens.RepBase.20_05.humsub.fa                  0
```

Example of output with duplicated repeat sequences in Repeatome:

```
                                        duplicates
Homo_sapiens.RepBase.20_12.merged.fa1     LTR26B
Homo_sapiens.RepBase.20_12.merged.fa2      SVA_A
```

```
The duplicated repeats LTR26B at row number 222 have identical sequences: TRUE
The duplicated repeats LTR26B at row number 1166 have identical sequences: TRUE
The duplicated repeats SVA_A at row number 671 have identical sequences: TRUE
The duplicated repeats SVA_A at row number 1207 have identical sequences: TRUE
There are duplicated sequence names in your FASTA files:
```

If duplicate sequences exist in one's repeatome, Arkas identifies the duplicate sequence name, and additionally determines if the sequences are identical. The user can then manually excise one of the duplicates.

3.2 Quantifying Transcripts with Unique Reference Transcripts in Repeatome

Arkas saves time by generating a reference fasta file only if one does not exist. Note that Arkas enforces the uniformity of references generated by the same version of Kallisto, including the Kallisto version which generates the reference within the name. The reference name that Arkas generates specifies the Kallisto version, and all of the group transcriptomes. In this case, we have already generated the reference fasta, and can regenerate the reference by deleting it.

```
suppressWarnings(suppressPackageStartupMessages(library(arkas)))
suppressPackageStartupMessages(library(arkasData))
suppressPackageStartupMessages(library(TxDbLite))
pathBase<-system.file("extdata",package="arkasData")
if(file.exists("/data")==FALSE){
    system("sudo mkdir /data && sudo mkdir /data/output")
     system("sudo chmod -R 777 /data")
 }
   if(file.exists("/data/output")==FALSE){
     system("sudo mkdir /data/output && sudo chmod -R 777 /data/output")
   }
OutputPath<-"/data/output"
fastqPath <- paste0(pathBase, "/fastq")
samples <- c(MrN="MrN", MrT="MrT")
FastaFiles <- c( "ERCC.fa", ## spike-in controls
                 "Homo_sapiens.RepBase.20_05.merged.fa") #no duplicates
indexName <- indexKallisto(fastaFiles=FastaFiles, fastaPath=FastaPath)$indexName
```

```
## kmer length size: 31
```

```
## Warning in .Call2("fasta_index", filexp_list, nrec, skip, seek.first.rec, :
## reading FASTA file Homo_sapiens.RepBase.20_05.merged.fa: ignored 147
## invalid one-letter sequence codes
```

```
## found no duplicated sequence names .... no dupes found
```

```
indexName  #prints the full index Name
```

```
## [1] "ERCC_mergedWith_Homo_sapiens.RepBase.20_05.mergedkallisto.0.42.4.fa.kidx"
```

```
Xtension<-"_*"

library(parallel)
results <- lapply(samples,
                  runKallisto,
                  indexName=indexName,
                  fastqPath=fastqPath,
                  fastaPath=FastaPath,
                  bootstraps=20,
                  outputPath=OutputPath,
                  extension=Xtension)
```

```
## Warning in lapply(prefixes, as.numeric): NAs introduced by coercion
```

```
## Warning in lapply(prefixes, as.numeric): NAs introduced by coercion
```

```
## Warning in lapply(prefixes, as.numeric): NAs introduced by coercion
```

```
## Warning in lapply(prefixes, as.numeric): NAs introduced by coercion
```

```
## Warning in lapply(prefixes, as.numeric): NAs introduced by coercion
```

3.3 Merging Kallisto Quantification

Arkas runs kallisto pseudo-alignment quantification into a KallistoExperiment object, as an S4 object. The SummarizedExperiment accessors are the same for KallistoExperiment objects because KallistoExperiment containers are a sub-class of SummarizedExperiments We enforce uniformity between kallisto versions because if the kallisto versions are not the same, then there exists data variance between different KallistoExperiments' assay data and indexed references. In order to annotate the merged KallistoExperiment, TxDbLite is used to create SQL-lite transcriptome/repeatome databases. This tutorial already created the transcriptome package which can be accessed using transcriptomes() accessor on the KallistoExperiment object.

```
suppressPackageStartupMessages(library(arkas))
library(arkasData)
suppressPackageStartupMessages(library(TxDbLite))
samples<-c("n1","n2","n4","s1","s2","s4")
pathBase<-system.file("extdata",package="arkasData")
merged <- mergeKallisto(samples, outputPath=pathBase)
```

```
## Setting transcriptome automatically from Kallisto call string.
```

```
assays(merged)
```

```
## List of length 4
## names(4): est_counts eff_length est_counts_mad tpm
```

```
kallistoVersion(merged)
```

```
## [1] "0.42.3"
```

```
transcriptomes(merged)
```

```
## [1] "ErccDbLite.ERCC.97, EnsDbLite.Hsapiens.81, RepDbLite.Hsapiens.2007"
```

```
libraryPath<-system.file("extdata","Libraries",package="arkasData")
command<-paste0("sudo R CMD INSTALL ",libraryPath,"/",dir(libraryPath))
lapply(command,function(x) system(x))
```

```
## [[1]]
## [1] 0
##
## [[2]]
## [1] 0
##
## [[3]]
## [1] 0
```

```
merged<-annotateFeatures(merged, level="transcript") #annotate features using transcriptomes
```

```
## Loading required package: ErccDbLite.ERCC.97
```

```
## Loading required package: EnsDbLite.Hsapiens.81
```

```
## Warning in .Seqinfo.mergexy(x, y): The 2 combined objects have no sequence levels in common. (Use
##   suppressWarnings() to suppress this warning.)
```

```
## Loading required package: RepDbLite.Hsapiens.2007
```

```
## Warning in .Seqinfo.mergexy(x, y): The 2 combined objects have no sequence levels in common. (Use
##   suppressWarnings() to suppress this warning.)
```

```
tail(features(merged)) #annotated GRanges features
```

```
## GRanges object with 6 ranges and 9 metadata columns:
##            seqnames    ranges strand | tx_length gc_content       tx_id
##               <Rle> <IRanges>  <Rle> | <integer>  <numeric> <character>
##      SVA_D    SVA_D [1, 1386]      * |      1386  0.6075036       SVA_D
##      SVA_E    SVA_E [1, 1382]      * |      1382  0.6099855       SVA_E
##      SVA_F    SVA_F [1, 1375]      * |      1375  0.6094545       SVA_F
```

```
##    AluYb11  AluYb11 [1,  289]      * |       289  0.6332180     AluYb11
##    AluYb10  AluYb10 [1,  288]      * |       288  0.6354167     AluYb10
##   AluYb8a1 AluYb8a1 [1,  287]      * |       287  0.6376307    AluYb8a1
##               gene_id   gene_name    entrezid  tx_biotype gene_biotype
##           <character> <character> <character> <character>  <character>
##      SVA_D        <NA>        <NA>        <NA>         SVA other_repeat
##      SVA_E        <NA>        <NA>        <NA>         SVA other_repeat
##      SVA_F        <NA>        <NA>        <NA>         SVA other_repeat
##    AluYb11        <NA>        <NA>        <NA>         Alu         SINE
##    AluYb10        <NA>        <NA>        <NA>         Alu         SINE
##   AluYb8a1        <NA>        <NA>        <NA>         Alu         SINE
##           biotype_class
##             <character>
##      SVA_D        repeat
##      SVA_E        repeat
##      SVA_F        repeat
##    AluYb11        repeat
##    AluYb10        repeat
##   AluYb8a1        repeat
##   -------
##   seqinfo: 1543 sequences from 3 genomes (N/A, GRCh38, RepBase20_07); no seqlengths
```

4 RPKM/FPKM to Transcripts-Per-Million (TPM)

Many RNA-Seq experiments derive counts as reads per kilobase per million, or fragments per kilobase per million. The quantification from RNA-Seq transcript abundances is dependent on transcript length in proportion to relative abundance. The estimated probability of reads generated by a transcript is given by counting the number of reads that align to transcript divided by total number of mapped reads. Where the definition of transcript-per-million, TPM, is given as mean transcript length in kilobases multiplied by RPKM. The mean for transcript dependent lengths gives weighted measure for expression of the lengths across iso-forms. Arkas handles RPKM to TPM conversion by dividing the selected transcript RPKM score by the sum of all RPKM, and multiplying by 1e6 (Li,2009)

```
suppressWarnings(suppressPackageStartupMessages(library(arkas)))
suppressPackageStartupMessages(library(arkasData))
suppressPackageStartupMessages(library(TxDbLite))
samples<-c("n1","n2","n4","s1","s2","s4")
pathBase<-system.file("extdata",package="arkasData")
merged <- mergeKallisto(samples, outputPath=pathBase)
```

```
## Setting transcriptome automatically from Kallisto call string.
```

```
tail(tpm(merged))
```

```
##                n1          n2        n4        s1        s2       s4
## SVA_D    2.0282325 2.995275258 2.5212409 6.1764362 2.4633671 4.310314
## SVA_E    0.3240263 0.509904284 0.2511909 0.1922470 0.0000000 0.000000
## SVA_F    0.9372103 0.311834338 2.0003786 2.3443996 1.0634931 1.842919
## AluYb11  0.0000000 0.000000000 0.0000000 0.0000000 0.0000000 0.000000
## AluYb10  0.0000000 0.004649301 0.0000000 0.3948314 0.0000000 0.353331
## AluYb8a1 0.0000000 0.000000000 0.0000000 0.0000000 0.0771715 0.000000
```

```
tail(tpm(merged))
```

```
##                n1          n2        n4        s1        s2       s4
## SVA_D    2.0282325 2.995275258 2.5212409 6.1764362 2.4633671 4.310314
## SVA_E    0.3240263 0.509904284 0.2511909 0.1922470 0.0000000 0.000000
## SVA_F    0.9372103 0.311834338 2.0003786 2.3443996 1.0634931 1.842919
## AluYb11  0.0000000 0.000000000 0.0000000 0.0000000 0.0000000 0.000000
## AluYb10  0.0000000 0.004649301 0.0000000 0.3948314 0.0000000 0.353331
## AluYb8a1 0.0000000 0.000000000 0.0000000 0.0000000 0.0771715 0.000000
```

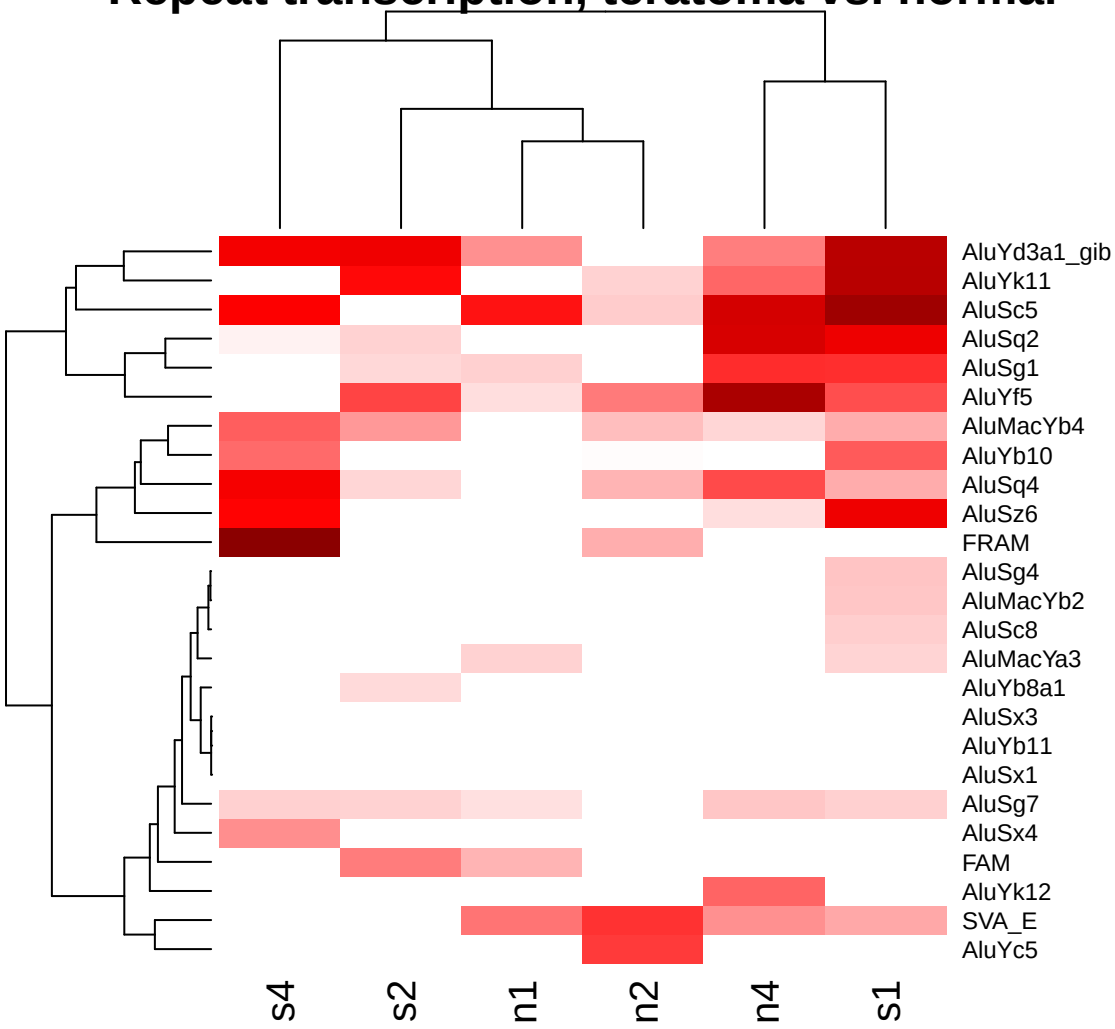
5 Repeat Analysis

After creating the repeatome, arkas can be used to analyze repetitive elements For instance, one can plot repeat element transcripts using (counts/bootstrap MADs) as effect size, or analyze families of Alu, LTR, or Endongenous Retroviruses. Below we plot a heat map of quantified transcripts from our repeatome for species Homo Sapiens.

```
topKbyMAD <- function(kexp, k=25) {
  tpm(kexp)[rev(order(rowMeans(counts(kexp) / mad(kexp))))[1:k],]
}
suppressWarnings(suppressPackageStartupMessages(library(arkas)))
suppressPackageStartupMessages(library(arkasData))
suppressPackageStartupMessages(library(TxDbLite))
samples<-c("n1","n2","n4","s1","s2","s4")
pathBase<-system.file("extdata",package="arkasData")
merged <- mergeKallisto(samples, outputPath=pathBase)
```

```
## Setting transcriptome automatically from Kallisto call string.
```

```
    heatmap(log1p(topKbyMAD(merged)), scale="none",
        col=colorRampPalette(c("white","red","darkred"))(255),
        main="Repeat transcription, teratoma vs. normal")
```



6 References

Bo, Li, et. al, "RNA-Seq Gene Expression Estimation with Read Mapping Uncertainty." Oxford Journals Bioinformatics. Oxford Journal, 9 Dec. 2009. Web. 20 June 2015. http://bioinformatics.oxfordjournals.org/content/26/4/493.full.