RAProject

Mingzhi Ye

2022/10/10

```
sce0 <- readRDS('toyTCRdata.rds')
sce <- clonoStats(sce0, method = 'EM')

#A<-sce0$contigs[sce0$contigs[, 'chain'] %in% c('TRA','TRB')]
for (i in 1:length(sce0$contigs)){
  #sce0$contigs[[i]]  "DFrame"  "S4Vectors"
  #sce0$contigs[[i]]<-sce0$contigs[[i]] %>% filter(chain %in% c("TRA","TRB"))
  #sce0$contigs[[i]]<-sce0$contigs[[i]][filter(sce0$contigs[[i]]$chain %in% c("TRA","TRB")),]
  sce0$contigs[[i]]<-sce0$contigs[[i]][order(sce0$contigs[[i]]$chain),]

}
```

Generate sample pool

```
samplePool=DataFrameList()
poolSize=1
for (i in 1:length(sce0$contigs)){
  if(length(sce0$contigs[[i]]$chain)==2 & sce0$contigs[[i]]$chain[1]=='TRA' & sce0$contigs[[i]]$chain[2]

    samplePool[[poolSize]]=sce0$contigs[[i]]
    poolSize=poolSize+1
  }

}
poolSize=poolSize-1
```

Functions for generate a sample

```
getPoisson<-function(){
  a=-1
  while(a<0 | a>4){
    a=rpois(1, 1)
  }
  return(a)
}
getClonotype<-function(){
```

```
  numberA=getPoisson()
  numberB=getPoisson()
  result=DataFrame()
  indexInPool=sample.int(poolSize, 1, replace=TRUE)
  if (numberA>=1){
    result=rbind(result,samplePool[[indexInPool]][1,])
    numberA=numberA-1
  }
  if (numberB>=1){
    result=rbind(result,samplePool[[indexInPool]][2,])
    numberB=numberB-1
  }
  while(numberA>0 | numberB>0){
    indexInPool=sample.int(poolSize, 1, replace=TRUE)
    if (numberA>=1){
      result=rbind(result,samplePool[[indexInPool]][1,])
      numberA=numberA-1
    }
    else{
      result=rbind(result,samplePool[[indexInPool]][2,])
      numberB=numberB-1
    }
  }
  if(length(result$barcode)){
    for(i in 1:length(result$barcode)){
      result$barcode[i]=result$barcode[1]
      result$sample[i]='sim'
    }
  }
  return(result)

}
```

Use the functions to generate a sample with size 50

However, function 'clonoStats' is not compatible with SimpleDFrameList, how to transform SimpleDFrameList to CompressedSplitDFrameList

```
SampleSize=50
getSample<-function(samplesize){
  sample=DataFrame()
  for(i in 1:samplesize){
    sample=rbind(sample,getClonotype())
  }
  samplelist<-split(sample,sample$barcode)
  # sample1<-as(sample,'SplitDFrameList')
  return(samplelist)

}
# samplelist=getSample(SampleSize)
# EMpredicted <- clonoStats(samplelist, method = 'EM')
#UNpredicted <- clonoStats(samplelist, method = 'unique')
```

Generate hashmap. Key is clonotype in Truth(sample pool), Value is percentage of the clonotype in Truth

```
TruthPercentageMap=hashmap()

for(i in 1:poolSize){

  key=paste(samplePool[[i]]$cdr3[1],samplePool[[i]]$cdr3[2],sep=' ')

  if(is.null(TruthPercentageMap[[key]])){
    TruthPercentageMap[[key]]<-1/poolSize
  }
  else{
    TruthPercentageMap[[key]]<-TruthPercentageMap[[key]]+1/poolSize
  }

}
```

Get Variation Distance

```
getDistance<-function(predicted){

  clonotypes=clonoNames(predicted)
  abundance=clonoAbundance(predicted)
  SumAbundance=sum(abundance)
  distance=0
  seen <- hashmap()
  for(i in 1:length(abundance)){
    if(is.null(TruthPercentageMap[[clonotypes[i]]])){
      distance=distance+abundance[i]/SumAbundance
    }
    else{
      seen[[clonotypes[i]]]<-TruthPercentageMap[[clonotypes[i]]]
      distance=distance+abs(abundance[i]/SumAbundance-TruthPercentageMap[[clonotypes[i]]])
    }
  }

  distance=distance+Reduce("+", values(TruthPercentageMap)) -Reduce("+", values(seen))
  return(distance)
}
```

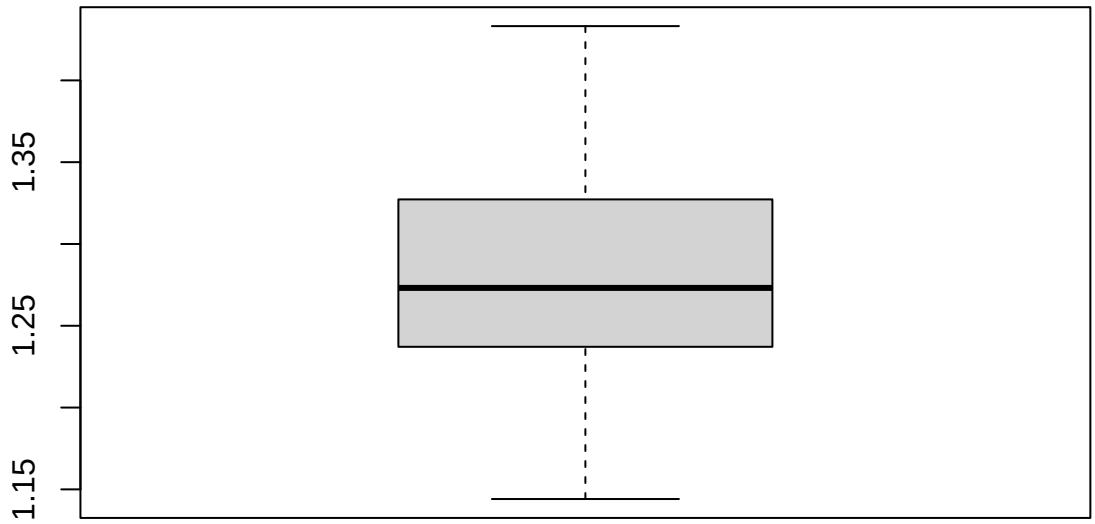
Calculate distances for samples

```
DistanceListFromEM=c()
DistanceListFromUN=c()
for(i in 1:20){
```

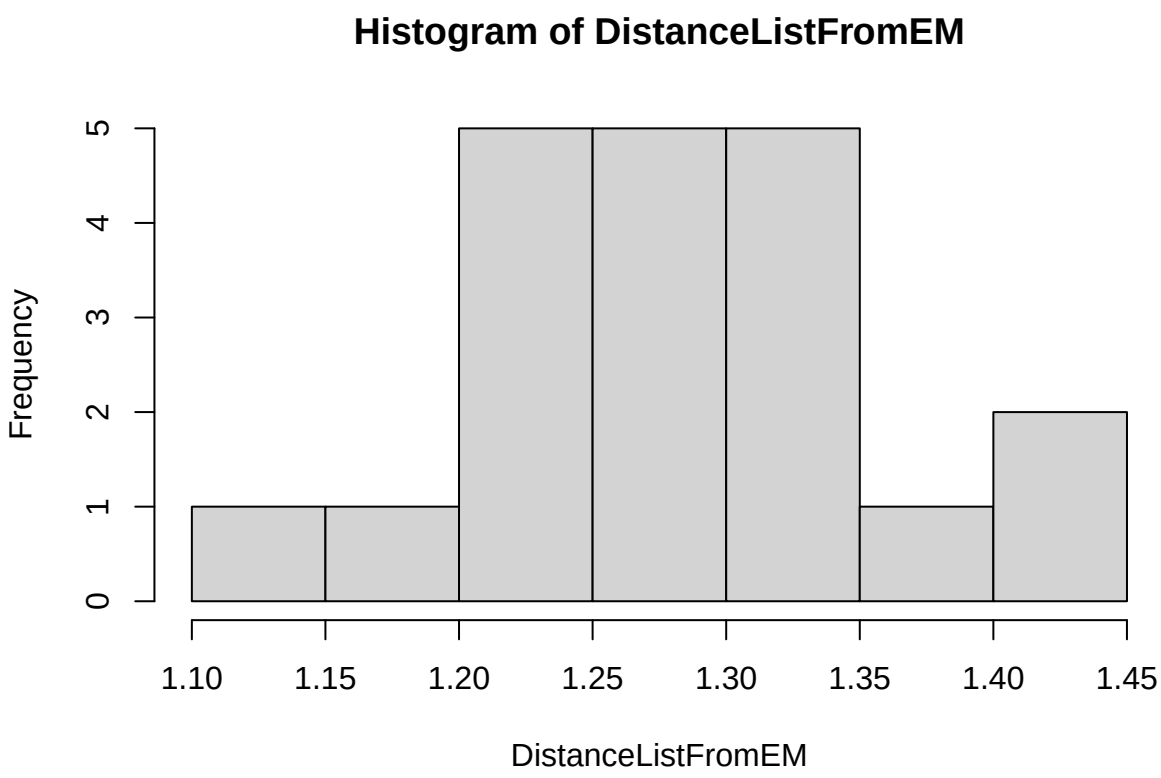
```
  samplelist=getSample(200)
  EMpredicted <- clonoStats(samplelist, method = 'EM')
  distance1=getDistance(EMpredicted)
  DistanceListFromEM[i]=distance1
  # UNpredicted <- clonoStats(samplelist, method = 'unique')
  # distance2=getDistance(UNpredicted)
  # DistanceListFromUN[i]=distance2
}
```

Visualize

```
boxplot(DistanceListFromEM)
```




```
hist(DistanceListFromEM)
```


Histogram of DistanceListFromEM
Frequency
0
1
2
3
4
5
1.10
1.15
1.20
1.25
1.30
1.35
1.40
1.45
DistanceListFromEM