Estatísticas Pós-Extração: Recomendação do TRUST4

Jean Resende

Contexto

Após a extração dos receptores TCR e BCR, os desenvolvedores do TRUST4 recomendam a da função contida em **trust-stats.py**. No entanto, essa função é focada nos dados output do TRUST4, como foco mais específico no arquivo *report*. O que pretendo fazer aqui, é converter os cálculos aplicados nesta função em **python** para **R** e mostrar como aplicar nos dados brutos extraídos com TRUST4 e também aos dados de clonótipos calculados e agrupados com o **immunarch**.

Executando trust-stats.py nos dados report obtidos com TRUST4

Primeiro gerei um txt contendo so nomes dos arquivos para rodar a função nestes dados.

```
# -- tabelas originais
outputTrust4_report <-
  "../../../../Projetos/Bigdata/BigData/BigData/repertorio_tcrbcr_acc/data/outputTrust4_re

files <- list.files(outputTrust4_report)

writeLines(files, "samplesNames.txt") # gerando samplesNames.txt
file.exists("samplesNames.txt")


# -- tabelas clones
files_clones <- list.files("../00_clones/data/")

writeLines(files_clones, "samplesNames_clones.txt") # gerando samplesNames.txt
file.exists("samplesNames_clones.txt")
```

Aqui executo a função para os dados obtidos com o TRUST4.

```
while read SAMP
    do
    echo "processing ${SAMP}"
    python3 trust-stats.py -r ../../../../Projetos/Bigdata/BigData/BigData/repertorio_tcrb
    done < samplesNames.txt
```

Para os arquivos gerados com o TRUST4, a função funciona perfeitamente. No entanto, para os dados que gerei com os clonótipos, a função retorna NA para todos os valores.

```
while read SAMP
    do
    echo "processing ${SAMP}"
    python3 trust-stats.py -r ../00_clones/data_clones/${SAMP} > results_trust-stats_clone
    done < samplesNames_clones.txt
```

Converter funções para o R

Input das tabelas de report e clones

Visto que o script trust-stats,py retorna apenas NA para os arquivos referentes aos clones, vou montar as funções em R e assim executar essas funções nos dados report gerados pelo TRUST4 e aos dados gerados na etapa de clones.

Começo importando para o ambiente as tabelas referentes aos dados gerados pelo TRUST4 e as tabelas de clones.

```
# -- importar tabelas para o R

## -- sequencias brutas
dir_sequencias <- "../../../../Projetos/Bigdata/BigData/BigData/repertorio_tcrbcr_acc/data

arquivos_sequencias <- list.files(dir_sequencias)

tcrbcr_sequencias <- list()

# loop para ler cada arquivo e armazenar os dados na lista
for (arquivo in arquivos_sequencias) {
  nome <- gsub("\\.tsv$", "", arquivo)
```

```
  dados <- read.delim(file.path(dir_sequencias , arquivo), sep = "\t")
  tcrbcr_sequencias[[nome]] <- dados
}

## -- clones
dir_clones <- "../00_clones/data_clones"

arquivos_clones <- list.files(dir_clones)

tcrbcr_clones <- list()

# loop para ler cada arquivo e armazenar os dados na lista
for (arquivo in arquivos_clones) {
  nome <- gsub("\\.tsv$","", arquivo)
  dados <- read.delim(file.path(dir_clones, arquivo), sep = "\t")
  tcrbcr_clones[[nome]] <- dados
}
```

Aqui estou removendo os objetos do ambiente R exceto as listas contendo as tabelas que irei aplicar as funções.

```
obj_a_manter <- c("tcrbcr_sequencias", "tcrbcr_clones")
obj_no_ambiente <- ls()
obj_a_remover <- setdiff(obj_no_ambiente, obj_a_manter)
rm(list = obj_a_remover)
rm(obj_a_remover, obj_no_ambiente)
```

Montagem das funções

Montei a as funções em R para o cálculo da abundância, riqueza, cpk, entropia e clonalidade.

```
# abundance
calc.abundance <- function(count){
  return(sum(count))
}

# richness
calc.richness <- function(count){
  return(length(count))
}
```

```r
# cpk
calc.cpk <- function(count){
  return(length(count) / sum(count) * 1000)
}

# entropy
calc.entropy <- function(count){
  j <- 0
  for (i in seq_along(count)) {
    t <- (-count[i])/sum(count) * log(count[i]/sum(count))
    j <- j + t
  }
  return(j)
}

# clonality
calc.clonality <- function(count){
  return(1 - calc.entropy(count) / log(length(count)))
}
```

Execução das funções

Apliquei as funções nas tabelas contendo as sequências diretas do TRUST4 e nos clones.

```r
calc_tcrbcr_sequencias <- list()

for (nome in names(tcrbcr_sequencias)) {

  calc_tcrbcr_sequencias[[nome]] <- data.frame(
    chain = c("IGH","IGK","IGL","TRA","TRB","TRG","TRD"),
    Abundance = rep(NA, 7),
    Richness = rep(NA, 7),
    CPK = rep(NA, 7),
    Entropy = rep(NA, 7),
    Clonality = rep(NA, 7)
  )
}

for (nome in names(tcrbcr_sequencias)) {
  df <- tcrbcr_sequencias[[nome]]
```

```r
df2 <- calc_tcrbcr_sequencias[[nome]]

# abundance
df2[df2$chain == "IGH", "Abundance"] <-
  calc.abundance(df$X.count[substr(df$V, 1,3) == "IGH"])

df2[df2$chain == "IGK", "Abundance"] <-
  calc.abundance(df$X.count[substr(df$V, 1,3) == "IGK"])

df2[df2$chain == "IGL", "Abundance"] <-
  calc.abundance(df$X.count[substr(df$V, 1,3) == "IGL"])

df2[df2$chain == "TRA", "Abundance"] <-
  calc.abundance(df$X.count[substr(df$V, 1,3) == "TRA"])

df2[df2$chain == "TRB", "Abundance"] <-
  calc.abundance(df$X.count[substr(df$V, 1,3) == "TRB"])

df2[df2$chain == "TRG", "Abundance"] <-
  calc.abundance(df$X.count[substr(df$V, 1,3) == "TRG"])

df2[df2$chain == "TRD", "Abundance"] <-
  calc.abundance(df$X.count[substr(df$V, 1,3) == "TRD"])


# richness
df2[df2$chain == "IGH", "Richness"] <-
  calc.richness(df$X.count[substr(df$V, 1,3) == "IGH"])

df2[df2$chain == "IGK", "Richness"] <-
  calc.richness(df$X.count[substr(df$V, 1,3) == "IGK"])

df2[df2$chain == "IGL", "Richness"] <-
  calc.richness(df$X.count[substr(df$V, 1,3) == "IGL"])

df2[df2$chain == "TRA", "Richness"] <-
  calc.richness(df$X.count[substr(df$V, 1,3) == "TRA"])

df2[df2$chain == "TRB", "Richness"] <-
  calc.richness(df$X.count[substr(df$V, 1,3) == "TRB"])
```

```r
df2[df2$chain == "TRG", "Richness"] <-
  calc.richness(df$X.count[substr(df$V, 1,3) == "TRG"])

df2[df2$chain == "TRD", "Richness"] <-
  calc.richness(df$X.count[substr(df$V, 1,3) == "TRD"])

# cpk
df2[df2$chain == "IGH", "CPK"] <-
  calc.cpk(df$X.count[substr(df$V, 1,3) == "IGH"])

df2[df2$chain == "IGK", "CPK"] <-
  calc.cpk(df$X.count[substr(df$V, 1,3) == "IGK"])

df2[df2$chain == "IGL", "CPK"] <-
  calc.cpk(df$X.count[substr(df$V, 1,3) == "IGL"])

df2[df2$chain == "TRA", "CPK"] <-
  calc.cpk(df$X.count[substr(df$V, 1,3) == "TRA"])

df2[df2$chain == "TRB", "CPK"] <-
  calc.cpk(df$X.count[substr(df$V, 1,3) == "TRB"])

df2[df2$chain == "TRG", "CPK"] <-
  calc.cpk(df$X.count[substr(df$V, 1,3) == "TRG"])

df2[df2$chain == "TRD", "CPK"] <-
  calc.cpk(df$X.count[substr(df$V, 1,3) == "TRD"])

# entropy
df2[df2$chain == "IGH", "Entropy"] <-
  calc.entropy(df$X.count[substr(df$V, 1,3) == "IGH"])

df2[df2$chain == "IGK", "Entropy"] <-
  calc.entropy(df$X.count[substr(df$V, 1,3) == "IGK"])

df2[df2$chain == "IGL", "Entropy"] <-
  calc.entropy(df$X.count[substr(df$V, 1,3) == "IGL"])

df2[df2$chain == "TRA", "Entropy"] <-
  calc.entropy(df$X.count[substr(df$V, 1,3) == "TRA"])
```

```
  df2[df2$chain == "TRB", "Entropy"] <-
    calc.entropy(df$X.count[substr(df$V, 1,3) == "TRB"])

  df2[df2$chain == "TRG", "Entropy"] <-
    calc.entropy(df$X.count[substr(df$V, 1,3) == "TRG"])

  df2[df2$chain == "TRD", "Entropy"] <-
    calc.entropy(df$X.count[substr(df$V, 1,3) == "TRD"])

  # clonality
  df2[df2$chain == "IGH", "Clonality"] <-
    calc.clonality(df$X.count[substr(df$V, 1,3) == "IGH"])

  df2[df2$chain == "IGK", "Clonality"] <-
    calc.clonality(df$X.count[substr(df$V, 1,3) == "IGK"])

  df2[df2$chain == "IGL", "Clonality"] <-
    calc.clonality(df$X.count[substr(df$V, 1,3) == "IGL"])

  df2[df2$chain == "TRA", "Clonality"] <-
    calc.clonality(df$X.count[substr(df$V, 1,3) == "TRA"])

  df2[df2$chain == "TRB", "Clonality"] <-
    calc.clonality(df$X.count[substr(df$V, 1,3) == "TRB"])

  df2[df2$chain == "TRG", "Clonality"] <-
    calc.clonality(df$X.count[substr(df$V, 1,3) == "TRG"])

  df2[df2$chain == "TRD", "Clonality"] <-
    calc.clonality(df$X.count[substr(df$V, 1,3) == "TRD"])


  calc_tcrbcr_sequencias[[nome]] <- df2
}

calc_tcrbcr_sequencias$`130723_UNC9-SN296_0386_BC2E4WACXX_ACTTGA_L003_report`
```

```
  chain Abundance Richness      CPK   Entropy  Clonality
1   IGH         0        0      NaN 0.0000000 1.00000000
2   IGK        16        9 562.5000 2.1006789 0.04393982
3   IGL         9        3 333.3333 0.6837389 0.37763403
```

```
4   TRA         0         0      NaN 0.0000000 1.00000000
5   TRB         5         3 600.0000 0.9502705 0.13502648
6   TRG         0         0      NaN 0.0000000 1.00000000
7   TRD         0         0      NaN 0.0000000 1.00000000
```

```r
calc_tcrbcr_clones <- list()

for (nome in names(tcrbcr_clones)) {

  calc_tcrbcr_clones[[nome]] <- data.frame(
    chain = c("IGH","IGK","IGL","TRA","TRB","TRG","TRD"),
    Abundance = rep(NA, 7),
    Richness = rep(NA, 7),
    CPK = rep(NA, 7),
    Entropy = rep(NA, 7),
    Clonality = rep(NA, 7)
  )
}

for (nome in names(tcrbcr_clones)) {
  df <- tcrbcr_clones[[nome]]
  df2 <- calc_tcrbcr_clones[[nome]]

  # abundance
  df2[df2$chain == "IGH", "Abundance"] <-
    calc.abundance(df$count[substr(df$V, 1,3) == "IGH"])

  df2[df2$chain == "IGK", "Abundance"] <-
    calc.abundance(df$count[substr(df$V, 1,3) == "IGK"])

  df2[df2$chain == "IGL", "Abundance"] <-
    calc.abundance(df$count[substr(df$V, 1,3) == "IGL"])

  df2[df2$chain == "TRA", "Abundance"] <-
    calc.abundance(df$count[substr(df$V, 1,3) == "TRA"])

  df2[df2$chain == "TRB", "Abundance"] <-
    calc.abundance(df$count[substr(df$V, 1,3) == "TRB"])

  df2[df2$chain == "TRG", "Abundance"] <-
    calc.abundance(df$count[substr(df$V, 1,3) == "TRG"])
```

```r
df2[df2$chain == "TRD", "Abundance"] <-
  calc.abundance(df$count[substr(df$V, 1,3) == "TRD"])


# richness
df2[df2$chain == "IGH", "Richness"] <-
  calc.richness(df$count[substr(df$V, 1,3) == "IGH"])

df2[df2$chain == "IGK", "Richness"] <-
  calc.richness(df$count[substr(df$V, 1,3) == "IGK"])

df2[df2$chain == "IGL", "Richness"] <-
  calc.richness(df$count[substr(df$V, 1,3) == "IGL"])

df2[df2$chain == "TRA", "Richness"] <-
  calc.richness(df$count[substr(df$V, 1,3) == "TRA"])

df2[df2$chain == "TRB", "Richness"] <-
  calc.richness(df$count[substr(df$V, 1,3) == "TRB"])

df2[df2$chain == "TRG", "Richness"] <-
  calc.richness(df$count[substr(df$V, 1,3) == "TRG"])

df2[df2$chain == "TRD", "Richness"] <-
  calc.richness(df$count[substr(df$V, 1,3) == "TRD"])

# cpk
df2[df2$chain == "IGH", "CPK"] <-
  calc.cpk(df$count[substr(df$V, 1,3) == "IGH"])

df2[df2$chain == "IGK", "CPK"] <-
  calc.cpk(df$count[substr(df$V, 1,3) == "IGK"])

df2[df2$chain == "IGL", "CPK"] <-
  calc.cpk(df$count[substr(df$V, 1,3) == "IGL"])

df2[df2$chain == "TRA", "CPK"] <-
  calc.cpk(df$count[substr(df$V, 1,3) == "TRA"])

df2[df2$chain == "TRB", "CPK"] <-
  calc.cpk(df$count[substr(df$V, 1,3) == "TRB"])
```

```r
df2[df2$chain == "TRG", "CPK"] <-
  calc.cpk(df$count[substr(df$V, 1,3) == "TRG"])

df2[df2$chain == "TRD", "CPK"] <-
  calc.cpk(df$count[substr(df$V, 1,3) == "TRD"])

# entropy
df2[df2$chain == "IGH", "Entropy"] <-
  calc.entropy(df$count[substr(df$V, 1,3) == "IGH"])

df2[df2$chain == "IGK", "Entropy"] <-
  calc.entropy(df$count[substr(df$V, 1,3) == "IGK"])

df2[df2$chain == "IGL", "Entropy"] <-
  calc.entropy(df$count[substr(df$V, 1,3) == "IGL"])

df2[df2$chain == "TRA", "Entropy"] <-
  calc.entropy(df$count[substr(df$V, 1,3) == "TRA"])

df2[df2$chain == "TRB", "Entropy"] <-
  calc.entropy(df$count[substr(df$V, 1,3) == "TRB"])

df2[df2$chain == "TRG", "Entropy"] <-
  calc.entropy(df$count[substr(df$V, 1,3) == "TRG"])

df2[df2$chain == "TRD", "Entropy"] <-
  calc.entropy(df$count[substr(df$V, 1,3) == "TRD"])

# clonality
df2[df2$chain == "IGH", "Clonality"] <-
  calc.clonality(df$count[substr(df$V, 1,3) == "IGH"])

df2[df2$chain == "IGK", "Clonality"] <-
  calc.clonality(df$count[substr(df$V, 1,3) == "IGK"])

df2[df2$chain == "IGL", "Clonality"] <-
  calc.clonality(df$count[substr(df$V, 1,3) == "IGL"])

df2[df2$chain == "TRA", "Clonality"] <-
  calc.clonality(df$count[substr(df$V, 1,3) == "TRA"])
```

```
  df2[df2$chain == "TRB", "Clonality"] <-
    calc.clonality(df$count[substr(df$V, 1,3) == "TRB"])

  df2[df2$chain == "TRG", "Clonality"] <-
    calc.clonality(df$count[substr(df$V, 1,3) == "TRG"])

  df2[df2$chain == "TRD", "Clonality"] <-
    calc.clonality(df$count[substr(df$V, 1,3) == "TRD"])


  calc_tcrbcr_clones[[nome]] <- df2
}

calc_tcrbcr_clones$`130723_UNC9-SN296_0386_BC2E4WACXX_ACTTGA_L003_report`
```

```
  chain Abundance Richness      CPK   Entropy  Clonality
1   IGH         0        0      NaN 0.0000000 1.00000000
2   IGK        16        9 562.5000 2.1006789 0.04393982
3   IGL         9        3 333.3333 0.6837389 0.37763403
4   TRA         0        0      NaN 0.0000000 1.00000000
5   TRB         5        3 600.0000 0.9502705 0.13502648
6   TRG         0        0      NaN 0.0000000 1.00000000
7   TRD         0        0      NaN 0.0000000 1.00000000
```

Salvando as tabelas e listas

```
# funcao para salvar cada dataframe individualmente
salvar_dataframes <- function(lista, dir_destino){
  for (nome_df in names(lista)) {
    arquivo <- paste0(dir_destino,"/", nome_df, ".tsv")
    write.table(lista[[nome_df]], arquivo, sep = "\t", row.names = FALSE)
  }
}

# salva os dataframes como .tsv
salvar_dataframes(calc_tcrbcr_sequencias, dir_destino = "results_pipeline_report")
salvar_dataframes(calc_tcrbcr_clones, dir_destino = "results_pipeline_clones")

# salva as listas
save(calc_tcrbcr_sequencias, file = "calc_tcrbcr_sequencias.RData")
```

```r
save(calc_tcrbcr_clones, file = "calc_tcrbcr_clones.RData")
```