Leitura de arquivos Illumina com snpStats

Paulo Barros

Arquivos *Final Report* e de Mapa

Nesta prática vamos trabalhar com dados de sequencias de um chip comercial de galinhas da *Illumina*. Um arquivo ***Final Report*** é um arquivo de texto gerado a partir do Genome Studio contendo as informações de genótipos dos indivíduos.

O *Final Report* pode conter diversos campos com informações sobre a genotipagem, aqui vamos cobrir alguns mais importantes:

Table 1: Campos de um *Final Report*

Campo	Descrição
Sample Name	Nome da amostra
SNP Name	Identificador do SNP
Allele1 Top Allele2 Top	Alelos 1 e 2 em nucleotideos correspondentes aos Alelos A e B, reportados na fita Top
Sample ID	Identificador da Amostra
GC Score	Medida de qualidade calculada para cada genótipo, varia de 0 a 1
Allele1 Forward Allele2 Forward	Alelos 1 e 2 em nucleotideos correspondentes aos Alelos A e B, reportados na fita Forward
Allele1 AB Allele2 AB	Alelos 1 e 2 em genótipos AB

Para consultar todos os campos disponíveis no *Final Report* você pode visitar o site da Illumina.

Explorando um *Final Report*

Abaixo você pode ver algumas linhas iniciais de um dos arquivos *Final Report* que serão trabalhos nesta prática.

```
[Header] # <1>
GSGT Version    2.0.5
Processing Date 10/19/2020 12:00 PM
Content     Chicken_50K_CobbCons_15000986_A.bpm
Num SNPs    57636
Total SNPs  57636
Num Samples 192
Total Samples   192
File    1 of 192
[Data] # <2>
SNP Name    Sample ID   Allele1 - AB    Allele2 - AB    GC Score
10080_COIII 203900190022_R01C01 B   B   0.6976
10668_COIII 203900190022_R01C01 B   B   0.7118
12187_ND4   203900190022_R01C01 B   B   0.6915
15466_CYTB_MT   203900190022_R01C01 B   B   0.3856
16262_ND6   203900190022_R01C01 B   B   0.8686
16370_ND6   203900190022_R01C01 B   B   0.8878
5831_ND2_MT 203900190022_R01C01 A   A   0.7214
```

① `[Header]`: As primeiras linhas de um *Final Report* nos trazem informações sobre o processo de genotipagem. Nesta seção podemos encontrar o número de SNPs, de amostras, e se o arquivo é um arquivo único ou individual (um *Final Report* por amostra).

② `[Data]`: Na seção data inicia o arquivo de fato, é possível então identificar quais colunas estão disponíveis.

> **i** Atenção!
>
> Nem sempre os *Final Reports* apresentam todos os campos necessários, para fazer a edição é necessário então ter acesso ao projeto original da genotipagem, e fazer uma nova edição para gerar os arquivos no Genome Studio da Illumina. Isso em cenários reais pode variar bastante entre empresas e projetos.

Arquivo de Mapa

É no arquivo de Mapas que encontramos as informações de identificação e da posição física de cada SNP no genoma.

Table 2: Arquivo de Mapa

Campo	Descrição
Name	Identificador do SNP
Chr	Cromossomo
Position	Posição no genoma em pares de base
GenTrain Score	Métrica de qualidade do cluster de SNP. É o escore do SNP no algoritmo de clusterização GenTrain

Abaixo você pode ver algumas linhas iniciais do arquivo de mapas que utilizaremos. Em um arquivo de mapas, cada SNP ocupa uma linha do arquivo.

```
Name    Chr Position    GenTrain Score
10080_COIII 0   0   0.7310036
10668_COIII 0   0   0.7387226
12187_ND4   0   0   0.7660004
15466_CYTB_MT   0   0   0.799299
16262_ND6   0   0   0.8815182
16370_ND6   0   0   0.8978758
5831_ND2_MT 0   0   0.7440125
8472_COXII_MT   0   0   0.7310036
Gga_rs10721746  23  5869156 0.873894
Gga_rs10721817  1   107404689   0.8605233
```

Fazendo a leitura dos genótipos com o `snpStats`

> **O nosso objetivo é fazer a leitura dos *Final Reports* para cada indivíduo e organizá-los em um único arquivo de genótipo para futuras análises.**

`snpStats`

Para esta prática vamos precisar dos pacotes `BiocManager` e `snpStats`. Você pode fazer a instalação rodando o código abaixo:

```
if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")
BiocManager::install("snpStats")
```

Após a instalação podemos então carregar os pacotes necessários:

```
library(snpStats)
library(tidyverse)
```

De maneira geral, **em um arquivo de genótipos cada linha deve representar um indivíduo** e conter além de **sua identificação**, as **colunas contendo os genótipos para cada SNP**. A depender do software utilizado para as análises genômicas, a configuração do arquivo de genótipo pode envolver mais campos e a codificação dos genótipos também pode variar.

Nesta prática, vamos construir o arquivo de genótipos em três passos:

1. Criar uma lista com os nomes de todos os *Final Reports* pra leitura.
2. Criar vetores que contenham o ID dos Animais e o ID dos SNPs.
3. Usar o pacote `snpStats` para gerar o arquivo de genótipos.

Criando a lista de *Final Reports*

Os arquivos para esta prática estão na pasta `data/01_snpStats`. Vamos usar a função `list.files` para gerar nossa lista de arquivos a serem lidos.

```
filenames <- list.files(path = "data/01_snpStats/",
           pattern = "Final_Report_.*.txt",
           full.names = TRUE)
```

Criando os vetores de ID dos SNPs

Aqui é bem simples, faremos a leitura do arquivo de mapas e vamos extrair a primeira coluna que é a de ID dos SNPs.

```
snpids <- read.table("data/01_snpStats/SNP-Table-Chickens50k.txt",
                     sep = "\t",
                     header = TRUE) |>
          pull(Name)
```

Criando os vetores de ID dos Animais

Aqui vamos utilizar a função `map` do pacote `purrr` para iterar sobre todos os nossos *Final Reports* e capturar somente a informação de ID dos animais.

```
animids<- filenames |>                                                    ①
            map(\(x){                                                     ②
              scan(x,skip=11,nlines=1,sep="\t",what="character")[2]       ③
            }) |>
            reduce(c)                                                     ④
```

① Passamos nosso vetor de `filenames` para a funçào `map`
② `(\x){}`: este trecho cria uma **função anônima** dentro do map. Isso faz com que a cada rodada da iteração `x` seja substituído pelo valor correspondente no na nossa lista de arquivos e as operações entre `{}` sejam executadas.
③ Aqui usamos a função `scan` para ler o arquivo:

- `x` : é o arquivo atual na lista de arquivos passadas para o map
- `skip`: é utilizado para pular as 11 linhas iniciais do arquivo, lembram delas?
- `nlines`: informamos que só queremos ler 1 linha do arquivo
- `[2]`: nosso arquivo possui 5 colunas, com a notação de colchetes estamos extraindo o valor da segunda coluna que contém o ID do animal

4. `reduce(c)`: como a função map retorna uma lista de ids, aqui usamos `reduce` para transformar a lista em um vetor com a função `c()` de concatenação

Agora já temos tudo que precisamos para gerar o arquivo de genótipos!

Gerando o arquivo de genótipos

A função `read.snps.long` é a responsável por receber o vetor de *Final Reports*, fazer a leitura e organizar o nosso arquivo de genótipos.

```
genodat<-read.snps.long(file=filenames,                                   ①
                        sample.id=animids,                                ②
                        snp.id=snpids,                                    ③
                        diploid=NULL,
                         ↪  fields=c(sample=2,snp=1,allele1=3,allele2=4,confidence=5),
                         ↪                                                ④
                        codes=c("A","B"),                                 ⑤
                        threshold=0.5,                                    ⑥
```

```
                        lower=TRUE,
                        skip=11,                                    ⑦
                        sep="\t",
                        verbose=TRUE,
                        in.order=FALSE,
                        every=10000)
```

① Informamos o vetor de `filenames`
② O vetor de ID dos animais `animids`
③ O vetor de ID dos SNPs `snpids`
④ Aqui informamos pra função em quais colunas do *Final Report* ela vai encontrar as informações referentes a amostra (*sample*), SNP, alelos (*allele1* e *allele2*) e o GC Score (*confidence*).
⑤ Aqui informados de que maneira os genótipos estão codificados no *Final Report*.
⑥ Definimos o valor mínimo de qualidade por amostra (*GC Score*)
⑦ Ignoramos as 11 primeiras linhas do *Final Report*

```
Reading one call per input line # <1>
   Sample id is in field 2 # <1>
   SNP id is in field 1 # <1>
   Allele 1 is in field 3 # <1>
   Allele 2 is in field 4 # <1>
   Confidence score is in field 5 # <1>
Reading SnpMatrix with 20 rows and 57636 columns # <2>
                           Cumulative totals
                     ----------------------------------
    File     Line   Accepted Rejected  No call  Skipped    File name
      20    57647    1100749    51957       14      220
 ↪  ..._Report_Chk_9.txt
1100749 genotypes successfully read # <3>
51957 genotypes were rejected due to low confidence # <3>
14 genotypes were not called # <3>
Warning message: # <4>
In read.snps.long(file = filenames, sample.id = animids, snp.id = snpids,  :
  220 lines of input file(s) were skipped
```

① Nos informa quais colunas foram utilizadas para leitura
② Nos diz quantas linhas e colunas o arquivo de gentótipo possui na primeira passagem.
③ Exibe estatísticas da leitura, como o número de genótipos lidos e rejeitados.
④ Nos informa o total de linhas que foram ignoradas na leitura.

Explorando o objeto de genótipos `genodat`

Quantos indivíduos e marcadores temos?

```
dim(genodat)
```

```
[1]    20 57636
```

O que encontramos nas linhas?

```
rownames(genodat) |>
  head()
```

```
[1] "203900190022_R01C01" "203900190022_R05C02" "203900190022_R06C01"
[4] "203900190022_R06C02" "203900190025_R01C01" "203900190025_R01C02"
```

E nas colunas?

```
colnames(genodat) |>
  head()
```

```
[1] "10080_COIII"   "10668_COIII"   "12187_ND4"     "15466_CYTB_MT"
[5] "16262_ND6"     "16370_ND6"
```

Salvando o arquivo de genótipos

Podemos agora salvar o nosso arquivo de genótipo para utilizar em análises daqui pra frente. Podemos salvá-lo tanto em formato de texto quanto no formato `Rdata`, um formato específico do R e otimizado para salvar objetos que podem ser recuperados facilmente em qualquer sessão do R.

> Lembrete!
>
> Lembre que se você não especificar o caminho o arquivo será salvo na sua pasta raiz do projeto, aquela pasta que contém o .Rproj. Se desejar salvar em algum outro subdiretório basta informar o caminho no momento de salvar o arquivo.
> Nos exemplos abaixo estamos salvando na subpasta `data/01_snpStats`

Salvando .Rdata

```
save(genodat,file="data/01_snpStats/genodat.Rdata")
```

Salvando em formato SNPMatrix

```
sampleIDs<-sprintf('%-20s',rownames(genodat))                    ①
rownames(genodat)<-sampleIDs                                     ②
write.SnpMatrix(genodat,                                         ③
                file="data/01_snpStats/genodat.txt",
                quote=F,
                row.names=T,
                col.names=F,
                na=5,
                sep="")
```

① Aqui usamos a função `sprintf` para formatar nossos nomes de IDs para que todos tenham 20 caracteres de comprimento.
② Então atualizamos os nomes das linhas do nosso arquivo `genodat` com os IDs formatados.
③ Por fim, salvamos o nosso arquivo em formato SNPMatrix. Observem que neste formato nossas colunas de genótipo não possuem separação entre elas. `sep=""`.

Explorando nosso arquivo `genodat.txt`

E essa é a cara do nosso arquivo de genótipos.

```
203900190022_R01C01 22252202212015211112111101100 ...
203900190022_R05C02 22252202210105122222111201120 ...
203900190022_R06C01 22252202211105121211211022200 ...
203900190022_R06C02 22252202222005222121202112210 ...
203900190025_R01C01 22252202102115022112220112110 ...
203900190025_R01C02 02050000202015021112200020210 ...
203900190025_R02C01 22252202211215021112200010210 ...
203900190025_R02C02 22252202112115020012200211210 ...
203900190025_R03C01 22252202202115020002210011200 ...
203900190025_R03C02 22252202222115022112200011110 ...
```

Neste formato os genótipos estão representados na forma de **dosagem de alelos**.

Genótipo	Código
AA	2
AB	1
BB	0
Missing	5

Código Completo

```
# Instalando os pacotes necessários

if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")
BiocManager::install("snpStats")

# Carregando pacotes

library(snpStats)
library(tidyverse)
library(data.table)

# Gerando vetor de Final Reports

filenames <- list.files(path = "data/01_snpStats/",
           pattern = "Final_Report_.*.txt",
           full.names = TRUE)

# Gerando vetor de SNP ids

snpids <- read.table("data/01_snpStats/SNP-Table-Chickens50k.txt",
                     sep = "\t",
                     header = TRUE) |>
          pull(Name)

# Gerando vetor de Animal ids

animids<- filenames |>
            map(\(x){
              scan(x,skip=11,nlines=1,sep="\t",what="character")[2]
            }) |>
            reduce(c)
```

```
# Criando o arquivo de genótipo

genodat<-read.snps.long(file=filenames,
                        sample.id=animids,
                        snp.id=snpids,
                        diploid=NULL,

                         ↪  fields=c(sample=2,snp=1,allele1=3,allele2=4,confidence=5),
                        codes=c("A","B"),
                        threshold=0.5,
                        lower=TRUE,
                        skip=11,
                        sep="\t",
                        verbose=TRUE,
                        in.order=FALSE,
                        every=10000)

# Salvando em formato Rdata

save(genodat,file="data/01_snpStats/genodat.Rdata")


# Salvando em formato SNPMatrix

sampleIDs<-sprintf('%-20s',rownames(genodat))
rownames(genodat)<-sampleIDs
write.SnpMatrix(genodat,
                file="data/01_snpStats/genodat.txt",
                quote=F,
                row.names=T,
                col.names=F,
                na=5,
                sep="")
```