GITC 2019 演習問題　解答編

復習問題１　UNIX 基本コマンド

1.　アプリケーション ＞ ユーティリティ ＞ ターミナル

1-1.　$ cd ~/data/1_unix

1-2.　$ pwd

1-3.　$ ls -R

2.

2-1.　$ mkdir ~/unixtest
　　$ mkdir ~/unixtest/FASTA-EX
　　$ cp sprot/*.fasta ~/unixtest/FASTA-EX

2-2.　$ ls ~/unixtest/FASTA-EX

3.　　$ cd ~/unixtest/FASTA-EX

3-1.　$ grep '^>' *

3-2.　$ grep '^>' * | less

3-3.　$ man grep　、 "/filename" と入力して検索実行

3-4.　$ grep -h '^>' * | less

FASTA ファイルの配列名を表す行頭文字と、コマンド上におけるリダイレクトはどちらも「>」である。
リダイレクトではなく文字であることを示すため '' で囲うこと。

4.

4-1.　$ cat *HUMAN.fasta

4-2.　$ cat *HUMAN.fasta > human.fasta

4-3.　$ less human.fasta

4-4.　$ cat *MOUSE.fasta > mouse.fasta
　　$ cat *DROME.fasta > drome.fasta

***5.** 以下は、~/unixtest/FASTA-EX がカレントディレクトリの場合

```
$ cd ..
$ tar zcvf FASTA-EX.tar.gz ./FASTA-EX
$ tar ztvf FASTA-EX.tar.gz
$ rm -r FASTA-EX
```

上記のようにした場合は、解凍した際に FASTA-EX ディレクトリが作成されてその下にファイル群が配置される。一方

```
$ tar zcvf ../FASTA-EX.tar.gz .
```

のように対象ディレクトリにカレントディレクトリ(.)を指定すると、解凍時にカレントディレクトリに直に大量のファイルが作成されてしまうので注意すること。

***6.** $ chmod u+rw,g-rwx,o-rwx FASTA-EX.tar.gz または、
　　$ chmod 600 FASTA-EX.tar.gz

***7.** find コマンドは検索の起点になるディレクトリパスを最初に指定し、評価式 -name の後ろにファイル名を指定することによって評価式に従ってディレクトリツリーを検索する。この時指定するファイル名にはワイルドカード等を使用したパターンマッチによる検索も可能。

```
$ find ~ -name SYYM_DROME.sprot
/Users/nibb/data/1_unix/sprot/ext/flybase/SYYM_DROME.sprot
```

復習問題２　エディタとスクリプト

１）（テキストを参照せよ）
２）（略）
３）実行結果は以下のようになる。

```
GO matchs;
4 1433T_MOUSE.sprot.tmp
```

実行できない場合、以下のようにして実行権を付与すること。

```
$ chmod +x example4.sh
```

４）　以下に一例を挙げる。
この例では最後に .tmp ファイルを rm コマンドで消去するようにしている。

```
#!/bin/sh
param=$1
grep ${param} 1433T_HUMAN.sprot > line.tmp
echo "${param} matchs;"
wc -l line.tmp
rm line.tmp
```

復習問題３Ｒ　解答

1. (a) `h <- women[,1] * 2.54`
   (b) `w <- women[,2] * 0.454`
   (c) `women2 <- data.frame(height=h, weight=w)`

2. `women2[,2] / (women2[,1]/100)^2`
値は `24.05638 23.65213 23.45671 ... 22.19570 22.26216` となる。

3. (a) 男性で喫煙歴がある人のデータを抽出し、その行数を数える。`A %in% B`（A が B の要素のうちどれかと一致する）を使う。

```
> cancer.subset <- subset(cancer, gender=="male" & smoking %in% c("former", "current"))
> nrow(cancer.subset)    （→62 個）
```

   (b) `cancer.subset` から `gene1` の発現データを抜き出してその平均値をとる。

```
> mean(cancer.subset$gene1)    （→11.44719）
```

   (c) タブ区切りテキストファイルとして保存。

```
> write.table(cancer.subset, sep="¥t", file="cancer.subset.txt")
```

4. (a) `gene1` と `gene2` の散布図を作成し、`gender` によって点を色分けする。
解 1)　x 軸、y 軸のデータを指定する一般的な書き方。`xlab, ylab` で、各軸のラベルを見やすく書き直している。

```
> plot(cancer$gene1, cancer$gene2, col=cancer$gender, xlab="gene1", ylab="gene2")
```

解 2）モデル式を使った書き方。

```
> plot(gene2 ~ gene1, cancer, col=gender)
```

(参考 1) 解 1 は `attach` を使うと以下のように簡潔に書ける。`attach` は変数名をデータフレームからとる（変数名のサーチパスに加える）ことを指示する。

```
> attach(cancer)
> plot(gene1, gene2, col=gender)
```

(参考 2)プロットの色を変更したい場合は以下のような書き方がある。

```
> color <- c("red", "blue")     # 色を 1 が赤、2 が青に設定する
> plot(gene2 ~ gene1, cancer, col=color[gender])
```

（この動作を理解するために、`color[cancer$gender]`を実行して見よ）

   (b) 散布図に回帰直線を引く。まず `lm` で線形モデルへのあてはめを行い、結果を変数 `(ex.lm)`に格納する。それを用いて `abline` で回帰直線を引く。

```
> ex.lm <- lm(gene2 ~ gene1, cancer)
> abline(ex.lm)
```

回帰直線の有意性については、`summary` で確認する。

```
> summary(ex.lm)
```

出力結果の最後の行から `p-value` は `0.3965` と有意水準より大きいので､有意ではない。

5. `gender` によってデータを分割すると、`stage` による違いが観察されなくなった。これは、

もともと観察された stage III での gene1 の発現量の減少は、stage III の患者が gene1 の発現が低い女性に偏っているためであることを示唆している。このことは

```
> subset(cancer, stage=="stage III")
```

によって確認できる。

6. (a,b) cancer から 5〜10 カラムを抜き出して変数 expr に代入し、cor を実行する。

```
> expr <- cancer[,5:10]
> cor(expr)
```

(c) 相関係数の絶対値が 0.5 以上であるかどうかは、以下のコマンドで調べられる。

```
> abs(cor(expr)) >= 0.5
      gene1 gene2 gene3 gene4 gene5 gene6
gene1  TRUE FALSE FALSE FALSE FALSE  TRUE
gene2 FALSE  TRUE  TRUE  TRUE  TRUE FALSE
gene3 FALSE  TRUE  TRUE  TRUE  TRUE FALSE
gene4 FALSE  TRUE  TRUE  TRUE  TRUE FALSE
gene5 FALSE  TRUE  TRUE  TRUE  TRUE FALSE
gene6  TRUE FALSE FALSE FALSE FALSE  TRUE
```

この結果から、相関の有無によって(gene1,gene6) と (gene2,gene3,gene4,gene5) の 2 つのグループに分けられることがわかる。
多変数間の関係を解析する多変量解析には、クラスタリングや主成分分析など様々な手法があるが、相関係数を計算することは、多くの手法においてその基盤となっている。

7. 関数は以下の通り。ただし仮引数 x は（関数内で一貫していれば）どんな名前に置き換えてもよい。

```
RMS <- function(x) {
   return( sqrt(mean(x^2)) )
}
```

RMS(1:5) の値は 3.316625 となる。

8. (b)プログラム中に 2 つある plot コマンドのうち、最初のものは対角線上（x 軸と y 軸の名前が等しい場合）、2 番目がそれ以外のプロットを作成する。
タイトルやラベルを無視すると以下の通りになる。
（左上）plot(cancer[,1])
（2 行 1 列目）plot(cancer[,2], cancer[,1])

タイトルやラベルまでつけると以下のようになる。
（左上）plot(cancer[,1], main=names(cancer[,1:4])[1])
（2 行 1 列目） plot(cancer[,2], cancer[,1], xlab=names(cancer[,1:4])[2], ylab=names(cancer[,1:4])[1])

復習問題４　NGS 基本データフォーマット

1-1)　NCBI（https://www.ncbi.nlm.nih.gov）にアクセスし、SRR073576 で検索をかけ、情報を得ることができる。シングルリードであることが分かる。

1-2)（カレントディレクトリに sra 形式のファイルをダウンロードしたい場合）

prefetch SRR073576 --output-directory .

1-3) fastq-dump SRR073576.sra

今回はシングルリードなので、--split-files オプションはなくても良い

* 大問 1 補足：

fastq-dump コマンドはアクセッション番号を指定して直接 fastq データを取ることも可能である。

ex) fastq-dump SRR073576

2-1)　第 21 染色体

2-2)　41　（ $ wc ex4.bed ）

2-3)　4

2-4)　33031813,　33025906

3-1)　bed <- read.table("ex4.bed", header=F, sep="¥t")

head(bed)

3-2)　table(bed[,10])

4-1)　10

4-2)　3

復習問題５　クオリティコントロールと NGS 基本ツール

1-1) Pairs written(passing filters) : 60,885(60.9%)

Quality-trimmed 824,581(5.4%)

1-2) less コマンドでファイルを見る

1-3) trim 前 400,000 なので read としては 4 で割って、100k read

trim 後 243,540 なので read 数は 60,885 となり、log の値と一致している。

1-4) Per base sequence quality のタブを見る

2. seqkit stats ecoli.[23].fastq

3.

```
$ bowtie2 -x eco -U ecoli.2.fastq,ecoli.3.fastq -S eco_ex.sam
```

ファイルのカンマ区切りの後にスペースを入れないこと

4.

```
$ samtools view -bS eco_ex.sam -o eco_ex.bam
```

5.

```
$ samtools sort eco_ex.bam -o eco_ex_sorted.bam
```

4.5 は現行 samtools なら以下で一括に可能

```
$ samtools sort eco_ex.sam –o eco_ex_sorted.bam
```

6.

```
$ samtools index eco_ex_sorted.bam
```

7.

```
$ samtools view eco_ex_sorted.bam chr:337-2799 | wc
```

277 リード

```
$ samtools view eco_ex_sorted.bam chr:4179268-4183296 | wc
```

1,015 リード

復習問題 6　UNIX によるテキストファイル処理

1) `$ grep '^@' ex6.sam`

2) `$ grep -v '^@' ex6.sam > ex6_2.sam`

3) `$ awk '{print $2}' ex6_2.sam`

4) `$ awk '{print $2}' ex6_2.sam | sort | uniq`

5)*　2 カラム目の値が **4** であるフラグメント名を抽出することになる

```
$ awk '$2==4{print}' ex6_2.sam | awk '{print $1}'
```

解答

実践演習 1　RNA-Seq 解析結果の集計(1)

1) ファイルが 1,2,3,...の順に表示されるのは B)の方。

```
$ paste results/ecoli.{?,1?}.htseq > ecoli.count_all.tmp1
```

2) cut で該当する列を抽出し、次に_で始まる行を除く。grep –v はマッチする行を除いて、それ以外の行を出力する。

```
$ cut –f 1,2,4,6,8,10,12,14,16,18,20,22,24 ecoli.count_all.tmp1 >
ecoli.count_all.tmp2
$ grep -v '^_' ecoli.count_all.tmp2 > ecoli.count_all
```

実践演習 2　RNA-Seq 解析結果の集計(2)

1) セパレータはタブ(sep="¥t")、ヘッダはなし(header=F)、1 列目を行の名前として読み込む(row.names=1)。

```
> eco_rna <- read.table("ecoli.count_all", sep="¥t", header=F,
row.names=1)
```

2) eco_rna の 2-13 列目を列方向にベクトルとして取り出して sum 関数を適用する。

```
> eco_rna_readsum <- apply(eco_rna, 2, sum)
```

3) eco_rna を行方向のベクトルとみて、同じ要素数のベクトル eco_rna_readsum で割る。それを 1,000,000 倍する。その後、転置行列をとる。

```
> eco_rna_cpm0 <- apply(eco_rna, 1, '/', eco_rna_readsum) * 1000000
> eco_rna_cpm <- t(eco_rna_cpm0)
```

4) apply 関数を使って関数 calc_means を eco_rna_cpm の各行に対して適用する（calc_means があらかじめ定義されていることが前提）。結果は、前問と同様に行と列が入れ替わるので、転置行列をとる。以下では、関数の適用と転置行列をとる操作を一つのコマンドにまとめている。

```
> eco_rna_mean <- t( apply(eco_rna_cpm, 1, calc_means) )
```