Introduction To R: Part Two - Solutions

Part One

1. Create a vector of the first 16 prime numbers.

```r
#library
library(primes)

#create vector
prime_nos<-generate_primes(min=0, max=55)
```

2. Use this vector to create a 4x4 matrix called `my_mat` using default parameters.

```r
#create matrix
my_mat<-matrix(prime_nos,4,4)
my_mat
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2   11   23   41
## [2,]    3   13   29   43
## [3,]    5   17   31   47
## [4,]    7   19   37   53
```

3. Add the numbers 1-4 as a new row to the matrix.

```r
#add seq 1-4 to matrix
my_mat<-rbind(my_mat,seq(1:4))
my_mat
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2   11   23   41
## [2,]    3   13   29   43
## [3,]    5   17   31   47
## [4,]    7   19   37   53
## [5,]    1    2    3    4
```

4. Replace the entry at the position 2,3 with 3.

```r
#change entry
my_mat[2,3]=3
my_mat
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2   11   23   41
## [2,]    3   13    3   43
## [3,]    5   17   31   47
## [4,]    7   19   37   53
## [5,]    1    2    3    4
```

5. Write a loop that checks if the entries of the fifth row are even. If they are print "You found me", otherwise print "Try again"

```r
#for entries of row five of my_mat
for (entry in my_mat[5,]){
  # if remainder when entry/2 is 0 --> even
  ifelse(entry%%2==0, print("You found me"),print("Try again"))
}
```

```
## [1] "Try again"
## [1] "You found me"
## [1] "Try again"
## [1] "You found me"
```

6. Remove the third row.

```r
#remove row 3
my_mat<-my_mat[-3,]
```

7. Print out the final matrix.

```r
#print mat
print(my_mat)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2   11   23   41
## [2,]    3   13    3   43
## [3,]    7   19   37   53
## [4,]    1    2    3    4
```

Part Two

You have been asked to analyse the results of a differential expression analysis. You have been supplied with `results.txt`, the output of this analysis. This includes:

- baseMean: The average of the normalised counts per gene across all samples
- log2FoldChange: Gives an idea of change in expression due to a test condition with respect to control
- lfcSE: Standard error of log2 fold change
- pval: P value, the result of a hypothesis test to test whether the change in expression can be attributed to the test condition
- padj: P value adjusted for multiple testing

1. Read in the file and assign the variable name `de`.

```r
##read in file
#watch sep!
results<-"https://raw.githubusercontent.com/mahoney-r/Tutorial/master/results.txt?token=ANSGCWY4UQEMCSX
de<-read.table(results, sep=";", header=T)
```

2. Head the first 10 rows.

```r
## head first 10
head(de,10)
```

```
##                  Gene    baseMean log2FoldChange      lfcSE     pvalue
## 1  ENSMUSG00000000001 1221.834637    -0.15668005 0.10350752 0.13010018
## 2  ENSMUSG00000000003    0.000000             NA         NA         NA
## 3  ENSMUSG00000000028   49.916704    -0.31269744 0.22403967 0.16279777
## 4  ENSMUSG00000000037   36.997848    -0.50574401 0.32658616 0.12148329
## 5  ENSMUSG00000000049    4.886867     0.45600561 0.69269668 0.51034224
## 6  ENSMUSG00000000056 1592.542030     0.08987534 0.11414093 0.43104377
## 7  ENSMUSG00000000058  845.533588     0.01029839 0.09313232 0.91195098
## 8  ENSMUSG00000000078 1271.626803    -0.06611899 0.07953884 0.40581589
## 9  ENSMUSG00000000085 1084.228656     0.23970112 0.11064683 0.03028351
## 10 ENSMUSG00000000088 3229.148877     0.14223467 0.09510309 0.13476207
##         padj
## 1  0.5675216
## 2         NA
## 3  0.6236769
## 4  0.5482040
## 5  0.9047002
## 6  0.8636636
## 7  0.9912804
## 8  0.8543908
## 9  0.2623773
## 10 0.5776027
```

3. Tail the last 8 rows of columns 2 to 3.

```
##last 8, col 2:3
tail(de[,2:3],8)
```

```
##         baseMean log2FoldChange
## 39931  1019.0217    -0.10968101
## 39941   341.4632    -0.08535274
## 39951   338.6000    -0.22491519
## 39961  8205.2044    -0.03217649
## 39971 14621.6366    -0.02662043
## 39981     0.0000             NA
## 39991   424.2634     0.12857616
## 40001  2270.7993     0.13939845
```

4. Identify the number of columns and rows.

```
dim(de)
```

```
## [1] 39629     6
```

5. Change the name of the third column to "L2FC".

```
##abbreviate col 3 name
colnames(de)[3]="L2FC"
```

6. Extract columns 1:3, rows 1:100 and save as `sliced_df`. Head the result.

```
#slice up the df
sliced_df<-de[1:100,1:3]
```

7. Are there any duplicates? How many? If their are duplicates, remove them, then set the first column as rownames and remove the first column. (Continue with dataframe from 5. `de` NOT `sliced_df`)

```
##check for duplicate
dim(de[duplicated(de),])
```

```
## [1] 3901    6
```

```
#or
length(which(duplicated(de)))
```

```
## [1] 3901
```

```r
##remove duplicate
de_rmdup <- de[!duplicated(de), ]

##double check
length(which(duplicated(de_rmdup)))
```

```
## [1] 0
```

```r
#rownames
rownames(de_rmdup)=de_rmdup[,1]
de_rmdup$Gene=NULL
```

8. Are there any missing values? If so, remove rows containing them.

```r
##check for missing
## >0 -> na's present
length(is.na(de_rmdup))
```

```
## [1] 178640
```

```r
#remove na
de_clean<-na.omit(de_rmdup)
```

9. The information regarding the following gene was mistakenly left out of the dataset, correct this mistake.
   - ENSMUSG00000039287 570.805924 -0.4648999 0.09045180 2.751007e-07 6.393878e-04

```r
##add row
de_clean<-rbind(de_clean, ENSMUSG00000039287=c(570.805924,  -0.4648999, 0.09045180, 2.751007e-07,   6.3
```

10. The Wald statistic is generated by dividing log2 fold change by lfcSE and is used to generate the p value. It is missing from this dataset. Add a column that contains a Wald Statistic for each gene and call it `stat`.

```r
##stat col
de_clean$stat=de_clean$L2FC/de_clean$lfcSE
```

11. Using both (a) summary and (b) a for loop, find the mean of each column storing the loop output in a vector.

```r
##means with summary
summary(de_clean)
```

```
##     baseMean             L2FC               lfcSE              pvalue
##  Min.   :     4.4   Min.   :-7.525888   Min.   :0.03174   Min.   :0.0000
##  1st Qu.:    66.5   1st Qu.:-0.131177   1st Qu.:0.10472   1st Qu.:0.1524
##  Median :   455.6   Median : 0.002231   Median :0.15767   Median :0.4328
##  Mean   :  1697.4   Mean   :-0.033878   Mean   :0.23167   Mean   :0.4465
##  3rd Qu.:  1547.6   3rd Qu.: 0.115133   3rd Qu.:0.28261   3rd Qu.:0.7245
##  Max.   :608181.6   Max.   : 4.737127   Max.   :3.01634   Max.   :1.0000
##       padj                stat
##  Min.   :0.0000001   Min.   :-6.845152
##  1st Qu.:0.6095321   1st Qu.:-0.761491
##  Median :0.8655831   Median : 0.015087
##  Mean   :0.7425658   Mean   : 0.005614
##  3rd Qu.:0.9656504   3rd Qu.: 0.802140
##  Max.   :0.9999918   Max.   : 6.049182
```

```r
#means with loop
mean_vec<-c()
```

```r
for (coln in 1:length(colnames(de_clean)) ) {
  mn<-mean(de_clean[,coln])
  mean_vec<-c(mean_vec,mn)
}
```

12. Print the results of 11 (b).

```r
#mean vector
mean_vec
```

```
## [1]  1.697422e+03 -3.387830e-02  2.316726e-01  4.465016e-01  7.425658e-01
## [6]  5.614222e-03
```

13.How many genes are < 0.05 in both the pval and padj columns?

```r
###significant genes
nrow(subset(de_clean, de_clean$pvalue<0.05 & de_clean$padj<0.05))
```

```
## [1] 597
```

14. Use a for loop and an if else statement to fill a new column called `Significance`. If the padj column is < 0.05 the Significance value should be "Sig" otherwise it should be "Not Sig".

```r
# for each gene
for (i in 1:length(de_clean$pvalue)) {
  #if padj < 0.05
  if (de_clean$padj[i] < 0.05 ) {
    #new col is sig
    de_clean$Signficance[i]="Sig"
  }
  #otherwise
  else{
    de_clean$Signficance[i]="Not Sig"
  }
}
```

15. There is a cleaner and easier way to do this. Repeat the exercise in 13 without loops or ifelse, except this time add values to the new column `Expression`. If the value in the `L2FC` column is > 1 or $<$ -1, the corresponding value in `Expression` should be "big_change", otherwise it should be "little_change". Hint: Read through section 4.1 - 4.5 of the tutorial for inspiration!

```r
# using []
#create col
#set all to little
de_clean$Expression="little_change"
#when l2fc is > 1 or < -1, change to big
de_clean$Expression[abs(de_clean$L2FC)>1]="big_change"
```

16. Find the dimensions of the dataframe that satisfies the following conditions: pvalue < 0.05 OR L2FC $>$ 1

```r
##rows and columns f subset
dim(de_clean[de_clean$pvalue<0.05 | de_clean$L2FC>1,])
```

```
## [1] 2575    8
```

17. Replace ENSMUSG in the rownames with MOUSE.

```r
##change rowname prefix
rownames(de_clean)<-gsub("ENSMUSG","MOUSE", rownames(de_clean))
```

18. Find the row numbers whose rownames have the following pattern: "126".

```r
##pattern match
grep("126", rownames(de_clean))
```

```
##  [1]    14   194  1217  1218  1239  1252  1253  1387  1663  2690  2691  2692
## [13]  2693  2694  2695  4735  5718  6262  6722  7225  7226  7227  7694  8115
## [25]  8388  8706  9289  9630  9996 10635 10670 10671 10672 10673 11915 12108
## [37] 12670 13595 13849 14908 14909 14910 15486 15819 16305 16306 17188 17256
## [49] 17257 17258 17259 17260 17261 17262
```

19. Select all genes whose expression values < -1 or > 1 and whose adjusted p value is < 0.05. Call the new dataframe `interesting_res`.

```r
#filter using p val and lfc
interesting_res=subset(de_clean,abs(de_clean$L2FC)>1 & de_clean$padj<0.05)
```

20. Order by adjusted p value in increasing order of signifcance. Call the new dataframe `interesting_res_ordered`.

```r
#order by p value
interesting_res_ordered <- interesting_res[order(interesting_res$padj), ]
```

21. Write out the final dataset to a comma separated file called `DE_RESULTS.csv`. See if you can read it back in without issues.

```r
write.table(interesting_res_ordered, "DE_RESULTS.csv", sep=",")
```

1. Love, M.I., Huber, W., Anders, S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2 Genome Biology 15(12):550 (2014)