Q 1a

Shell:

```
awk '$3 == "gene"&& /protein_coding/' Homo_sapiens.GRCh38.87.gtf > 1a.txt
python3 1a.py | sort -nk1
```

Python:

```
import pandas as pd
f = open('1a.txt','r')
a = []
for lines in f:
        line = lines.strip().rstrip('\n').split('\t')
        a.append(line[0])
dat = pd.Series(a)
print(dat.value_counts())
```

Result:

```
Genes        Numbers
1              2052
2              1255
3              1076
4               751
5               876
6              1045
7               906
8               676
9               781
10              732
11             1276
12             1034
13              327
14              623
15              597
16              866
17             1197
18              270
19             1470
20              544
21              233
22              438
```

Q 1b

Shell:

```
awk '$3 == "gene" && /protein_coding/' Homo_sapiens.GRCh38.87.gtf > protein_coding.txt
```

Python:

```
import numpy as np
f = open('protein_coding.txt','r')
a  = []
for lines in f:
        line  = lines.strip().rstrip('\n').split('\t')
        a.append(int(line[4])-int(line[3])+1)
print('Max: {0}'.format(np.max(a)))
print('Min: {0}'.format(np.min(a)))
print('Mean: {0}'.format(np.mean(a)))
print('Std: {0}'.format(np.std(a)))
```

Results:

```
Max: 2304997
Min: 78
Mean: 67025.36280747458
Std: 130396.57914493038
```

Q 1c

Shell:

```
awk '$3 == "exon" && /transcript_name/' Homo_sapiens.GRCh38.87.gtf | tr ';' '\t'| cut -f 11 > exon.txt
```

Python:

```
#awk '$3 == "transcript" && /protein_coding/' Homo_sapiens.GRCh38.87.gtf | t    r ';' '\t'| cut -f 
import numpy as np
import pandas as pd

f = open('exon.txt','r')


a = []
for lines in f:
        line = lines.strip().rstrip('\n').split("\"")
        a.append(line[1])
dat = pd.Series(a)
#print(dat.value_counts())
b = dat.value_counts().tolist()
print('Max: {0}'.format(np.max(b)))
print('Min: {0}'.format(np.min(b)))
print('Mean: {0}'.format(np.mean(b)))
print('Std: {0}'.format(np.std(b)))
```

Result:

```
Max: 363
Min: 1
Mean: 5.9704598943445015
Std: 6.78526409135388
```

Q 2a

```r
set.seed(100)
raw_dat = read.table('expression.txt',header = 1,row.names = 'name')
raw_dat = as.matrix(raw_dat)
dat = kmeans(raw_dat,iter.max=20,centers = 3)
dat
```

```
## K-means clustering with 3 clusters of sizes 12, 68, 20
##
## Cluster means:
##       exp_1    exp_2    exp_3    exp_4    exp_5    exp_6    exp_7    exp_8
## 1 100.62083 90.06000 80.10500 70.14500 59.87250 50.34750 39.46667 30.25833
## 2  55.16382 54.80956 55.07838 55.17368 55.00029 54.93853 54.96632 55.07059
## 3   9.96500 20.26400 30.16050 39.87400 49.95800 60.53250 70.05900 79.76000
##      exp_9   exp_10
## 1 20.59167  9.40000
## 2 54.80588 55.05941
## 3 89.76250 99.93000
##
## Clustering vector:
##   gene_1   gene_2   gene_3   gene_4   gene_5   gene_6   gene_7   gene_8
##        2        2        2        2        3        2        1        2
##   gene_9  gene_10  gene_11  gene_12  gene_13  gene_14  gene_15  gene_16
##        2        3        2        2        2        1        3        2
##  gene_17  gene_18  gene_19  gene_20  gene_21  gene_22  gene_23  gene_24
##        2        2        2        3        1        2        2        2
##  gene_25  gene_26  gene_27  gene_28  gene_29  gene_30  gene_31  gene_32
##        3        2        2        1        2        3        2        2
##  gene_33  gene_34  gene_35  gene_36  gene_37  gene_38  gene_39  gene_40
##        2        2        3        2        2        2        2        3
##  gene_41  gene_42  gene_43  gene_44  gene_45  gene_46  gene_47  gene_48
##        2        1        2        2        3        2        2        2
##  gene_49  gene_50  gene_51  gene_52  gene_53  gene_54  gene_55  gene_56
##        1        3        2        2        2        2        3        1
##  gene_57  gene_58  gene_59  gene_60  gene_61  gene_62  gene_63  gene_64
##        2        2        2        3        2        2        1        2
##  gene_65  gene_66  gene_67  gene_68  gene_69  gene_70  gene_71  gene_72
##        3        2        2        2        2        3        2        2
##  gene_73  gene_74  gene_75  gene_76  gene_77  gene_78  gene_79  gene_80
##        2        2        3        2        1        2        2        3
##  gene_81  gene_82  gene_83  gene_84  gene_85  gene_86  gene_87  gene_88
##        2        2        2        1        3        2        2        2
##  gene_89  gene_90  gene_91  gene_92  gene_93  gene_94  gene_95  gene_96
##        2        3        1        2        2        2        3        2
##  gene_97  gene_98  gene_99 gene_100
##        2        1        2        3
##
## Within cluster sum of squares by cluster:
## [1] 186.4359 993.1548 319.9382
##  (between_SS / total_SS =  99.4 %)
##
## Available components:
##
```

```
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

Determine the background expression level by taking the average.

```
apply(raw_dat,2,mean)
```

```
##   exp_1   exp_2   exp_3   exp_4   exp_5   exp_6   exp_7   exp_8   exp_9
## 51.5789 52.1305 53.0980 53.9103 54.5765 55.5064 56.1249 57.0310 57.6915
##  exp_10
## 58.5544
```

By looking at the cluster means shown before, we can see:

Cluster1:decreasing expression
Cluster3:increasing expression

Genes of different clusters are shown as below:
Decreasing: gene_7, gene_14, ... gene_98
Increasing: gene_5, gene_10, ... gene_100

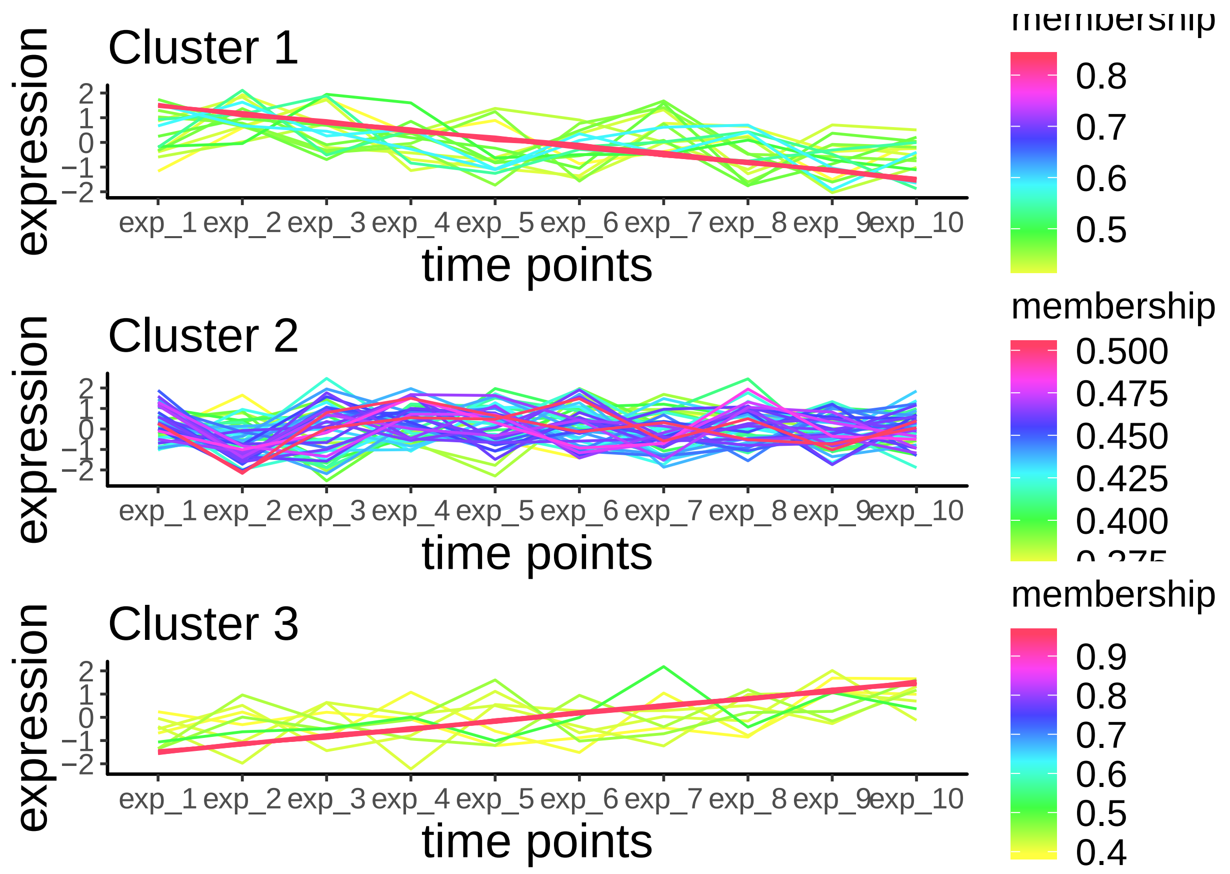
```
sort(dat$cluster)
```

```
##   gene_7  gene_14  gene_21  gene_28  gene_42  gene_49  gene_56  gene_63
##        1        1        1        1        1        1        1        1
##  gene_77  gene_84  gene_91  gene_98   gene_1   gene_2   gene_3   gene_4
##        1        1        1        1        2        2        2        2
##   gene_6   gene_8   gene_9  gene_11  gene_12  gene_13  gene_16  gene_17
##        2        2        2        2        2        2        2        2
##  gene_18  gene_19  gene_22  gene_23  gene_24  gene_26  gene_27  gene_29
##        2        2        2        2        2        2        2        2
##  gene_31  gene_32  gene_33  gene_34  gene_36  gene_37  gene_38  gene_39
##        2        2        2        2        2        2        2        2
##  gene_41  gene_43  gene_44  gene_46  gene_47  gene_48  gene_51  gene_52
##        2        2        2        2        2        2        2        2
##  gene_53  gene_54  gene_57  gene_58  gene_59  gene_61  gene_62  gene_64
##        2        2        2        2        2        2        2        2
##  gene_66  gene_67  gene_68  gene_69  gene_71  gene_72  gene_73  gene_74
##        2        2        2        2        2        2        2        2
##  gene_76  gene_78  gene_79  gene_81  gene_82  gene_83  gene_86  gene_87
##        2        2        2        2        2        2        2        2
##  gene_88  gene_89  gene_92  gene_93  gene_94  gene_96  gene_97  gene_99
##        2        2        2        2        2        2        2        2
##   gene_5  gene_10  gene_15  gene_20  gene_25  gene_30  gene_35  gene_40
##        3        3        3        3        3        3        3        3
##  gene_45  gene_50  gene_55  gene_60  gene_65  gene_70  gene_75  gene_80
##        3        3        3        3        3        3        3        3
##  gene_85  gene_90  gene_95 gene_100
##        3        3        3        3
```

Visulization:

```
library(TCseq)
set.seed(100)
raw_dat = read.table('expression.txt',header = 1,row.names = 'name')
raw_dat = as.matrix(raw_dat)
dat <- timeclust(raw_dat, algo = "cm", k = 3, standardize = TRUE,iter.max=20)
```

```
#algorithm : fuzzy c means
p <- timeclustplot(dat, categories="time points", cols = 1,axis.text.size = 11)
```



```
cluster <- clustCluster(dat)
```

Q 2b

```
pca = prcomp(t(raw_dat), scale. = TRUE)
pca$rotation[,1:2]
```

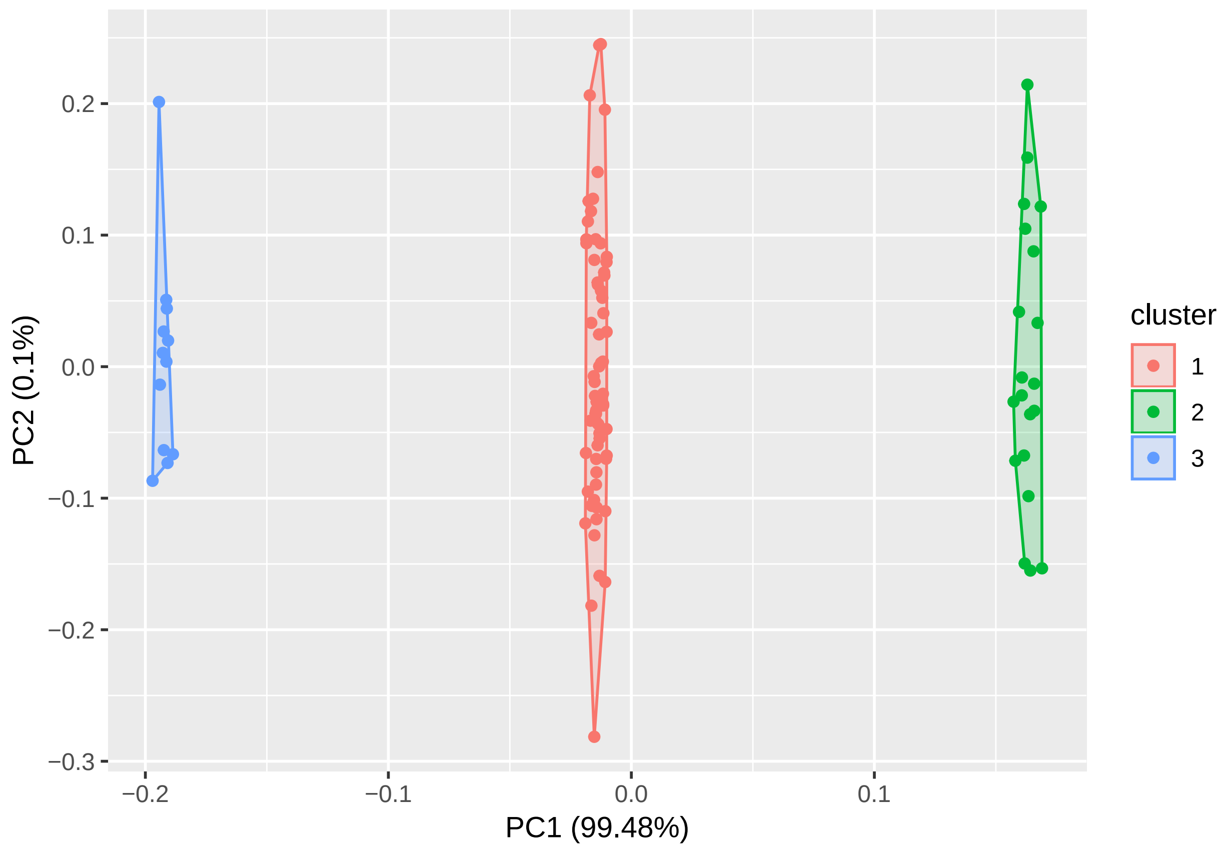
```
##                   PC1          PC2
## gene_1  -0.0047795996  0.201675099
## gene_2   0.0699050617  0.193922827
## gene_3  -0.0510177399 -0.113879577
## gene_4  -0.0477146095  0.105522342
## gene_5   0.1587807053 -0.005939070
## gene_6  -0.1077337521 -0.045827442
## gene_7  -0.1583814706  0.009232513
## gene_8   0.0705297975  0.157180068
## gene_9  -0.0131340177  0.076179789
## gene_10  0.1585624053 -0.008299127
## gene_11 -0.0887693102 -0.113331002
## gene_12 -0.0638909937 -0.174383400
## gene_13 -0.0700226892 -0.105386622
## gene_14 -0.1584288142  0.016305096
## gene_15  0.1589699092 -0.011194666
## gene_16 -0.0631413565  0.155662020
## gene_17  0.0110530422  0.067773561
## gene_18 -0.0197281962  0.085044172
## gene_19 -0.0709936570 -0.123023729
## gene_20  0.1583143185 -0.014786272
## gene_21 -0.1586699841  0.011914432
## gene_22  0.0857337998  0.208762976
## gene_23  0.0241925275  0.079425202
## gene_24  0.0032858690 -0.197233421
## gene_25  0.1588621886 -0.010418140
## gene_26  0.0388610333  0.049569126
## gene_27  0.0600651466  0.067263327
## gene_28 -0.1587031625  0.004338189
## gene_29 -0.0404195077  0.006153274
## gene_30  0.1584079030 -0.008345352
## gene_31 -0.0102770457  0.098308061
## gene_32  0.0117426006  0.066540418
## gene_33 -0.0218013316  0.130314045
## gene_34 -0.0166374451  0.142284728
## gene_35  0.1588664642 -0.012439996
## gene_36  0.0859104279  0.066361511
## gene_37 -0.0006619524 -0.204722465
## gene_38  0.0680823629 -0.099013557
## gene_39  0.0528012809  0.021170973
## gene_40  0.1589505138 -0.007154829
## gene_41  0.0690001219  0.016836838
## gene_42 -0.1587476360  0.007200572
## gene_43  0.0452146704  0.096557740
## gene_44  0.0128175747 -0.057867412
## gene_45  0.1585863253 -0.012244536
## gene_46 -0.0247735737  0.049410945
## gene_47 -0.0270554543  0.174629530
## gene_48  0.0303054684 -0.054427080
## gene_49  -0.1588572622  0.005775476
## gene_50   0.1583053287 -0.012030595
## gene_51  -0.0889245991  0.110519466
## gene_52   0.0042185934  0.097608383
## gene_53  -0.0266793232 -0.139796391
## gene_54  -0.0503329114  0.104121379
## gene_55   0.1586948196 -0.002927111
## gene_56  -0.1585603099  0.013372175
## gene_57  -0.0258445662  0.116839421
## gene_58   0.0589771430 -0.039650178
## gene_59  -0.0400502567  0.121651271
## gene_60   0.1588453334 -0.008828301
## gene_61  -0.0244834814  0.186284282
## gene_62  -0.0654471025 -0.068649284
## gene_63  -0.1586671884  0.012025664
## gene_64  -0.0463450563  0.114726428
## gene_65   0.1587022644 -0.014951534
## gene_66   0.0156527001  0.166676479
## gene_67  -0.0113497865  0.166558862
## gene_68  -0.0078035957  0.118271142
## gene_69  -0.0664610717 -0.161415775
## gene_70   0.1588391096 -0.014234701
## gene_71  -0.0967788529 -0.049453631
## gene_72   0.0667843991 -0.088550111
## gene_73   0.0125061465  0.097215793
## gene_74   0.0364971652 -0.018079600
## gene_75   0.1590755728 -0.013200717
## gene_76   0.0679763630  0.053359651
## gene_77  -0.1587982673  0.012567353
## gene_78  -0.0126192631  0.146023737
## gene_79  -0.0659976973  0.117663375
## gene_80   0.1584616683 -0.011635933
## gene_81  -0.0460409978 -0.176441104
## gene_82  -0.1089421465 -0.066331453
## gene_83  -0.0216870697  0.081500783
## gene_84  -0.1586090976  0.013978029
## gene_85   0.1588833558 -0.016791623
## gene_86   0.0889782404 -0.148204487
## gene_87   0.0330413369 -0.185777947
## gene_88  -0.0440247927 -0.167636511
## gene_89   0.0796318922  0.076715523
## gene_90   0.1586197701 -0.019630765
## gene_91  -0.1590580606  0.011059032
## gene_92   0.0509109785  0.070455645
## gene_93   0.0300880388  0.213010669
## gene_94   0.0578398941 -0.112253437
## gene_95   0.1588453530 -0.011947352
## gene_96   0.0598700955 -0.122203730
## gene_97   0.0169841517  0.039327297
## gene_98  -0.1586202429  0.017397623
## gene_99   0.0750095612  0.044227988
## gene_100  0.1588394047 -0.007947226
```

```
#screeplot(pca, type="lines",col=3)
```

```
library(tidyverse)
```

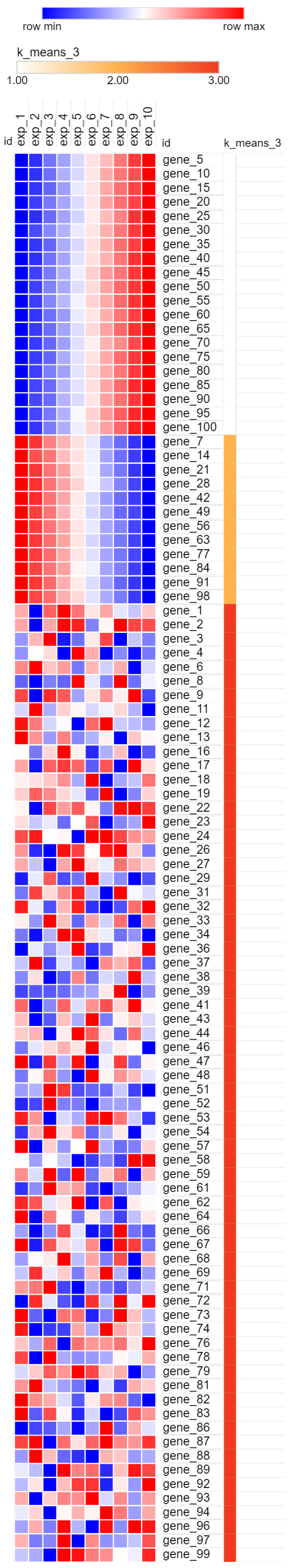
```
## -- Attaching packages ---------------------------------------------------------------------------------

## v ggplot2 3.1.0       v purrr   0.3.1
## v tibble  2.0.1       v dplyr   0.8.0.1
## v tidyr   0.8.3       v stringr 1.4.0
## v readr   1.3.1       v forcats 0.4.0

## -- Conflicts ----------------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggfortify)
library(cluster)
dftemp = merge(raw_dat,as.data.frame(dat@cluster),by =0,all.x=T) #based on cluster from part(a)
df = dftemp[,-1]
rownames(df) = c(dftemp[,1])
# library(ggfortify)
# autoplot(prcomp(df), data = df, colour = 'dat$cluster')
autoplot(fanny(df[-11], 3), frame = TRUE)
```



Q 2c

web tool: https://software.broadinstitute.org/morpheus/



row min
row max
k_means_3
1.00
2.00
3.00
id
exp_1
exp_2
exp_3
exp_4
exp_5
exp_6
exp_7
exp_8
exp_9
exp_10
id
k_means_3
gene_5
gene_10
gene_15
gene_20
gene_25
gene_30
gene_35
gene_40
gene_45
gene_50
gene_55
gene_60
gene_65
gene_70
gene_75
gene_80
gene_85
gene_90
gene_95
gene_100
gene_7
gene_14
gene_21
gene_28
gene_42
gene_49
gene_56
gene_63
gene_77
gene_84
gene_91
gene_98
gene_1
gene_2
gene_3
gene_4
gene_6
gene_8
gene_9
gene_11
gene_12
gene_13
gene_16
gene_17
gene_18
gene_19
gene_22
gene_23
gene_24
gene_26
gene_27
gene_29
gene_31
gene_32
gene_33
gene_34
gene_36
gene_37
gene_38
gene_39
gene_41
gene_43
gene_44
gene_46
gene_47
gene_48
gene_51
gene_52
gene_53
gene_54
gene_57
gene_58
gene_59
gene_61
gene_62
gene_64
gene_66
gene_67
gene_68
gene_69
gene_71
gene_72
gene_73
gene_74
gene_76
gene_78
gene_79
gene_81
gene_82
gene_83
gene_86
gene_87
gene_88
gene_89
gene_92
gene_93
gene_94
gene_96
gene_97
gene_99

Figure 1:

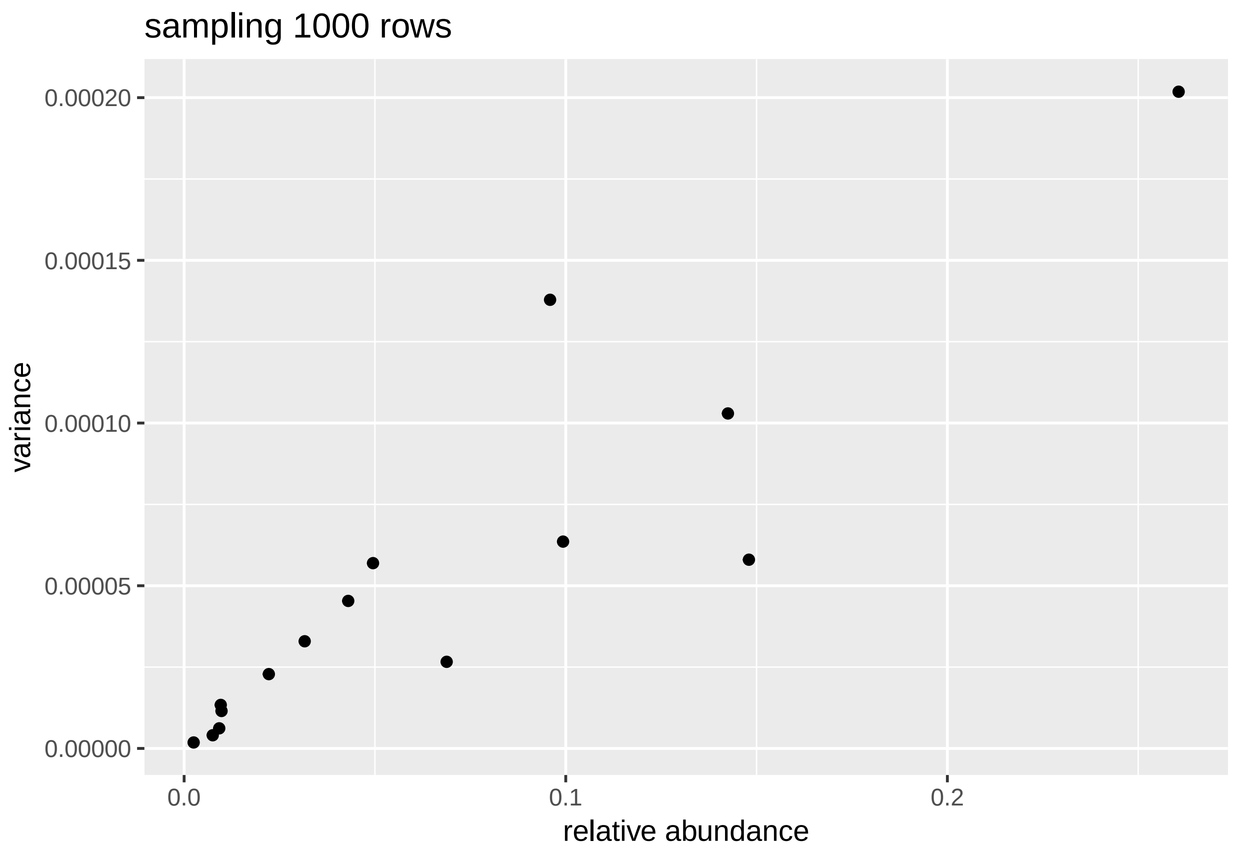
Q 3a

```r
library(dplyr)
library(ggplot2)
```

```r
set.seed(100)
n =10
raw_dat = read.table('data1.txt')
dat = matrix(0,n,15,byrow = 1)
for (i in seq(n)){
y = unlist(sample_n(raw_dat,1000))
a = table(factor(y,levels = 1:15))
dat[i,] = a
}
#relative abundance
dat/1000
```

```
##        [,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7]  [,8]  [,9] [,10] [,11]
##  [1,] 0.003 0.007 0.010 0.005 0.009 0.023 0.032 0.042 0.063 0.074 0.102
##  [2,] 0.001 0.007 0.010 0.004 0.005 0.019 0.029 0.033 0.043 0.064 0.109
##  [3,] 0.003 0.007 0.010 0.010 0.009 0.030 0.031 0.033 0.043 0.070 0.100
##  [4,] 0.001 0.007 0.008 0.017 0.011 0.016 0.034 0.046 0.057 0.065 0.113
##  [5,] 0.003 0.012 0.007 0.011 0.013 0.017 0.022 0.042 0.058 0.061 0.076
##  [6,] 0.004 0.006 0.011 0.012 0.009 0.022 0.042 0.045 0.041 0.071 0.100
##  [7,] 0.000 0.007 0.010 0.011 0.008 0.028 0.036 0.045 0.049 0.063 0.092
##  [8,] 0.003 0.005 0.018 0.009 0.006 0.021 0.032 0.041 0.043 0.076 0.093
##  [9,] 0.003 0.007 0.005 0.008 0.010 0.027 0.034 0.047 0.050 0.071 0.079
## [10,] 0.004 0.010 0.009 0.009 0.012 0.019 0.024 0.056 0.048 0.073 0.095
##       [,12] [,13] [,14] [,15]
##  [1,] 0.098 0.146 0.139 0.247
##  [2,] 0.106 0.142 0.153 0.275
##  [3,] 0.108 0.145 0.148 0.253
##  [4,] 0.097 0.144 0.135 0.249
##  [5,] 0.108 0.135 0.152 0.283
##  [6,] 0.101 0.139 0.154 0.243
##  [7,] 0.090 0.123 0.161 0.277
##  [8,] 0.103 0.148 0.144 0.258
##  [9,] 0.099 0.163 0.145 0.252
## [10,] 0.083 0.140 0.149 0.269
```

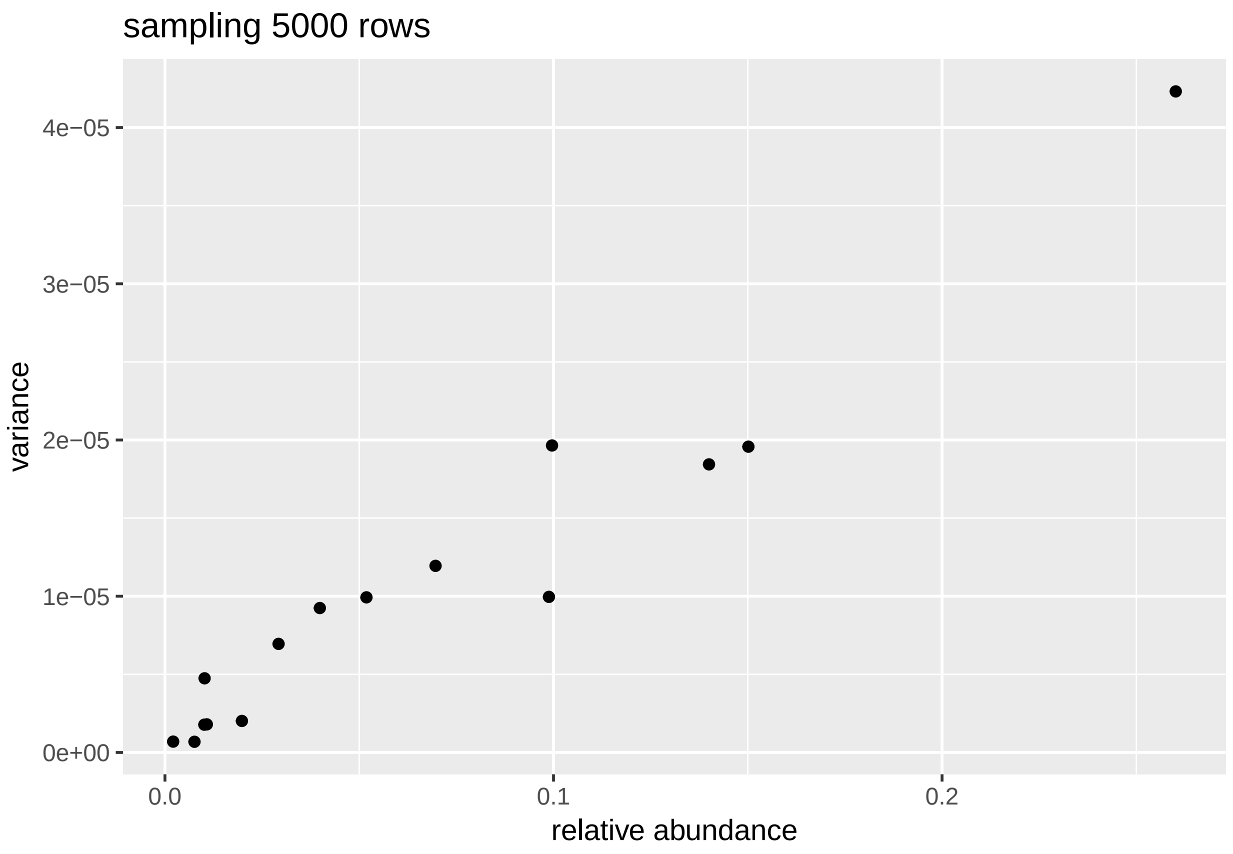
```r
dat = dat/1000
qplot(x = apply(dat, 2, mean),y = apply(dat, 2, var)) +
  labs(title = 'sampling 1000 rows',x = 'relative abundance',y='variance')
```



```r
#data.frame(apply(dat, 2, mean),apply(dat, 2, var))
```

Q 3b

```
set.seed(100)
n =10
raw_dat = read.table('data1.txt')
dat = matrix(0,n,15,byrow = 1)
for (i in seq(n)){
y = unlist(sample_n(raw_dat,5000))
a = table(factor(y,levels = 1:15))
dat[i,] = a
}
dat = dat/5000
qplot(x = apply(dat, 2, mean),y = apply(dat, 2, var)) +
  labs(title = 'sampling 5000 rows',x = 'relative abundance',y='variance')
```



```
#data.frame(apply(dat, 2, mean),apply(dat, 2, var))
```

Q 3c

The variance is greater in (a) because we sampled more times in (b) than (a), we get a better approximation of our sample in (b). Also we probably have more replicates in (b). There is a positive relationship between the variance and relative abundance. The variance becomes large when the relative abundance becomes huge.

Q 4a

```r
library(tidyverse)
library(ggplot2)
```

```r
set.seed(100)
raw_dat = read.table('data1.txt')
raw_dat2 = read.table('data2.txt')

#5 times
n = 5
dat = matrix(0,n,15,byrow = 1)
for (i in seq(n)){
y = unlist(sample_n(raw_dat,1000))
a = table(factor(y,levels = 1:15))
dat[i,] = a
}
dat = dat/1000
dat2 = matrix(0,n,15,byrow = 1)
for (i in seq(n)){
y = unlist(sample_n(raw_dat2,1000))
a = table(factor(y,levels = 1:15))
dat2[i,] = a
}
dat2 =dat2/1000
#test
r = c()
for (i in seq(dim(dat)[2])){
  if (t.test(dat[,i],dat2[,i],paired = 1)$p.value < 0.05){
    r = c(r,i)
  }
}
r
```

```
## [1] 1 3
```

gene_1 and gene_3 are significantly differentially expressed at the 0.05 level.

Q 4b

```r
#10 times
set.seed(100)
n = 10
dat = matrix(0,n,15,byrow = 1)
for (i in seq(n)){
y = unlist(sample_n(raw_dat,1000))
a = table(factor(y,levels = 1:15))
dat[i,] = a
}
dat = dat/1000
dat2 = matrix(0,n,15,byrow = 1)
for (i in seq(n)){
y = unlist(sample_n(raw_dat2,1000))
a = table(factor(y,levels = 1:15))
dat2[i,] = a
}
dat2 =dat2/1000
#reject
r = c()
for (i in seq(dim(dat)[2])){
  if (t.test(dat[,i],dat2[,i],paired = 1)$p.value < 0.05){
    r = c(r,i)
  }
}
r
```

```
## [1] 1 2 3
```

gene_1, gene_2 and gene_3 are now significant at the 0.05 level.

Q 4c

```r
#5 times
set.seed(100)
n = 5
m = 5000
dat = matrix(0,n,15,byrow = 1)
for (i in seq(n)){
y = unlist(sample_n(raw_dat,m))
a = table(factor(y,levels = 1:15))
dat[i,] = a
}
dat = dat/m
dat2 = matrix(0,n,15,byrow = 1)
for (i in seq(n)){
y = unlist(sample_n(raw_dat2,m))
a = table(factor(y,levels = 1:15))
dat2[i,] = a
}
dat2 = dat2/m
#reject
r = c()
for (i in seq(dim(dat)[2])){
  if (t.test(dat[,i],dat2[,i],paired = 1)$p.value < 0.05){
    r = c(r,i)
  }
}
r
```

```
## [1] 1 3 8
```

```r
#10 times
n = 10
m = 5000
dat = matrix(0,n,15,byrow = 1)
for (i in seq(n)){
y = unlist(sample_n(raw_dat,m))
a = table(factor(y,levels = 1:15))
dat[i,] = a
}
dat = dat/m
dat2 = matrix(0,n,15,byrow = 1)
for (i in seq(n)){
y = unlist(sample_n(raw_dat2,m))
a = table(factor(y,levels = 1:15))
dat2[i,] = a
}
dat2 = dat2/m
#reject
r = c()
for (i in seq(dim(dat)[2])){
  if (t.test(dat[,i],dat2[,i],paired = 1)$p.value < 0.05){
    r = c(r,i)
  }
}
```

```r
r
```

```
## [1]  1  2  3  8 15
```

From part a, gene_1, gene_3 and gene_8 are significant.
From part b, gene_1, gene_2, gene_3, gene_8 and gene_15 are significant.

Q 4d

```
dat1 = table(raw_dat)
dat2 = table(raw_dat2)
dat1 != dat2
```

```
## raw_dat
##     1     2     3     4     5     6     7     8     9    10    11    12
##  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE
##    13    14    15
## FALSE FALSE  TRUE
```

For actually different genes relative abundance, we got genes 1, 2, 3, 8, 12 and 15.

Additional simple simulation experiments:

```
n = 20
m = 10000
dat = matrix(0,n,15,byrow = 1)
for (i in seq(n)){
y = unlist(sample_n(raw_dat,m))
a = table(factor(y,levels = 1:15))
dat[i,] = a
}
dat = dat/m
dat2 = matrix(0,n,15,byrow = 1)
for (i in seq(n)){
y = unlist(sample_n(raw_dat2,m))
a = table(factor(y,levels = 1:15))
dat2[i,] = a
}
dat2 = dat2/m
#reject
r = c()
for (i in seq(dim(dat)[2])){
  if (t.test(dat[,i],dat2[,i],paired = 1)$p.value < 0.05){
    r = c(r,i)
  }
}
r
```

```
## [1]  1  2  3  8 12 15
```

After sampling 10,000 rows 20 times(increasing both number of rows and times), we found genes 1, 2, 3, 8, 12 and 15 are differentially exrpessed.
From part a to part c, we sampled more times than before which means we have a better approximation of our sample. Therefore, there are more diffrentially expressed genes found when we increase the sample size.