

**Note:** Please e-mail me your R program *only* (arthur.x.li@me.com) by the due date. Write *Assignment 3* and your **name** on the subject line of the e-mail.

## Assignment Format: Extra credit 1 point

Please read *AssignmentFormat.pdf* file for correct assignment format. You will be awarded with 1 point extra credit if you follow all the requirements in the *AssignmentFormat.pdf* file.

### Problem 1: 10 points

1. You need to read `chol.txt` file for this problem.
  - Use “apply” function to calculate the overall mean for each variable, except for the `sex` variable.
  - Use “aggregate” to calculate the sex-specific mean for each variable.
2. Create a new variable, `chol2`, which is based on the `chol` variable from the `chol` data frame. If `chol` is greater than the mean of `chol`, then `chol2 = 'Hi'`; otherwise, `chol2 = 'LOW'`
  - Use “tapply” function to calculate the standard deviation of `bmi` for each `chol2` category.
  - Use “tapply” function to calculate the standard deviation of `bmi` for each combination of `sex` and `chol2` categories.

### Problem 2: 10 points

Consider the simulated matrix, `mat`, with 10 rows and 20 columns.

```
> set.seed(91765)
> mat = matrix(rnorm(200), 10)
> mat[1,1] = NA
> dim(mat)
```

```
[1] 10 20
```

1. Calculate the median (use the `median` function) of each row by using the `for` loop
2. Calculate the median of each row by using the `apply` function

### Problem 3: 10 points

When performing a two-sample t-test, the data is organized into columns. However, sometimes data is organized into rows, such as genetic data. Each row contains values for each gene, and you would like to perform a two-sample t-test for each gene.

Write a function, `rowTtest`, to perform a row-wise two-sample t-test. This function will take two main arguments: the first one is a matrix or data frame with each row representing values for each gene, and with the number of columns equaling the sample size. The second argument is a character vector with length equaling sample size containing the phenotype information. The returned value is the input data frame with only those rows (genes) being statistically significant ( $p < (0.05/\# \text{ of Genes})$ , based on Bonforoni correction).

To test this function, you need to simulate a data frame or a matrix with `dim = c(1000, 20)`. Make sure to generate row names, from `G1 - G1000`. The first 10 columns are the cases and the second 10 columns are the controls. First you will start to simulate each value following standard normal distribution. Then add 4 to `G1 - G10` for only the cases. Test if you will be able to find the first 10 genes based on the function that you created. Note, you may not get those exact 10 genes, but it should be close.

Please do not use the iterative DO-loop (such as `for`, `while`, or `repeat`) inside the function.

### Here are some hint/suggestions for problem 3:

You need to simulate a matrix with dimension of 1000 by 20. For example, this matrix will look like the following:

```
> dim(gene)

[1] 1000    20

> head(gene)

      [,1]    [,2]    [,3]    [,4]    [,5]    [,6]    [,7]    [,8]
G1  3.373546  5.134965  3.113850  4.739115  2.865370  2.483627  3.381173  2.674582
G2  4.183643  5.111932  2.077745  4.386609  4.764557  4.629141  2.890578  4.951980
G3  3.164371  3.129222  5.619701  5.296397  4.570710  2.321806  1.829665  4.860004
G4  5.595281  4.210732  4.519270  3.196442  2.648306  5.179781  3.968697  5.060790
G5  4.329508  4.069396  3.944150  2.397374  1.970115  5.117655  3.739602  3.649416
G6  3.179532  2.337351  4.696418  4.933251  4.590479  2.762264  4.534430  3.869234
      [,9]    [,10]    [,11]    [,12]    [,13]    [,14]    [,15]
G1  4.263703  2.782880 -0.8043316 -1.4115219 -0.93910663  0.2264537  0.5232667
G2  3.170548  3.053771 -1.0565257  1.0838697  1.39366493 -0.8185942  0.9935537
G3  2.538365  4.091410 -1.0353958  1.1702224  1.62581486 -0.8471526  0.2737370
G4  5.683990  4.701351 -1.1855604  0.2947545  0.40900106 -1.9843326 -0.6949193
G5  2.455676  4.673422 -0.5004395 -0.5544277 -0.09255856 -0.8127788 -0.7180502
G6  3.809113  5.265553 -0.5249887 -0.4034407  0.20609871  1.4616707 -0.1019895
      [,16]    [,17]    [,18]    [,19]    [,20]
G1 -0.2139090  0.8576341  1.0496171  0.9514099 -2.07771241
G2 -0.1067233 -1.6253951  0.2903237  0.4570987 -0.45446091
G3 -0.4645893 -0.2342783  1.2421262 -0.3586935 -0.16555991
G4 -0.6842725 -1.0326545 -0.6850857 -1.0458614  0.89765209
G5 -0.7908007 -1.1411412 -0.6677681  0.3075345 -0.02948916
G6 -0.3389638 -1.5219369  0.9409138  1.9943876  1.85838843
```

Note that your simulated data will probably have different numbers. You need to provide the matrix `gene` above as the first argument of your function. You also need to create a character vector like the one below:

```
> pheno

[1] "case"    "case"    "case"    "case"    "case"    "case"    "case"
[8] "case"    "case"    "case"    "control" "control" "control" "control"
[15] "control" "control" "control" "control" "control" "control"
```

```
> length(pheno)
```

```
[1] 20
```

Within the function, you need to perform a t-test for each gene. For example, to get the p-value for the first gene, you can write the following:

```
> t.test(gene[1,] ~ pheno)$p.value
```

```
[1] 2.292288e-07
```

Similarly, to obtain the p-value for the 100th gene, you can do the following:

```
> t.test(gene[100,] ~ pheno)$p.value
```

```
[1] 0.2998438
```

In your function, you need to perform 1000 t-test and grab the p-value from each test and store these 1000 p-value to vector, named `p`. Generally we use 0.05 as a significant cutoff value. Since we are performing 1000 t-test, we need to use a Bonforoni adjusted p-value (0.05/1000). The returned value of the function is the input data frame or matrix with only those rows being significant.

## Problem 4: 10 points

Calculate the two-sample t-test for each numerical variable in the `chol` data frame (from the `chol.txt` file) by the `sex` variable by using the `apply` function. The result should be a matrix that contains five columns: F.mean (mean of the numerical variable for Female), M.mean (mean of the numerical variable for Male), t (for t-statistics), df (for degrees of freedom), and p (for p-value). The result should look like the one below:

```
> result
```

	F.mean	M.mean	t	df	p
age	22.885417	22.427083	0.20489826	187.9970	8.378733e-01
chol	202.200000	194.936842	0.75035027	188.0000	4.539818e-01
tg	79.452632	81.294737	-0.28380022	158.5931	7.769337e-01
ht	59.197187	63.659792	-3.42175696	186.0388	7.649414e-04
wt	109.510417	137.591667	-4.09786339	183.3389	6.254578e-05
sbp	117.125000	124.770833	-0.99609766	113.1632	3.213279e-01
dbp	73.312500	75.822917	-1.32775215	189.0118	1.858613e-01
vldl	15.231579	15.621053	-0.30479390	164.7068	7.609079e-01
hdl	51.400000	45.957895	3.37529526	187.9173	8.958234e-04
ldl	132.115789	131.252632	0.09293561	187.5109	9.260539e-01
bmi	2.977679	3.223266	-2.39434622	189.4748	1.762516e-02