Computer Lab 5: Two-Sample Comparisons

Complete all of the following questions, adding your inputs as code chunks (enclose within triple accent marks) within Rmarkdown.

The exercises are not marked and will not be factored into your course grade, but it is important to complete them to make sure you have the skills to answer assessment questions. You may consult any resource, including other students and the instructor. Please Knit this document to a PDF and upload your work via Canvas at the end of the session. Solutions will be posted for you to check your own answers.

Experimenting with the t and F distributions

1. Create a new function to generate t values from a two-sample difference of means of equal variance, rather than a one-sample mean. Call the new function tsimulation2 and work by copying, pasting, and modifying the code from tsimulation (find it on Canvas). Make the following changes to the copied tsimulation code to turn it into tsimulation2:

    a. Change the function name to tsimulation2.
    b. Instead of accepting a single sample size N, accept two sample sizes: Nx and Ny. (The other arguments it accepts are the same as for tsimulation.)
    c. Instead of generating a single sample of length N (stored in vector x), generate two samples: one of length Nx (stored in vector x), another of length Ny (stored in vector y).
    d. Use the two-sample, common-variance definition of t from lecture (Student's Two-sample t-test): the numerator is the difference of the two sample means; the denominator (SE) is the pooled sample variance times the square root of the harmonic mean of Nx and Ny. (See lecture slide 45.)

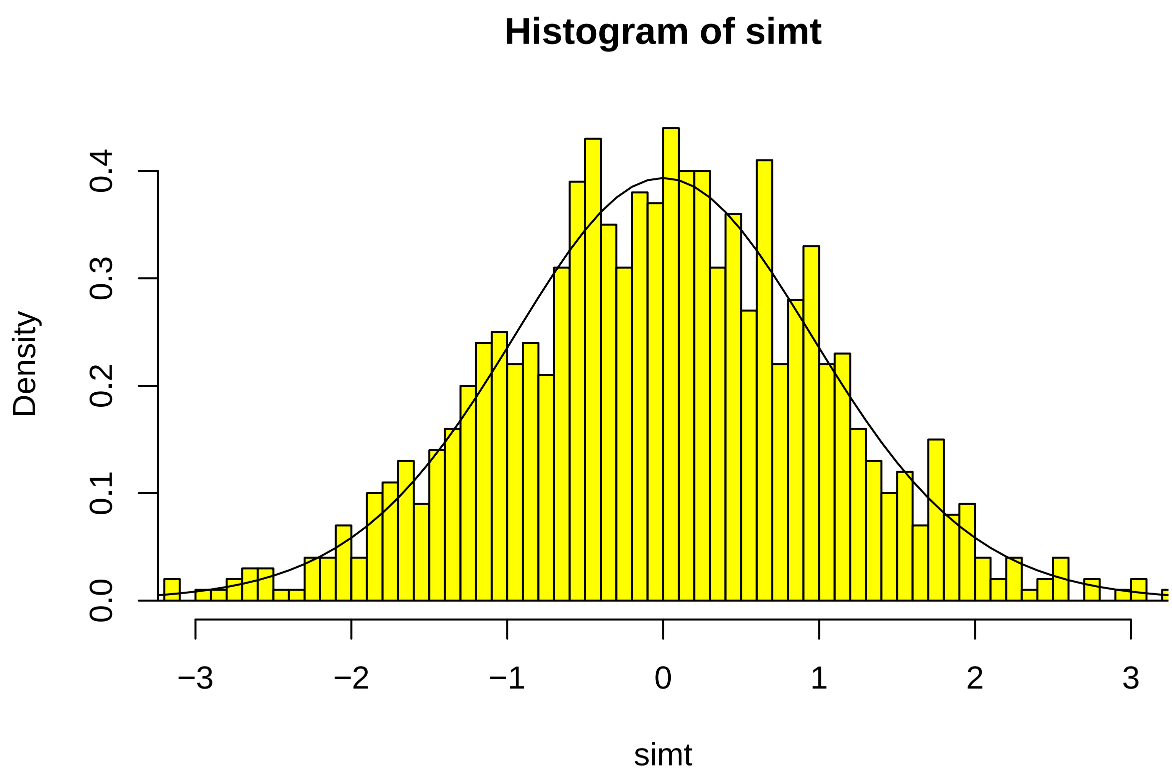
(Everything else should work the same way as for tsimulation. Note that x and y are sampled from the same distribution under the null hypothesis.)

Make sure your script compiles, and confirm that your function runs and generates a vector of length ntrials when called.

```r
tsimulation2 = function(Nx, Ny, ntrials, mu=0, sigma=1) {
  t.trials = numeric(ntrials)
  for (i in 1:ntrials) {
    x = rnorm(Nx, mean=mu, sd=sigma)
    y = rnorm(Ny, mean=mu, sd=sigma)
    diff = (mean(x)-mean(y))
    sp = sqrt(((Nx-1)*sd(x)^2+(Ny-1)*sd(y)^2)/(Nx+Ny-2))
    se=sp*sqrt(1/Nx+1/Ny)
    t = diff / se
    t.trials[i] = t
  }
  return(t.trials)
}
```

2. Verify visually that samples of t drawn using this simulation follow a Student t distribution with Nx+Ny-2 degrees of freedom. Do this by making a histogram of the results of t from an example simulation and overplotting the appropriate theoretical t distribution (use `dt` in R).

```
Nx = 10
Ny=10
simt = tsimulation2(Nx, Ny,1000)
hist(simt,breaks=seq(floor(min(simt)),ceiling(max(simt)),0.1),xlim=c(-3,3),
     freq=FALSE,col='yellow')
tplot = seq(-10,10,0.1)
lines(tplot, dt(tplot,Nx+Ny-2))
```



3. Write yet another function called fsimulation. Begin with the code from your function tsimulation2 (copy and paste to a new function). All the arguments it accepts should be the same as for tsimulation2 (Nx, Ny, mu, sigma, ntrials). Make the following changes to change the code into fsimulation:

    a. The new function name is fsimulation.
    b. Instead of calculating the two-sample t inside the loop, instead calculate F (the ratio of the two sample variances).
    c. Store it in the vector F.trials (replacing t.trials) and return F.trials at the end.

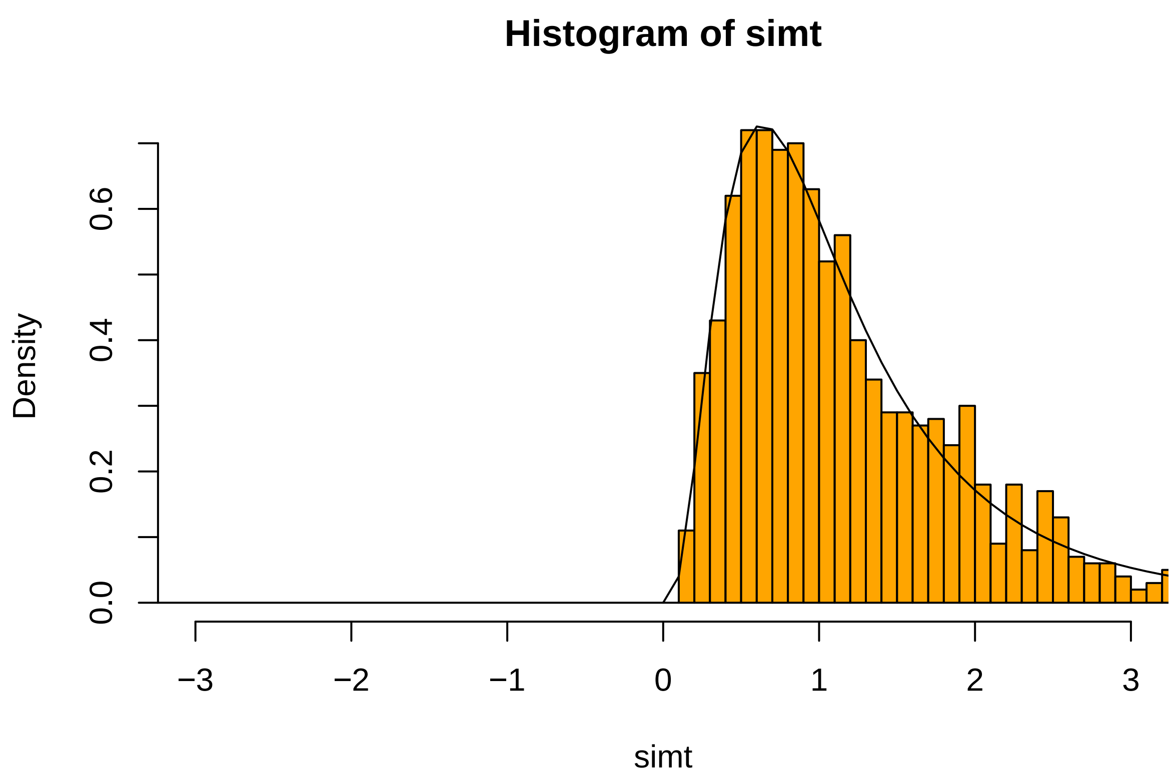
Make sure your script compiles and confirm that your function runs and generates a vector of length ntrials.

```
fsimulation = function(Nx, Ny, ntrials, mu=0, sigma=1) {
  f.trials = numeric(ntrials)
  for (i in 1:ntrials) {
    x = rnorm(Nx, mean=mu, sd=sigma)
    y = rnorm(Ny, mean=mu, sd=sigma)

    f = sd(x)^2/sd(y)^2
    f.trials[i] = f
  }
  return(f.trials)
}
```

4. Verify visually that trials of F drawn using this simulation follow an F distribution with nx-1 and ny-1 degrees of freedom by overplotting the appropriate F-distribution density function (`df` in R).

```
Nx = 10
Ny=10
simt = fsimulation(Nx, Ny,1000)
hist(simt,breaks=seq(floor(min(simt)),ceiling(max(simt)),0.1),xlim=c(-3,3),
     freq=FALSE,col='orange')
fplot = seq(-10,10,0.1)
lines(fplot, df(fplot, Nx-1, Ny-1))
```

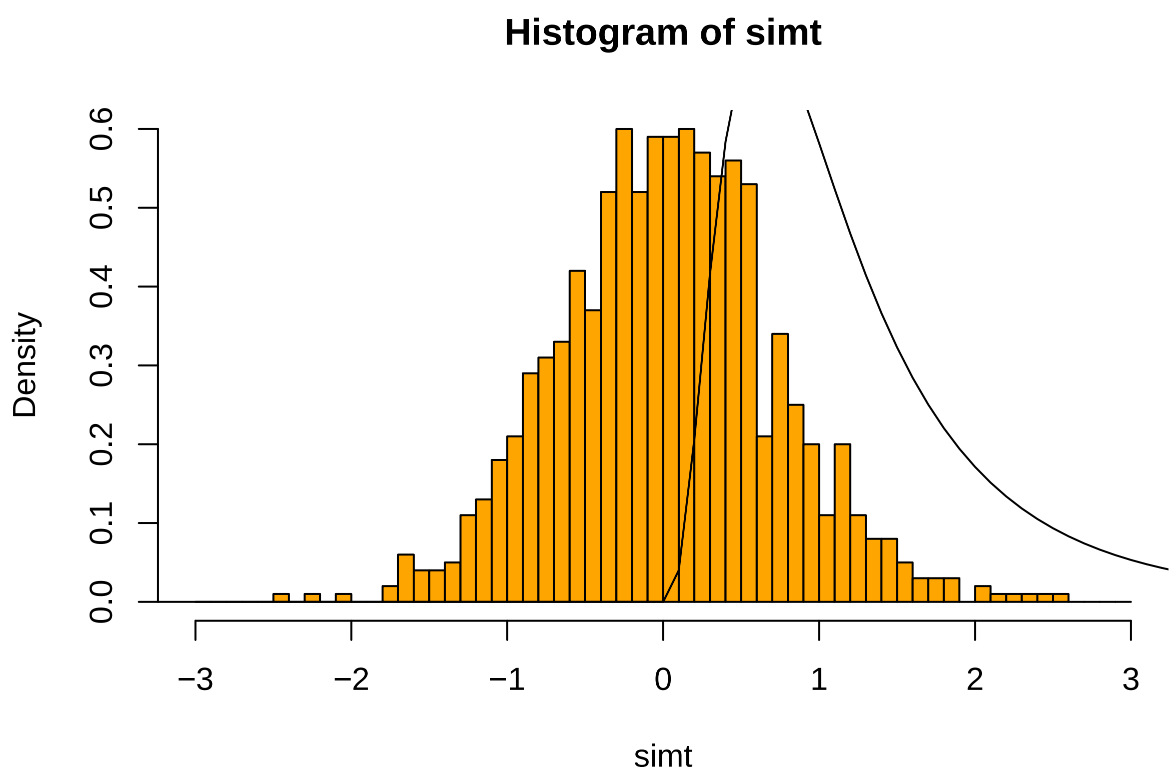


5. (Optional) Perform a logarithm transform of the trial F values from #4 (that is, create a new vector containing the logarithms of these values). Plot a histogram of these values. How does the skewness of this distribution compare to that of #4? Can you explain this? (A visual, qualitative assessment is fine.)

```
fsimulation2 = function(Nx, Ny, ntrials, mu=0, sigma=1) {
  f.trials = numeric(ntrials)
  for (i in 1:ntrials) {
    x = rnorm(Nx, mean=mu, sd=sigma)
    y = rnorm(Ny, mean=mu, sd=sigma)

    f = log(sd(x)^2/sd(y)^2)
    f.trials[i] = f
  }
  return(f.trials)
}

Nx = 10
Ny=10
simt = fsimulation2(Nx, Ny,1000)
hist(simt,breaks=seq(floor(min(simt)),ceiling(max(simt)),0.1),xlim=c(-3,3),
     freq=FALSE,col='orange')
fplot = seq(-10,10,0.1)
lines(fplot, df(fplot, Nx-1, Ny-1))
```



******

Comparing two normally-distributed samples

Suppose you are studying the effects of economic background on growth in schoolchildren. You collect data on the heights of 4th graders in two regions: a poor region and a wealthy region. These data are available as schools.csv.

6. Extract the heights of the two samples ("poor" and "wealthy" students) into separate vectors, and verify that both are consistent with a normal distribution using a Shapiro-Wilk test (`shapiro.test` in R).

```
df=read.csv('schools.csv')
df
```

```
##      region height gender
## 1   wealthy  134.3      M
## 2   wealthy  140.8      F
## 3   wealthy  139.5      M
## 4   wealthy  139.3      F
## 5   wealthy  152.3      M
## 6   wealthy  139.3      M
## 7   wealthy  136.7      M
## 8   wealthy  140.5      M
## 9   wealthy  138.7      M
## 10  wealthy  135.8      F
## 11  wealthy  148.3      M
## 12  wealthy  135.6      M
## 13  wealthy  136.6      F
## 14  wealthy  146.0      F
## 15  wealthy  139.2      M
## 16  wealthy  134.7      M
## 17  wealthy  137.4      F
## 18  wealthy  141.6      F
## 19  wealthy  147.9      F
## 20  wealthy  145.4      F
## 21  wealthy  135.8      F
## 22  wealthy  142.7      M
## 23  wealthy  142.1      F
## 24  wealthy  139.3      M
## 25  wealthy  128.7      F
## 26  wealthy  132.2      M
## 27  wealthy  139.5      M
## 28  wealthy  137.5      M
## 29  wealthy  137.7      M
## 30  wealthy  152.5      M
## 31  wealthy  141.7      F
## 32  wealthy  124.5      F
## 33  wealthy  128.9      M
## 34  wealthy  138.4      F
## 35  wealthy  136.6      F
## 36  wealthy  117.6      F
## 37  wealthy  137.6      M
## 38  wealthy  139.2      F
## 39  wealthy  134.7      F
## 40  wealthy  138.3      M
## 41  wealthy  132.7      F
## 42  wealthy  135.5      F
## 43  wealthy  138.1      M
```

```
## 44  wealthy  136.5      M
## 45  wealthy  138.5      F
## 46  wealthy  144.2      M
## 47  wealthy  151.9      F
## 48  wealthy  145.4      F
## 49  wealthy  142.8      M
## 50  wealthy  139.9      F
## 51  wealthy  140.1      F
## 52  wealthy  134.3      F
## 53  wealthy  135.6      F
## 54  wealthy  145.8      F
## 55  wealthy  150.7      F
## 56  wealthy  137.1      F
## 57  wealthy  138.4      F
## 58  wealthy  152.6      M
## 59  wealthy  140.9      M
## 60  wealthy  132.8      M
## 61  wealthy  124.4      F
## 62  wealthy  132.0      M
## 63  wealthy  129.8      M
## 64  wealthy  147.6      M
## 65  wealthy  147.4      F
## 66  wealthy  131.1      M
## 67  wealthy  133.5      F
## 68  wealthy  136.5      M
## 69  wealthy  134.2      F
## 70  wealthy  135.4      F
## 71     poor  130.9      F
## 72     poor  144.4      M
## 73     poor  144.2      M
## 74     poor  135.5      M
## 75     poor  142.2      F
## 76     poor  138.7      M
## 77     poor  134.6      M
## 78     poor  133.6      F
## 79     poor  119.1      F
## 80     poor  129.4      M
## 81     poor  137.8      F
## 82     poor  137.9      F
## 83     poor  130.2      F
## 84     poor  132.0      F
## 85     poor  129.4      F
## 86     poor  129.5      M
## 87     poor  135.4      F
## 88     poor  136.6      F
## 89     poor  144.4      M
## 90     poor  133.8      F
## 91     poor  134.3      M
## 92     poor  140.6      F
## 93     poor  140.9      M
## 94     poor  126.3      F
## 95     poor  142.3      F
## 96     poor  138.2      M
## 97     poor  131.8      M
```

```
## 98     poor  139.2      M
## 99     poor  132.3      M
## 100    poor  134.7      M
## 101    poor  144.9      M
## 102    poor  129.7      F
## 103    poor  131.1      M
## 104    poor  138.7      F
## 105    poor  137.0      F
## 106    poor  142.7      M
## 107    poor  135.8      M
## 108    poor  130.3      M
## 109    poor  138.0      M
## 110    poor  130.5      F
## 111    poor  142.3      M
## 112    poor  142.4      F
## 113    poor  148.4      F
## 114    poor  129.5      F
## 115    poor  141.4      F
## 116    poor  140.4      F
## 117    poor  130.3      M
## 118    poor  140.5      M
## 119    poor  138.3      F
## 120    poor  137.0      F
## 121    poor  132.1      M
## 122    poor  127.9      F
## 123    poor  133.9      M
## 124    poor  125.9      M
## 125    poor  133.5      F
## 126    poor  150.9      M
## 127    poor  138.6      M
## 128    poor  128.8      F
## 129    poor  123.0      F
## 130    poor  140.6      M
## 131    poor  133.9      M
## 132    poor  138.6      F
## 133    poor  129.8      M
## 134    poor  139.4      F
## 135    poor  141.0      F
## 136    poor  123.4      F
## 137    poor  141.8      M
## 138    poor  134.5      F
## 139    poor  132.8      M
## 140    poor  141.2      M
## 141    poor  138.7      M
## 142    poor  142.9      F
## 143    poor  128.6      M
## 144    poor  131.0      M
## 145    poor  136.6      F
## 146    poor  150.6      F
## 147    poor  127.9      F
## 148    poor  143.3      F
```

```r
poor_height = subset(df, region=='poor', select=height)
wealthy_height = subset(df, region=='wealthy', select=height)
```

```r
set.seed(0)
shapiro.test(poor_height$height)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  poor_height$height
## W = 0.98869, p-value = 0.7254
```

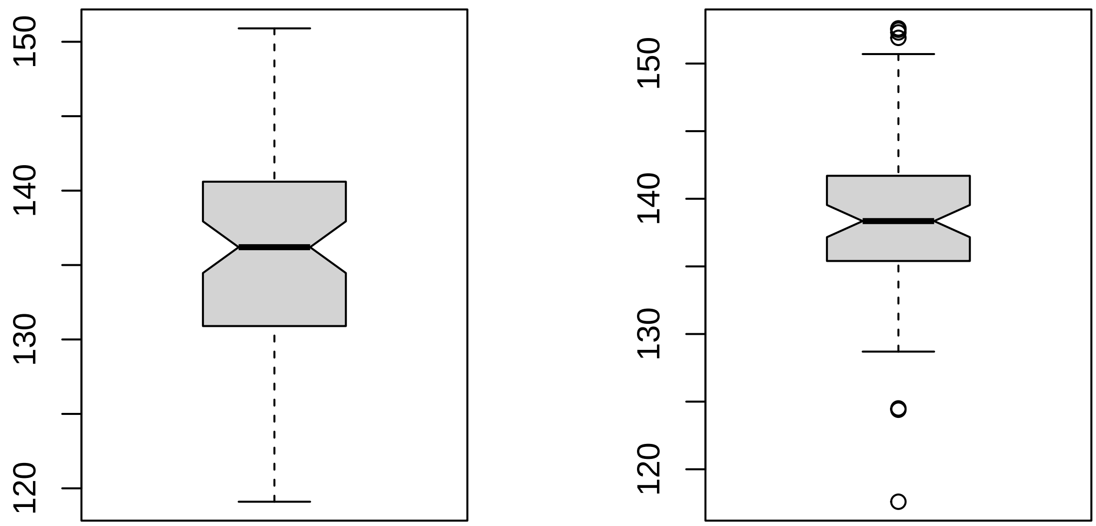
```r
shapiro.test(wealthy_height$height)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  wealthy_height$height
## W = 0.96703, p-value = 0.06192
```

7. Produce a boxplot comparing the heights of the two samples. Use the `notch=TRUE` option as a visual indicator of possible differences in means.

```r
poorh= poor_height$height
wh= wealthy_height$height
par(mfrow=c(1,2))
boxplot(poorh, notch=TRUE)

boxplot(wh, notch=TRUE)
```



8. Verify that the variances of the two samples are consistent. Do this with an F-test, doing the calculation yourself. The steps are:

   a. Compute the two variances, their ratio (F), and the degrees of freedom for both.
   b. Calculate the (two-tailed) p-value for this ratio using the F-distribution (pf in R)

```
sd1= sd(wealthy_height$height)
sd2=sd(poor_height$height)
F = sd1^2/sd2^2
df1= (length(wealthy_height$height)-1)
df2= (length(poor_height$height)-1)

2*(1-pf(F,df1,df2))
```

```
## [1] 0.657841
```

9. Use the convenience tool in R for F-tests (`var.test`) to confirm your number above.

```
var.test(wealthy_height$height, poor_height$height)
```

```
##
##  F test to compare two variances
##
## data:  wealthy_height$height and poor_height$height
```

```
## F = 1.1086, num df = 69, denom df = 77, p-value = 0.6578
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.6997481 1.7668251
## sample estimates:
## ratio of variances
##           1.108606
```

10. Check if the means are consistent using a Student's t-test, assuming equal variance. Do this **both** by calculating the t-score yourself (you can reuse the equations for sp and t from #1) and its appropriate p-value, **and** with the R convenience tool `t.test`, setting `var.equal=TRUE` to indicate that you are confident the variances are the same.

```r
poor.h= poor_height$height
wealthy.h= wealthy_height$height
mean.p=mean(poor.h)
mean.w=mean(wealthy.h)
sd.p=sd(poor.h)
sd.w=sd(wealthy.h)
n.p=length(poor.h)
n.w= length(wealthy.h)
sp = sqrt( ((n.p-1)*sd.p^2 + (n.w-1)*sd.w^2 ) / (n.p+n.w-2) )
t = (mean.p-mean.w) / (sp*sqrt(1/n.p+1/n.w))

2*(1-pt(abs(t),n.p+n.w-2))
```

```
## [1] 0.01324877
```

```r
t.test(poor_height$height, wealthy_height$height, var.equal=TRUE)
```

```
##
##  Two Sample t-test
##
## data:  poor_height$height and wealthy_height$height
## t = -2.5076, df = 146, p-value = 0.01325
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -4.7957986 -0.5682308
## sample estimates:
## mean of x mean of y
##  135.9051  138.5871
```

11. What if you weren't confident that the variances were the same? Repeat the `t.test` calculation above with `var.equal=FALSE`.

P value is same in both ways

Comparing two arbitrarily-distributed samples

The file moore.csv contains results from a study on social conformity: each of 45 subjects was paired with a partner of "high" or "low" apparent status and the extent to which each subject "conformed" with their partner's opinions was assessed. (Moore & Krupat 1971, Sociometry, 34, 122).

12. Load in the CSV file from disk. Within the data frame, the conformity score is saved as the variable "conformity" and the partner social status is saved as "partner.status". Produce a summary table of the data.

```
df_moore= read.csv('moore.csv')
summary(df_moore)
```

```
##  partner.status       conformity     fcategory            fscore     
##  Length:45          Min.   : 4.00   Length:45          Min.   :15.00  
##  Class :character   1st Qu.: 8.00   Class :character   1st Qu.:35.00  
##  Mode  :character   Median :12.00   Mode  :character   Median :43.00  
##                     Mean   :12.13                      Mean   :43.11  
##                     3rd Qu.:15.00                      3rd Qu.:55.00  
##                     Max.   :24.00                      Max.   :68.00  
```

13. Produce a box-plot of partner status (high/low) versus conformity. Does it look like there is a significant difference?

```
par(mfrow=c(1,1))
#plot(df_moore$partner.status, df_moore$conformity, xlab='partner_status', ylab='confromity')
```

The above plot function is showing error, i have checked the solution, but still not working for me

addition I run the following lines to check the na values, infinte values. howvere the plot function still not worked for me

```
any(is.infinite(df_moore$partner.status))
```

```
## [1] FALSE
```

```
any(is.infinite(df_moore$conformity))
```

```
## [1] FALSE
```

```
any(is.na(df_moore$partner.status))
```

```
## [1] FALSE
```

```
any(is.na(df_moore$conformity))
```

```
## [1] FALSE
```

```
df_moore <- na.omit(df_moore)
```

14. Check if the "conformity" scores are normal with a Shapiro-Wilk test. Examine the entire data set, as well as the two groups ("high" partner status and "low" partner status) individually.

```
shapiro.test(df_moore$conformity)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  df_moore$conformity
## W = 0.95638, p-value = 0.08882
```

```
high=df_moore$conformity[df_moore$partner.status=='high']
shapiro.test(high)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  high
## W = 0.97687, p-value = 0.8466
```

```
low=df_moore$conformity[df_moore$partner.status=='low']
shapiro.test(low)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  low
## W = 0.83806, p-value = 0.002103
```

15. Formally test whether or not there is a difference in the centres of the two distributions with a Wilcoxon Rank-Sum test. You can use the R convenience tool `wilcox.test`. (Note: you may initially get a warning message that may conceal the result—just run the task again if this happens.)

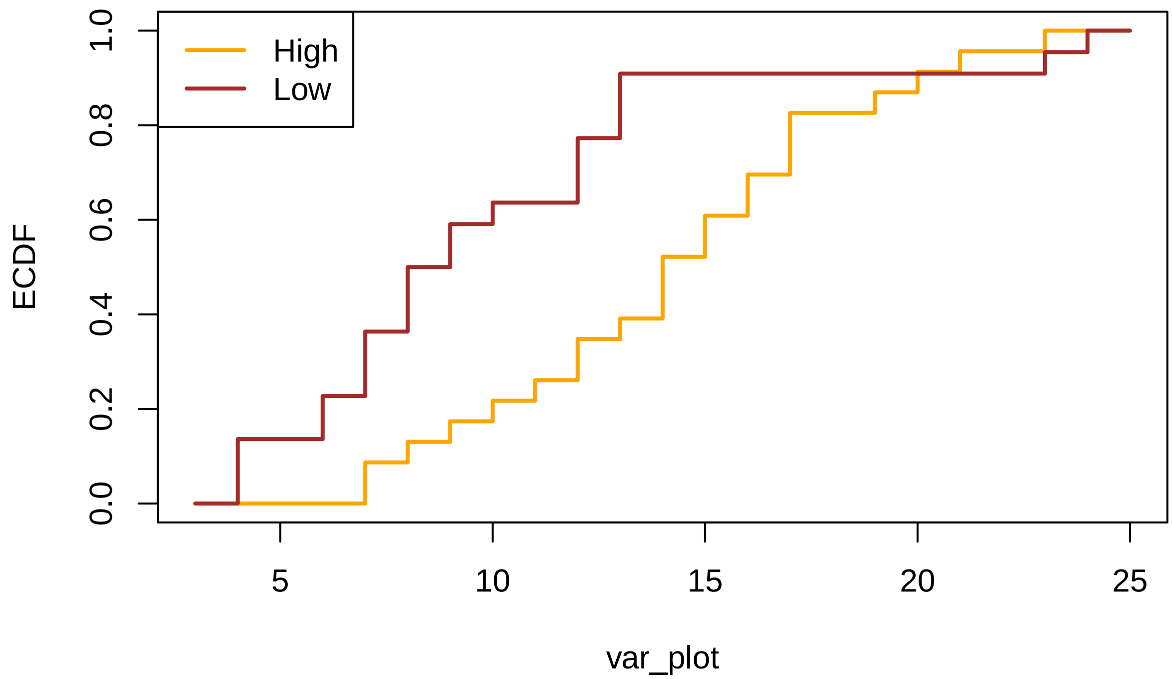
```
wilcox.test(high, low)
```

```
## Warning in wilcox.test.default(high, low): cannot compute exact p-value with
## ties
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  high and low
## W = 392, p-value = 0.001615
## alternative hypothesis: true location shift is not equal to 0
```

16. Plot the ECDFs of the two samples (put them both on the same plot in two different colours). Include a legend.

```r
xlim = c(min(df_moore$conformity)-1, max(df_moore$conformity)+1)
var_plot = c(xlim[1], sort(df_moore$conformity), xlim[2])
plot(var_plot,ecdf(high)(var_plot),typ='s',col='orange',lwd=2,ylab='ECDF',xlim=xlim)
lines(var_plot,ecdf(low)(var_plot),typ='s',col='brown',lwd=2)
legend('topleft',legend=c('High','Low'),col=c('orange','brown'),lwd=2)
```



17. Formally test whether or not there is a difference between the two distributions with a Kolmogorov-Smirnov test. You can use the R convenience tool `ks.test`. Can you confirm the value of the K-S statistic D from looking at the plot from #16?

```r
ks.test(high, low)
```

```
##
##  Exact two-sample Kolmogorov-Smirnov test
##
## data:  high and low
## D = 0.51779, p-value = 0.001195
## alternative hypothesis: two-sided
```

18. Formally test whether or not there is a difference between the two distributions with an Anderson-Darling test. You can use the R convenience tool `ad_test`. (You will have to load the R library twosamples.)

```r
#install.packages('twosamples')
library(twosamples)
ad_test(high,low)
```

```
## Test Stat   P-Value
##  527.1891    0.0015
```

Paired comparisons

Suppose 18 individuals are enrolled in a weight-loss programme. Their weights are measured before the program, and after the program. The values (in kg) are provided in the file weightloss.csv.

19. Calculate the amount of weight loss for each subject, and then calculate the single-sample t-score and corresponding p-value from the resulting difference vector (you can use the convenience function `t.test` or just do it the long way). Is the program effective (provides significant weight loss) at alpha=0.05? (Hint: given the phrasing of the question, is this a one-sided or two-sided test?)

```r
df2 = read.csv('weightloss.csv')
loss = df2$weightbefore-df2$weightafter

t.test(loss, alternative="greater")
```

```
##
##  One Sample t-test
##
## data:  loss
## t = 1.9194, df = 17, p-value = 0.03594
## alternative hypothesis: true mean is greater than 0
## 95 percent confidence interval:
##  0.1876545       Inf
## sample estimates:
## mean of x
##  2.003369
```

20. Use the convenience tool `t.test` in R to perform a paired t-test directly on the two-sample data without calculating the differences yourself. Make sure to set `paired=TRUE`, and also specify the `var.equal` and `alternative` arguments appropriately. Confirm that the result is the same as from the test on the differences.

```r
before_w = df2$weightbefore
after_w = df2$weightafter
t.test(before_w, after_w, alternative="greater", var.equal=TRUE, paired=TRUE)
```

```
##
##  Paired t-test
##
## data:  before_w and after_w
## t = 1.9194, df = 17, p-value = 0.03594
```

```
## alternative hypothesis: true mean difference is greater than 0
## 95 percent confidence interval:
##  0.1876545       Inf
## sample estimates:
## mean difference
##        2.003369
```

For comparison, see how the p-value changes for a variety of other tests:

21. Perform an unpaired Student's t-test (use `t.test` again but set `paired=FALSE`).

```r
t.test(before_w, after_w, alternative="greater", var.equal=TRUE, paired=FALSE)
```

```
##
##  Two Sample t-test
##
## data:  before_w and after_w
## t = 0.80515, df = 34, p-value = 0.2132
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  -2.203957       Inf
## sample estimates:
## mean of x mean of y
##  80.10474  78.10137
```

22. Perform an unpaired, unequal variance Welch's t-test (set `var.equal=FALSE`).

```r
t.test(before_w, after_w, alternative="greater", var.equal=FALSE, paired=FALSE)
```

```
##
##  Welch Two Sample t-test
##
## data:  before_w and after_w
## t = 0.80515, df = 31.901, p-value = 0.2133
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  -2.211722       Inf
## sample estimates:
## mean of x mean of y
##  80.10474  78.10137
```

23. Perform a Wilcoxon rank-sum (Mann-Whitney) test (use `wilcox.test` with `paired=FALSE`).

```r
wilcox.test(before_w, after_w, alternative="greater", var.equal=TRUE, paired=FALSE)
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  before_w and after_w
## W = 194, p-value = 0.1616
## alternative hypothesis: true location shift is greater than 0
```

24. Perform a Wilcoxon signed-rank test (use `wilcox.test` with `paired=TRUE`, or calculate the differences and use `wilcox.test` on the single sample of differences).

```r
wilcox.test(before_w, after_w, alternative="greater", var.equal=TRUE, paired=TRUE)
```

```
##
##  Wilcoxon signed rank exact test
##
## data:  before_w and after_w
## V = 126, p-value = 0.04071
## alternative hypothesis: true location shift is greater than 0
```

25. Perform a binomial test on the signs of differences. (Use a logical expression, then `table` in R to convert to counts. The convenience function is `binom.test`.)

```r
table((before_w-after_w) > 0)
```

```
##
## FALSE  TRUE
##     5    13
```

```r
binom.test(13,18,alternative="greater")
```

```
##
##  Exact binomial test
##
## data:  13 and 18
## number of successes = 13, number of trials = 18, p-value = 0.04813
## alternative hypothesis: true probability of success is greater than 0.5
## 95 percent confidence interval:
##  0.5021718 1.0000000
## sample estimates:
## probability of success
##              0.7222222
```

26. Review the p-values for #19-#25 above. Why do some tests provide a significant result but not others? Would you conclude that the program is effective or not?

Categorical comparisons and contingency tables

The columns colour1 and colour2 in the survey.csv file indicate the responses from this year's class regarding their colour preferences (red/green/blue for colour1 and black/white for colour2). One might wonder whether colour preferences are correlated.

27. Load in the raw data as a dataframe, and remove any lines with NA colour values if needed. Make a 2x3 contingency table from the two colour columns and store it in the variable "O", the observations matrix. (Use `table` in R.) Print it out.

```r
survey=read.csv('survey.csv',as.is=FALSE)
head(survey)
```

```
##   age height gender football_club beverage siblings av_height_est dogs
## 1  35     70   male       Chelsea     <NA>        4            60   No
## 2  26     75   male       Chelsea   coffee        3            66  Yes
## 3  25     73   male     Liverpool   coffee        1            60  Yes
## 4  23     69   male       Arsenal   coffee        1            70  Yes
## 5  23     69   male     Barcelona   coffee        1            NA  Yes
## 6  21     72   male     Liverpool   coffee        2            66  Yes
##   hometown_pop berlin_dist_est berlin_dist_unc distance fave_nums
## 1       200000           10000            5000      5.0      7,14
## 2      2000000             100              80      4.0      8,14
## 3     14000000             200              50      0.0      7,12
## 4        20000            3000             500      0.7       7,5
## 5         1000            1000             500      2.0     8,420
## 6        60000            1500             250     10.0    17,257
##   wake_time_wkday wake_time_wkend colour1 colour2
## 1           06:00           07:00    Blue   White
## 2           06:00           07:00    Blue   Black
## 3           09:00           11:30    Blue   Black
## 4           07:00           08:00     Red   White
## 5           07:00           08:30     Red   Black
## 6           08:00           12:00   Green   Black
```

```r
O = table(survey$colour1, survey$colour2)
O
```

```
## 
##          Black White
##   Blue      11     3
##   Green      3     3
##   Red        8     3
```

28. Calculate the row totals and column totals for this matrix. (Use the `apply` command in R.)

```r
apply(O,1,sum); apply(O,2,sum)
```

```
##  Blue Green   Red 
##    14     6    11
```

```
## Black White 
##    22     9
```

29. Calculate the row proportions and column proportions by dividing the answers from #28 by the sample size.

```r
n = sum(O)
apply(O,1,sum)/n
```

```
##      Blue     Green       Red
## 0.4516129 0.1935484 0.3548387
```

```r
apply(O,2,sum)/n
```

```
##     Black     White
## 0.7096774 0.2903226
```

30. Now calculate the expectation matrix E by taking the outer product of the row and column proportions (use `outer`), multiplied by n.

```r
row_frac = apply(O,1,sum)/n
col_frac = apply(O,2,sum)/n
E = outer(row_frac,col_frac)*n
E
```

```
##          Black    White
## Blue  9.935484 4.064516
## Green 4.258065 1.741935
## Red   7.806452 3.193548
```

31. Calculate the difference between observed and expected counts O-E. Do the differences suggest there might be a correlation?

```r
O-E
```

```
##
##              Black      White
##   Blue   1.0645161 -1.0645161
##   Green -1.2580645  1.2580645
##   Red    0.1935484 -0.1935484
```

32. Calculate the matrix (O-E)^2/E.

```r
(O-E)^2/E
```

```
##
##              Black      White
##   Blue  0.11405530 0.27880184
##   Green 0.37170088 0.90860215
##   Red   0.00479872 0.01173021
```

33. Calculate the sum (over all elements) of (O-E)^2/E and store the result in the new variable "chisq".

```r
chisq = sum((O-E)^2/E)
chisq
```

```
## [1] 1.689689
```

34. Calculate the degrees of freedom of this analysis. Use the formula from lecture: nu = (nrow-1)*(ncol-1). Store it in the new variable "dof".

```r
dof=(3-1)*(2-1)
```

35. Calculate the p-value using the chi-square CDF function (`pchisq`) and your answers for #33 and #34 above. (Note that chi-squared contingency tests are always one-sided, and pay attention to which side of the distribution you are on.) Is there evidence for a significant correlation between colour and shade preference?

```r
1-pchisq(chisq,dof)
```

```
## [1] 0.4296241
```

36. Use the R convenience function `chisq.test` on your observations matrix to check your answers from #33-35.

```r
chisq.test(O)
```

```
## Warning in chisq.test(O): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  O
## X-squared = 1.6897, df = 2, p-value = 0.4296
```

37. Compare this with the result from Fisher's Exact Test (`fisher.test`).

```r
fisher.test(O)
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  O
## p-value = 0.5092
## alternative hypothesis: two.sided
```