test

Christoffer Mondrup Kramer

2023-05-27

Test

Lets define helper functions and datasets up here:

Data

```
ex_3 <-read.table("Examdata3.tsv", header = FALSE)
ex_4 <- read.table("Examdata4.tsv", header = TRUE)
ex_5a <- read.table("Examdata5a.tsv", header = TRUE)
ex_5b <- read.table("Examdata5b.tsv", header = TRUE)

ex_3
```

```
##          V1       V2       V3
## 1  4.532599 4.304065 3.610918
## 2  4.543295 4.356709 3.555348
## 3  4.564348 4.382027 3.555348
## 4  4.615121 4.430817 3.663562
## 5  4.624973 4.442651 3.637586
## 6  4.634729 4.394449 3.610918
## 7  4.644391 4.418841 3.663562
## 8  4.663439 4.418841 3.663562
## 9  4.672829 4.406719 3.637586
## 10 4.718499 4.488636 3.688879
## 11 4.727388 4.477337 3.688879
## 12 4.736198 4.454347 3.688879
## 13 4.753590 4.499810 3.761200
## 14 4.762174 4.499810 3.713572
## 15 4.762174 4.510860 3.713572
## 16 4.779123 4.532599 3.713572
## 17 4.787492 4.488636 3.688879
## 18 4.787492 4.532599 3.784190
## 19 4.795791 4.553877 3.737670
## 20 4.828314 4.532599 3.806662
## 21 4.844187 4.564348 3.806662
## 22 4.852030 4.553877 3.806662
## 23 4.875197 4.553877 3.828641
## 24 4.905275 4.663439 3.850148
```

```
ex_4
```

```
##             X1           X2          X3
## 1  -0.81418033   0.514962846 0.655655027
## 2   1.82946078  14.326523624 0.479564355
## 3   4.27020877   8.557741197 0.728601310
## 4   3.27411652   5.762962762 0.002498775
## 5   3.17980151  -7.312071545 0.974144101
## 6   4.40963997  14.289056252 0.653679843
## 7   1.77956420  16.593911010 0.940950121
## 8   1.23127736  11.755569795 0.002731220
## 9   2.67942874   2.695023805 0.543744408
## 10  4.12373780   4.021139085 0.335395488
## 11 -1.48008833 -17.281575984 1.186521355
## 12  0.70486600 -14.166217670 0.096808821
## 13  0.14873099  -5.301345096 1.091196422
## 14  2.25500218   2.239294309 0.889545060
## 15  3.79730202   7.804164767 0.329150842
## 16  0.60305456 -14.758601603 0.064669794
## 17  3.24438255  13.826367967 0.466216535
## 18  0.16832633   9.759724716 1.166545282
## 19 -0.32855811  -4.086668134 1.092907294
## 20  1.55551103   5.954346700 0.175060345
## 21  2.21171144  -3.145816241 0.589181655
## 22  3.09182691  14.593059084 0.837248313
## 23  0.31998624  -0.746232622 0.987711346
## 24 -0.06171543  -6.714727777 0.080106543
## 25 -0.11738778  13.796826853 0.755393811
## 26  0.89193999 -14.052950405 0.444903895
## 27  3.76345571  11.235530216 0.817542689
## 28  4.87997165  10.391013307 0.476246890
## 29  1.06308601  -0.409708958 1.037563193
## 30  1.32538501   0.630109874 0.239500042
## 31 -2.30842188 -10.492949707 0.543512623
## 32  2.98166886   9.258008848 0.911201248
## 33  3.24443195   3.197443228 0.880340469
## 34  1.09707804   4.178967500 0.552181325
## 35  1.12891410 -10.990293534 0.227971007
## 36  0.49627775  -4.305705353 0.574749121
## 37  2.74370181  -0.514593109 1.114604142
## 38  2.88636916  14.335752749 1.100912003
## 39  1.50030219   7.731262382 0.484830644
## 40 -0.79800467  -6.058919824 0.188487823
## 41  1.67284809  -3.882386513 0.420258961
## 42 -0.41596063   0.312420271 0.327853477
## 43  0.86829933   2.798729446 0.845620260
## 44  2.63507649  -0.882570921 0.830425308
## 45 -0.35049017 -10.424461420 0.588912133
## 46  0.72529704   2.212525279 0.174624690
## 47 -5.20788414  -3.721534457 0.985940231
## 48  0.70065946  -7.160937667 0.901260515
## 49  2.95542345   9.133757381 0.391984235
## 50  2.01877959  -7.988417697 0.619616079
## 51 -1.77762952 -16.407923307 0.616914896
## 52  0.98566354  -4.791852550 1.059158415
## 53  0.71286970  -1.172828247 0.156581601
## 54 -1.26739144  -5.550791414 0.354342931
```

```
## 55  -1.24299712   0.996596787 0.598670644
## 56  -1.43459693  -8.563539512 0.951043348
## 57   0.75958559   2.888078607 0.450536533
## 58   2.52734800   3.723267699 0.510326946
## 59   1.78933530  -7.842703876 0.204158786
## 60   1.95296751   7.775520841 1.096899457
## 61  -1.49948399   5.748945344 0.100716540
## 62  -1.46573582  -0.626737220 0.407983214
## 63   2.20017896   6.502474733 1.158688579
## 64  -0.53059705  -9.065640574 0.367711201
## 65   2.40599684   1.856456203 0.541685939
## 66   1.62448744   5.561327278 0.222442846
## 67   4.98872543  12.348869744 0.840475438
## 68   0.91534362   8.670892817 0.414532371
## 69   0.46093418   0.781468276 0.608722818
## 70   3.36935989  17.275823142 0.278303142
## 71   1.34089690   1.949000850 0.747435227
## 72   2.38958171  -5.623336398 0.126708393
## 73   0.82973395   0.391956538 0.518495199
## 74   1.00701241  -6.303681537 0.428950721
## 75   2.47346326  23.024180418 0.193112335
## 76  -0.95432473  -5.137392075 0.457028308
## 77   1.76965219   0.988548376 0.593689963
## 78  -1.80497783 -14.941323296 0.918037680
## 79   2.24745700   6.255188816 0.390236035
## 80  -4.58474420 -18.950107479 1.197602203
## 81   0.57352226  -7.443649412 0.311953599
## 82  -2.62989073 -11.075059475 0.425118967
## 83  -1.35349214   7.406710184 0.302734283
## 84   4.03420821   1.648243238 1.082468619
## 85   1.40256177   1.781276451 0.322745223
## 86   3.55927510  18.222943598 0.112383114
## 87   1.15452474   3.303915646 0.438449941
## 88   3.92066471  11.537725603 0.373633710
## 89   3.09003526   7.686760990 0.986664351
## 90   4.15260640  11.849499409 0.130610463
## 91   0.22286709  -2.561254796 0.969933963
## 92   0.11227971   1.228532925 0.684978971
## 93  -0.72217398  -0.274890230 0.126980106
## 94   2.60586536   9.694254686 0.717996639
## 95  -1.65857979   1.679129168 0.179961037
## 96  -0.84739905  -3.383965250 0.248722169
## 97   0.07073406  -1.520442150 1.063425783
## 98   2.66678838   6.677659419 0.307308543
## 99   2.12234043   4.343898465 1.073086387
## 100 -2.17732616 -16.014946327 0.185117965
## 101  3.77061594  14.753175024 0.824558757
## 102  2.13247508   2.723733021 0.662672377
## 103  2.10870848   7.763080942 0.313206958
## 104 -0.62951850   5.180018466 1.174572723
## 105 -0.65713481 -11.652654576 0.301084039
## 106 -0.84372205  -1.228243270 0.509786999
## 107  0.85283713   7.675197660 1.015776030
## 108 -1.17337138  -0.672595458 0.190335187
## 109  3.03039090  18.274674905 0.914678284
## 110 -3.47684917   5.433132531 1.183457731
```

```
## 111  1.66049167  -5.567900639 1.188115004
## 112  0.69139327  -5.753517294 0.033178803
## 113 -0.16090747   0.359505832 1.002037232
## 114  1.72234271   6.166932809 0.262990118
## 115  2.28668426   6.753266954 0.142623186
## 116  2.23008854   1.477207768 0.459447501
## 117 -0.16133755   2.796916351 0.951208493
## 118 -2.14584215  -2.890247500 0.980064143
## 119  0.65665226  -8.142070608 0.095034751
## 120  3.77952714  12.667044631 0.643267987
## 121  0.87102495  12.439649768 1.101575229
## 122  0.14121376   8.726255357 0.713242045
## 123 -1.93726941  -8.440834926 0.524159139
## 124  3.16146126  -5.368927757 0.196500630
## 125  1.22998249   8.773327434 0.313103334
## 126  5.15653148  -0.126187751 1.002725313
## 127 -0.13303560  -1.379014417 0.467534748
## 128  1.42687539   9.261579310 1.093939866
## 129  3.44863505  14.076649848 0.702375636
## 130 -0.10947992   5.248044404 1.093466986
## 131 -1.31118048  -1.023940149 0.927140144
## 132  1.85817534  -2.299512659 1.002871299
## 133  0.96580978   6.753258325 0.322983111
## 134  0.87596055  -0.007017233 0.767239361
## 135  1.79243011   6.966791264 0.221504653
## 136  1.37388403   6.515432476 0.007535910
## 137 -0.68512337  -7.315413663 1.072410196
## 138 -0.45770338  -4.805697445 0.036384808
## 139  1.59247821  -2.333403079 1.192687264
## 140  2.09117945   4.070519158 0.815100538
## 141  2.13986768  -1.085227654 0.962938248
## 142  2.69794959   5.936897564 0.781039323
## 143 -1.34136351  -9.508215742 0.149562352
## 144 -1.45710656  -2.548307491 0.808759493
## 145  3.02637431   5.932015060 0.365919350
## 146 -3.17343161 -12.663258386 0.170654310
## 147  0.01851895  -4.611358659 0.650966295
## 148  4.92673747   6.688940027 0.712739542
## 149  1.12763338   1.054385985 0.265956835
## 150  3.25496625  10.328898287 0.411723397
```

```
ex_5a
```

```
##          x1      x2
## 1  -0.0056 -0.1657
## 2  -0.1698 -0.1585
## 3  -0.3469 -0.1879
## 4  -0.0894  0.0064
## 5  -0.1679  0.0713
## 6  -0.0836  0.0106
## 7  -0.1979 -0.0005
## 8  -0.0762  0.0392
## 9  -0.1913 -0.2123
## 10 -0.1092 -0.1190
## 11 -0.5268 -0.4773
## 12 -0.0842  0.0248
## 13 -0.0225 -0.0580
## 14  0.0084  0.0782
## 15 -0.1827 -0.1138
## 16  0.1237  0.2140
## 17 -0.4702 -0.3099
## 18 -0.1519 -0.0686
## 19  0.0006 -0.1153
## 20 -0.2015 -0.0498
## 21 -0.1932 -0.2293
## 22  0.1507  0.0933
## 23 -0.1259 -0.0669
## 24 -0.1551 -0.1232
## 25 -0.1952 -0.1007
## 26  0.0291  0.0442
## 27 -0.2280 -0.1710
## 28 -0.0997 -0.0733
## 29 -0.1972 -0.0607
## 30 -0.0867 -0.0560
```

```
ex_5b
```

```
##          x1      x2
## 1  -0.3478  0.1151
## 2  -0.3618 -0.2008
## 3  -0.4986 -0.0860
## 4  -0.5015 -0.2984
## 5  -0.1326  0.0097
## 6  -0.6911 -0.3390
## 7  -0.3608  0.1237
## 8  -0.4535 -0.1682
## 9  -0.3479 -0.1721
## 10 -0.3539  0.0722
## 11 -0.4719 -0.1079
## 12 -0.3610 -0.0399
## 13 -0.3226  0.1670
## 14 -0.4319 -0.0687
## 15 -0.2734 -0.0020
## 16 -0.5573  0.0548
## 17 -0.3755 -0.1865
## 18 -0.4950 -0.0153
## 19 -0.5107 -0.2483
## 20 -0.1652  0.2132
## 21 -0.2447 -0.0407
## 22 -0.4232 -0.0998
## 23 -0.2375  0.2876
## 24 -0.2205  0.0046
## 25 -0.2154 -0.0219
## 26 -0.3447  0.0097
## 27 -0.2540 -0.0573
## 28 -0.3778 -0.2682
## 29 -0.4046 -0.1162
## 30 -0.0639  0.1569
## 31 -0.3351 -0.1368
## 32 -0.0149  0.1539
## 33 -0.0312  0.1400
## 34 -0.1740 -0.0776
## 35 -0.1416  0.1642
## 36 -0.1508  0.1137
## 37 -0.0964  0.0531
## 38 -0.2642  0.0867
## 39 -0.0234  0.0804
## 40 -0.3352  0.0875
## 41 -0.1878  0.2510
## 42 -0.1744  0.1892
## 43 -0.4055 -0.2418
## 44 -0.2444  0.1614
## 45 -0.4784  0.0282
```

Helper functions

```
test_norm <- function(data_vector, name = " ", signigicance = 0.05, rounding = 4) {
  # You can chose the following significance levels
  # 0.01
  # 0.05
  # 0.10

  if (signigicance == 0.01){
    signigicance_col <- 2
  }

  else if (signigicance == 0.05){
    signigicance_col <- 3
  }

  else if (signigicance == 0.1){
    signigicance_col <- 4
  }


  # ----- QQ plot -----
  qq <- qqnorm(data_vector, plot.it = F)

  # ----- Hypothesis test ------
  # Create Testing table
  n <- c(5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 75, 100, 150, 200, 300)
  one <- c(0.8299, 0.8801, 0.9126, 0.9269, 0.9410,
        0.9479, 0.9538, 0.9599, 0.9632, 0.9671,
        0.9695, 0.9720, 0.9771, 0.9822, 0.9879, 0.9905, 0.9935)

  five <- c(0.8788, 0.9198, 0.9389,
        0.9508, 0.9591, 0.9652,
        0.9682, 0.9726, 0.9749,
        0.9768, 0.9787, 0.9801,
        0.9838, 0.9873, 0.9913, 0.9931, 0.9953)

  ten <- c(0.9032, 0.9351, 0.9503,
        0.9604, 0.9665, 0.9715,
        0.9740, 0.9771, 0.9792,
        0.9809, 0.9822, 0.9836,
        0.9866, 0.9895, 0.9928, 0.9942, 0.9960)
  testing_tbl <- data.frame(
  n,
  one,
  five,
  ten
  )

  # Find index of testing (n)
  sample_size <- length(data_vector)
  i <- 1
  prev_value = NaN
  for (n in testing_tbl$n){
```

```
    if (sample_size > testing_tbl$n[length(testing_tbl$n)]){
      i <- length(testing_tbl$n)
      break
    }

    if (n == sample_size){
      exact_sample_size <- TRUE
      break
    }

    else if ( n > sample_size & prev_value < sample_size){
      break
    }
    i <- i + 1
    prev_value <- n
  }

  # ----- Normal -------
  cor_coef <- cor(qq$x, qq$y)
  normality = FALSE
  if (cor_coef >= testing_tbl[i, signigicance_col]){
    plot(qq, xlab = "Theoretical quantiles", ylab = "Sample quantiles", main = paste0(name, "
QQ plot"))
    qqline(data_vector)
    legend("topleft",
           paste0("Rq: ", round(cor_coef, rounding), "\n",
                  "Test rq: ", testing_tbl[i, signigicance_col], "\n",
                  "Alpha: ", signigicance, "\n",
                  "n: ", n, "\n",
                  "NORMAL!"),
           cex = 0.65,
           bty = "n")
    normality = TRUE
    return(normality)
  }

  # ----- Not normal -----
  else {

    plot(qq, xlab = "Theoretical quantiles", ylab = "Sample quantiles", main = paste0(name, "
QQ plot"))
    qqline(data_vector)
    legend("topleft",
           paste0("Rq: ", round(cor_coef, rounding), "\n",
                  "Test rq: ", testing_tbl[i, signigicance_col], "\n",
                  "Alpha: ", signigicance, "\n",
                  "n: ", n, "\n",
                  "Not NORMAL!"),
           cex = 0.65,
           bty = "n")
  }
}
```

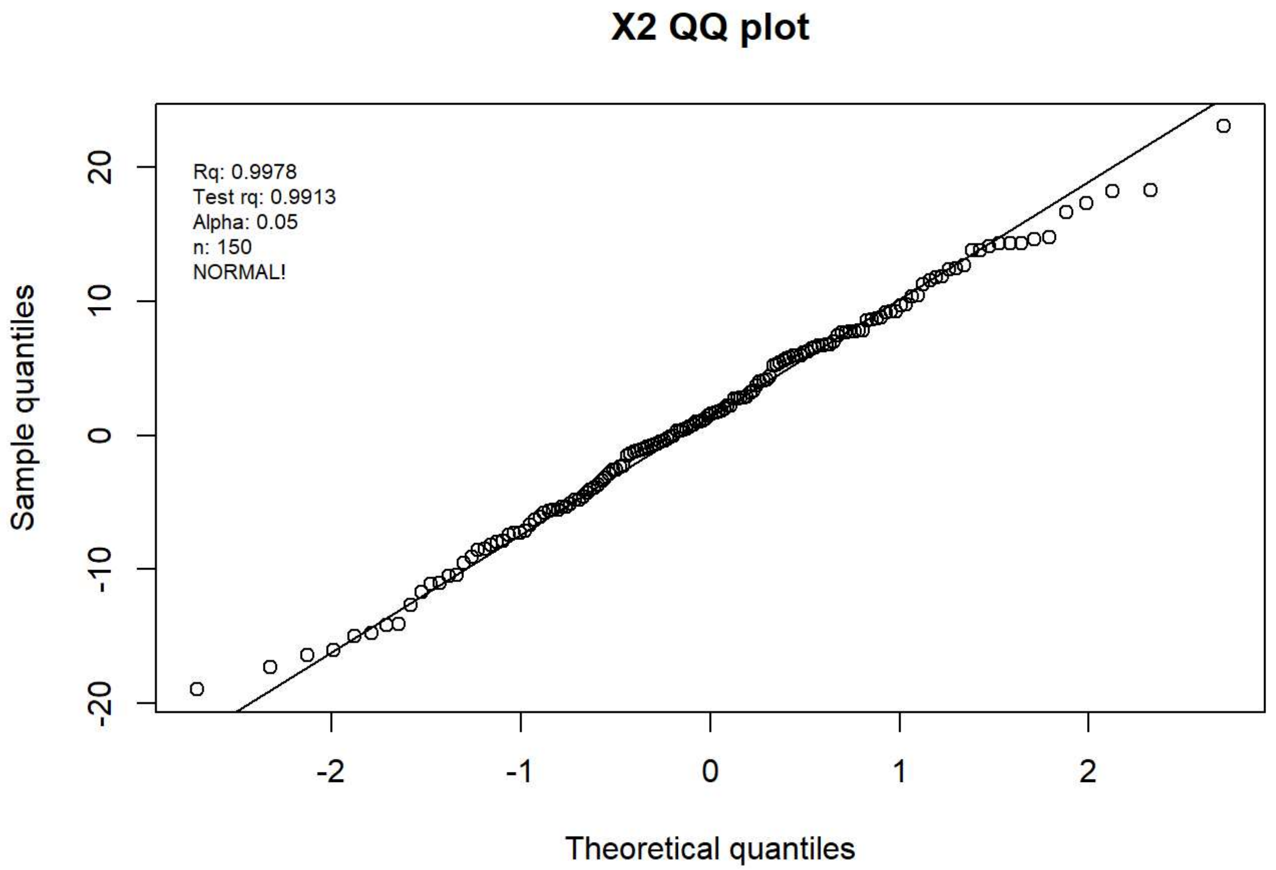
Q 1. p. 2

The data of this exercise is provided in Examdata4.tsv. All numerical values are rounded to 4 decimal places. Which of the following statements is not correct?

The correlation coefficient r_q for the QQ plot of X_2 is 0.9978

TRUE the r_q is 0.9978, this can be seen when performing the qqplot test with my function test norm:

```
test_norm(ex_4$X2, rounding = 4, name = "X2") # rounding = 4 us defualt just showing it to di
splay it for later
```

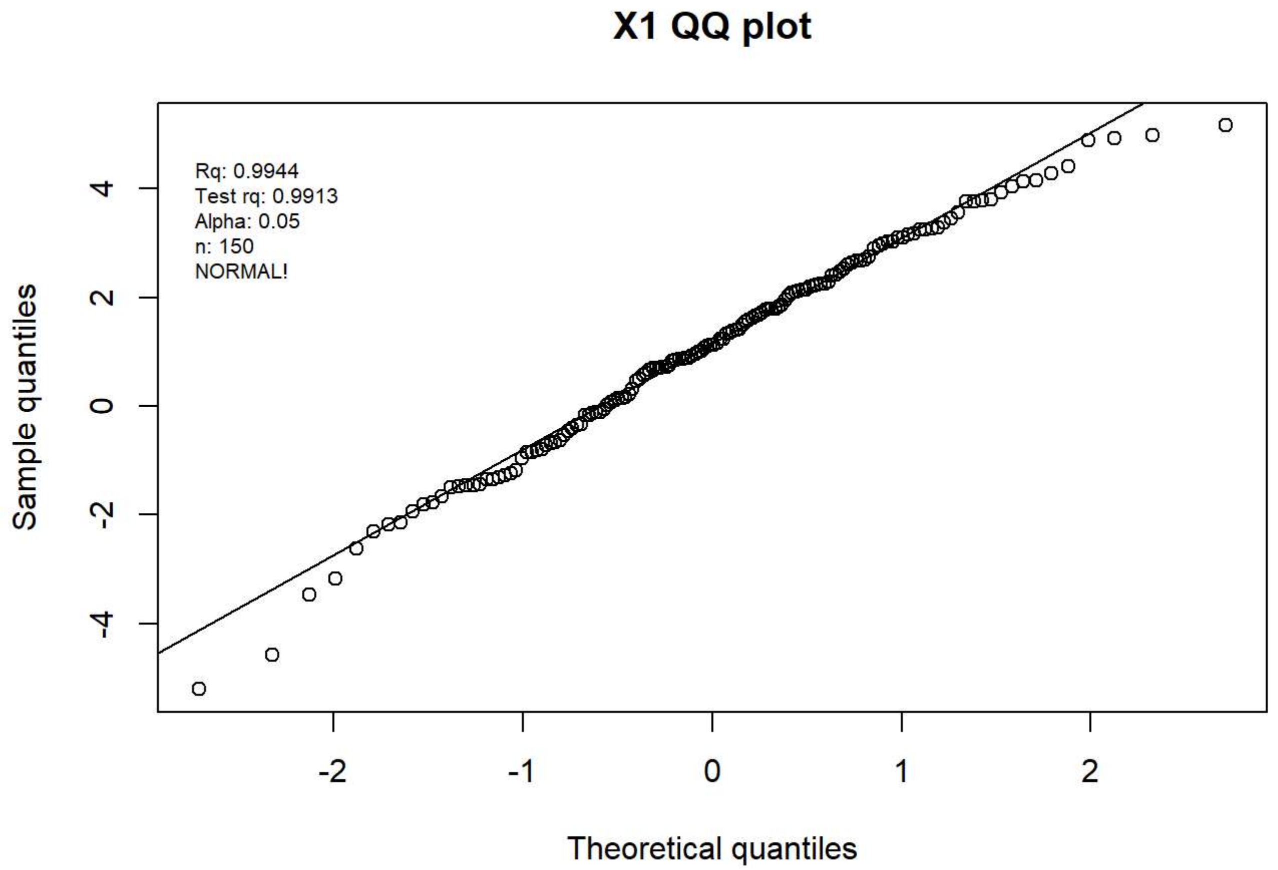



```
## [1] TRUE
```

X_1 is normally distributed with significance level of 0.05

TRUE Since there are 150 entries the rq must be higher than .9913, and since it is 0.9944 it is normally distributed:

```
test_norm(ex_4$X1, name = "X1")
```

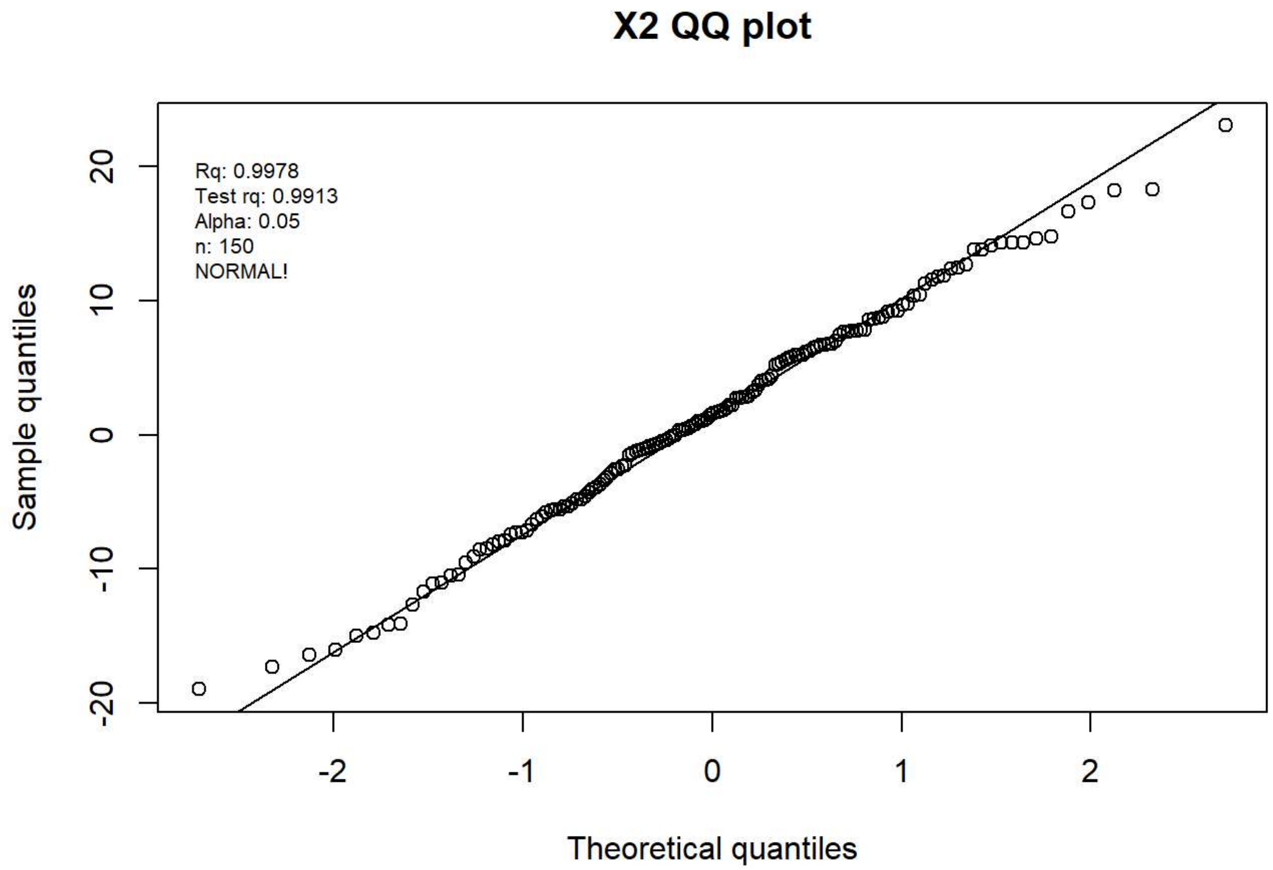



```
## [1] TRUE
```

X_2 is normally distributed with significance level 0.05

TRUE with a significance level of 0.9978, it is higher than 0.9913

```
test_norm(ex_4$X2, name = "X2")
```

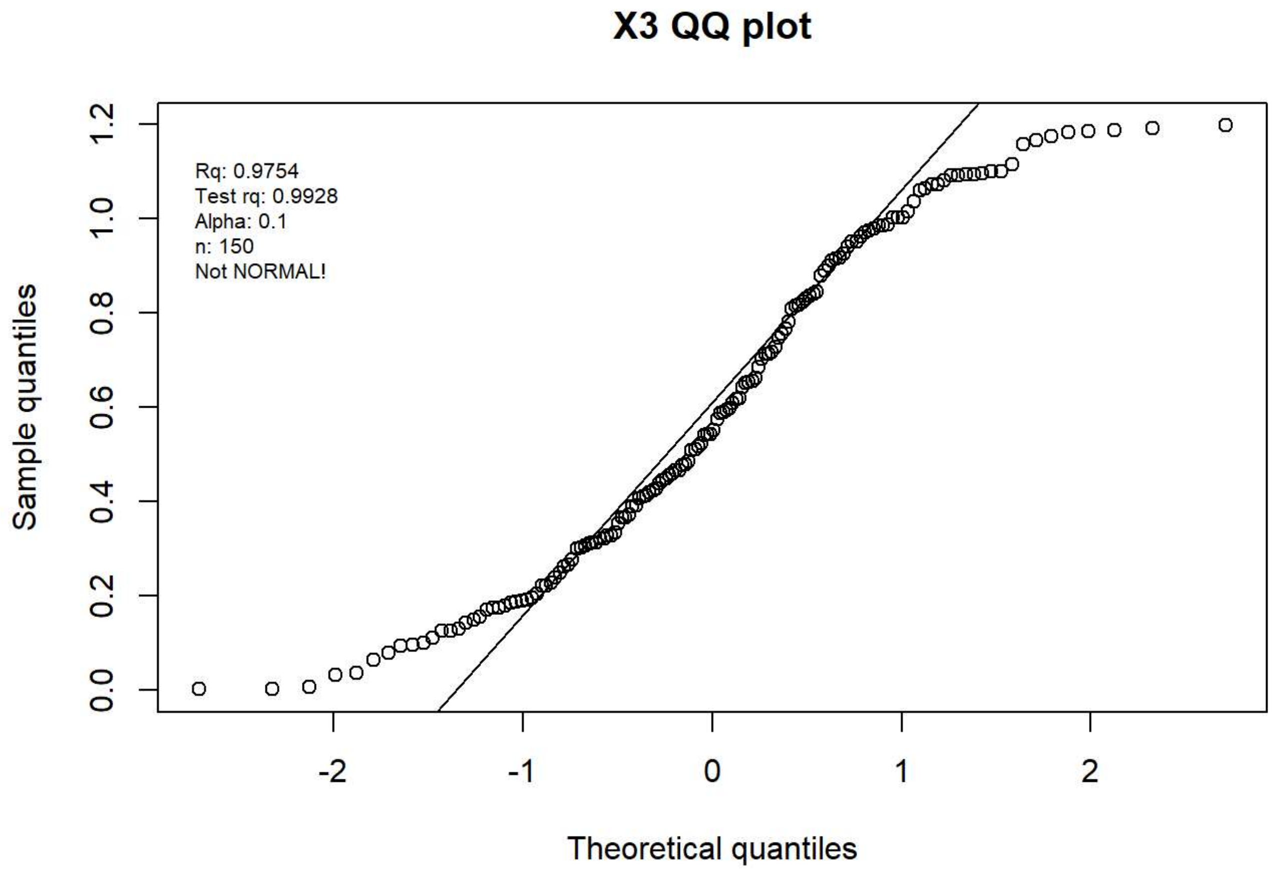



```
## [1] TRUE
```

X_3 is normally distributed with significance level 0.1

FALSE Since the significance is 0.1 it must be higher than 0.9928, however it is only 0.9754

```
test_norm(ex_4$X3, signigicance = 0.1, name = "X3")
```



Q2 p. 3

Assume that we have data that can be classified into 2 groups. Three classification methods (Method 1, Method 2 and Method 3) are applied to this data set. Their performance are evaluated according to their AUC (area under the curve) values, i.e.

- Method 1: 0.8 AUC
- Method 2: 0.85 AUC
- Method 3: 0.90 AUC

  Which method performs the best?

Q2 answer

Method 3 is the best, since its AUC is closer to 1.

Q3 p. 4

The data of this exercise is provided in Examdata5a.tsv and Examdata5b.tsv. We want to test the equality of their covariance matrices with significant level 0.05 assume that data are normally distributed. Null hypothesis is that their covariaance matrices are equal, what is the correct conclusion based on the Box's M-test.

We will perform the chi-squared test with a custom funciton. This function expects to have one dataframe with all the values and a second vector or matrix or column, which contains the group of each observation. Therefore we have to combine the dataframes and give them a second column which contains what group / data set they belong to:

```
ex_5a["group"] <- 1
ex_5b["group"] <- 2
ex_5_full <- rbind(ex_5a, ex_5b)
ex_5_full[, 3]
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2
## [39] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

We can now use the package:

```
box_m_test <- function(df, group_index, significance = 0.95){
  #This takes the full dataframe and the index of the column which contains the groups

  n <- nrow(df)
  p <- ncol(df[, -group_index])
  groups <- unique(df[, group_index])
  g <- length(groups)

  # get w
  w <- 0
  for (group in groups){
    mask <- df[,group_index] == group
    group_df <- df[mask, ]
    n_g <- nrow(group_df)
    s_g <- cov(group_df[, -group_index])
    w <- w + (n_g - 1) * s_g
  }
  # get w
  spooled <- w/(n - g)

  # get M
  M_sum <- 0
  for (group in groups){
    mask <- df[,group_index] == group
    group_df <- df[mask, ]
    n_g <- nrow(group_df)
    s_g <- cov(group_df[, -group_index])
    M_sum <- ((n_g - 1) * log(det(s_g))) + M_sum
  }
  M <- (n-g)*log(det(spooled)) - M_sum

  # get u
  sum_u <- 0
  for (group in groups){
    mask <- df[,group_index] == group
    group_df <- df[mask, ]
    n_g <- nrow(group_df)
    s_g <- cov(group_df[, -group_index])
    sum_u <- sum_u + (1/(n_g - 1))
  }
  u <-  ( sum_u - 1/(n-g)) * ( (2*p^2 + 3*p - 1) / (6*(p+1)*(g-1)))

  # Test statistic
  C <- (1-u)*M

  # Critcal value v
  critvalue <- qchisq(significance,p*(p+1)*(g-1)/2)   #v=p*(p+1)*(g-1)/2#

  ### final decision ####
  decisionflag <- (C > critvalue)

  writeLines(paste0("u :                          ", u, "\n",
                    "M statistic:                ", M, "\n",
                    "C value (Chi-squared value): ", C, "\n",
                    "Critival value (v):          ", critvalue, "\n",
```

```
                      "Equal covariance matrices:   ", decisionflag, "\n"))
}
```

We have to hypothesis:
- h0: The covariance matrices are equal
- h1: at least two of the covariance matrices are not equal
We can see if the approximated chi-squared value (C) is above the critical chi squared value for significance 0.05. If the value is above we accept h0 otherwise we reject h0:

```
box_m_test(ex_5_full,
           group_index = 3,
           significance = 0.95)
```

```
## u :                          0.0314249008745938
## M statistic:                 5.51147502867082
## C value (Chi-squared value): 5.33827747222204
## Critival value (v):          7.81472790325118
## Equal covariance matrices:   FALSE
```

As we can see the current chi value (5.3383) is below the critical chi-value (7.815), so we have to reject the null hypothesis. At least two of our co-variance matrices are not equal

Q4 . p. 5

Let $X = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix}$ be a random vector with properties $E(X) = \boldsymbol{\mu}$ and $Cov(X) = \begin{bmatrix} 2 & -0.5 & 0.5 \\ -2 & 2 & 0 \\ 3 & 0 & 6 \end{bmatrix}$. Further assume that X_1 is categorical. Which of the following statements is correct?

q 4

X_2 amd X_3 are uncorrelated

TRUE since both of them have are covariance of zero, they cannot have a direct correlation, since their variance does not differ together. ## X_2 and X_3 are independent
TRUE since they have zero covariance it is pretty safe to assume independence (p. 107)

The standard deviation of X_1 is 2

**FALSE* The standard deviation is defined as the square root of the variance. so let us calculate it

```
sqrt(2)
```

```
## [1] 1.414214
```

No the standard deviation is 1.41, but it does not make much sense to talk about, since it is categorical

There is a positve correlation between X_1 and X_2

No, since their co-variance is negative it cannot be positive. Moreover, x1 is categorical, which again does not make sense to talk about correlation.

Q. 5 p. 7

The data of this exercise is provided in Examdata3.tsv. All numerical values in the following are rounded to 4 decimal places. PCA analysis is done without standardization. Which of the following statements is not correct?

The correlation coefficient between the second prinicpal component and X_3 is negative

FALSE From a theoretical point of view principal components should not be able to be correlated, since the point of the dimension reduction is to only group correlated variables together, so each component is comprosied of different variables that are correlated. (see pca.pdf)

The first principal component is clearly dominant, since it explains more than 95% of the total variance

TRUE by looking at the summary of the principal component it it cleat that it explains +96% of the variance.

```
pc_ex3 <- prcomp(ex_3)
summary(pc_ex3)
```

```
## Importance of components:
##                           PC1     PC2     PC3
## Standard deviation     0.1527 0.02446 0.01897
## Proportion of Variance 0.9605 0.02466 0.01483
## Cumulative Proportion  0.9605 0.98517 1.00000
```

The first principal component is 0.6831X_1 + 0.5102X_2 + 0.5223X_3

FALSE when looking at the principal component it is ALMOST but not quite the coefficients above it is 0.6831X_1 + 0.5102X_2 + **0.5225X_3**

```
pc_ex3
```

```
## Standard deviations (1, .., p=3):
## [1] 0.15265434 0.02446027 0.01896934
##
## Rotation (n x k) = (3 x 3):
##           PC1        PC2        PC3
## V1 0.6831023  0.1594791 -0.7126974
## V2 0.5102195  0.5940118  0.6219534
## V3 0.5225392 -0.7884900  0.3244015
```

The sample mean vector is (4.7254, 4.4776, 3.7032)′)

TRUE We can check the mean vector of the original dataset:

```
colMeans(ex_3)
```

```
##       V1       V2       V3
## 4.725444 4.477574 3.703186
```

When this is rounded off to 4 decimals, it will yield the same sample mean vector.

Q. 6 p. 10

Let $X_1, X_2, \dots, X_{1000}$ be a random sample from the normal distribution $N_4(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Which of the following statements is correct?

Choose one answer

- ○ The distribution of $(\overline{X} - \boldsymbol{\mu})' \cdot \boldsymbol{S}^{-1} \cdot (\overline{X} - \boldsymbol{\mu})$ is approximately χ_3^2
- ○ $\overline{X}$ is distributed as $N_4(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
- ○ $1000 \cdot (\overline{X} - \boldsymbol{\mu})' \cdot \boldsymbol{\Sigma}^{-1} \cdot (\overline{X} - \boldsymbol{\mu})$ is distributed as χ_4^2
- ○ The distribution of $(X_1 - \boldsymbol{\mu}) \cdot \boldsymbol{\Sigma} \cdot (X_1 - \boldsymbol{\mu})'$ is χ_3^2.

Q 6

a)

FALSE the distribution is X_4^2 NOT X_3^2

b)

TRUE The sample mean vector will likewise follow the four variate normal distribution, since any subvector of a normal distributed vector will be normally distributed.

c)

TRUE even though the distribution is scaled it will still follow the same distribution.

d)

FALSE the distribution is X_4^2 NOT X_3^2