Tests and Confidence Intervals with R*

Before the beginning of the Fall term, students in a first-year Calculus class took a diagnostic test with two parts: Pre-calculus and Calculus. Their High School Calculus marks and their marks in University Calculus were also available. In order, the variables in the data file are: Identification code, Mark in High School Calculus, Score on the Pre-calculus portion of the diagnostic test, Score on the Calculus portion of the diagnostic test, and mark in University Calculus. Thanks to Dr. Cleo Boyd for permission to use these data.

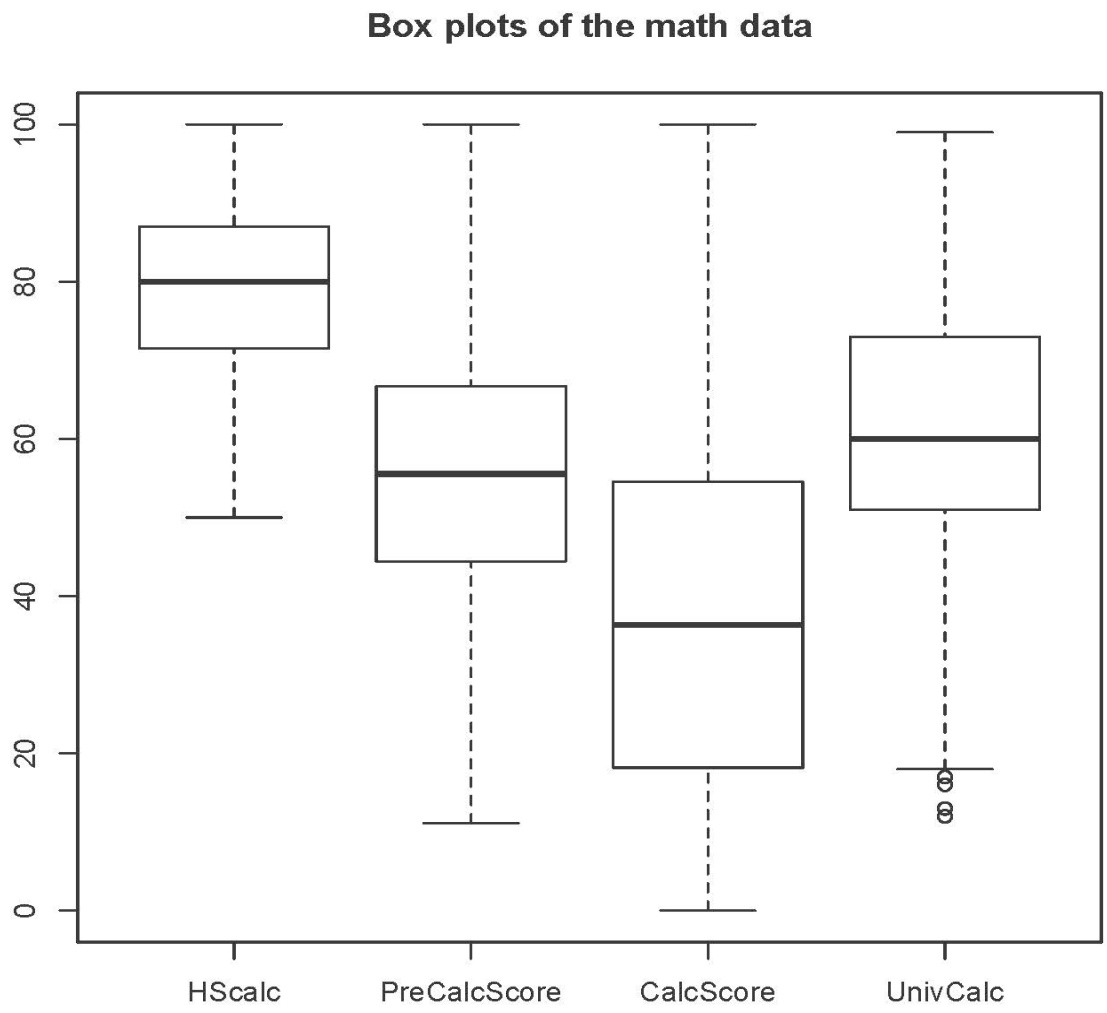
```
  1        65        2         0       39
  2        54        6         2       57
  3        77        4         4       62
  4        80        5         2       76
  5        87        4         4       86
  6        53        3         1       60

  .         .        .         .        .
  .         .        .         .        .
  .         .        .         .        .

290        83        4         3       56
291        81        6         3       70
292        73        5         9       60
293        80        5         2       50
294        56        4         2       50
295        80        6         1       61
```

```
> math = read.table("http://www.utstat.toronto.edu/~brunner/data/legal/mathtest.txt")
> head(math)
  V1 V2 V3 V4 V5
1  1 65  2  0 39
2  2 54  6  2 57
3  3 77  4  4 62
4  4 80  5  2 76
5  5 87  4  4 86
6  6 53  3  1 60
> colnames(math) = c("ID","HScalc","PreCalcScore","CalcScore","UnivCalc")
> head(math)
  ID HScalc PreCalcScore CalcScore UnivCalc
1  1     65            2         0       39
2  2     54            6         2       57
3  3     77            4         4       62
4  4     80            5         2       76
5  5     87            4         4       86
6  6     53            3         1       60
> summary(math)
       ID             HScalc        PreCalcScore     CalcScore        UnivCalc
 Min.   :  1.0   Min.   : 50.00   Min.   :1.000   Min.   : 0.000   Min.   :12.00
 1st Qu.: 74.5   1st Qu.: 71.50   1st Qu.:4.000   1st Qu.: 2.000   1st Qu.:51.00
 Median :148.0   Median : 80.00   Median :5.000   Median : 4.000   Median :60.00
 Mean   :148.0   Mean   : 78.51   Mean   :4.837   Mean   : 3.963   Mean   :60.91
 3rd Qu.:221.5   3rd Qu.: 87.00   3rd Qu.:6.000   3rd Qu.: 6.000   3rd Qu.:73.00
 Max.   :295.0   Max.   :100.00   Max.   :9.000   Max.   :11.000   Max.   :99.00
> attach(math) # Make variable names available
> # PreCalc score is out of 9 and Calc score is out of 11. Convert to percentages.
> PreCalcScore = 100 * PreCalcScore/9
> CalcScore = 100 * CalcScore/11
```

---

* Copyright information is on the last page.

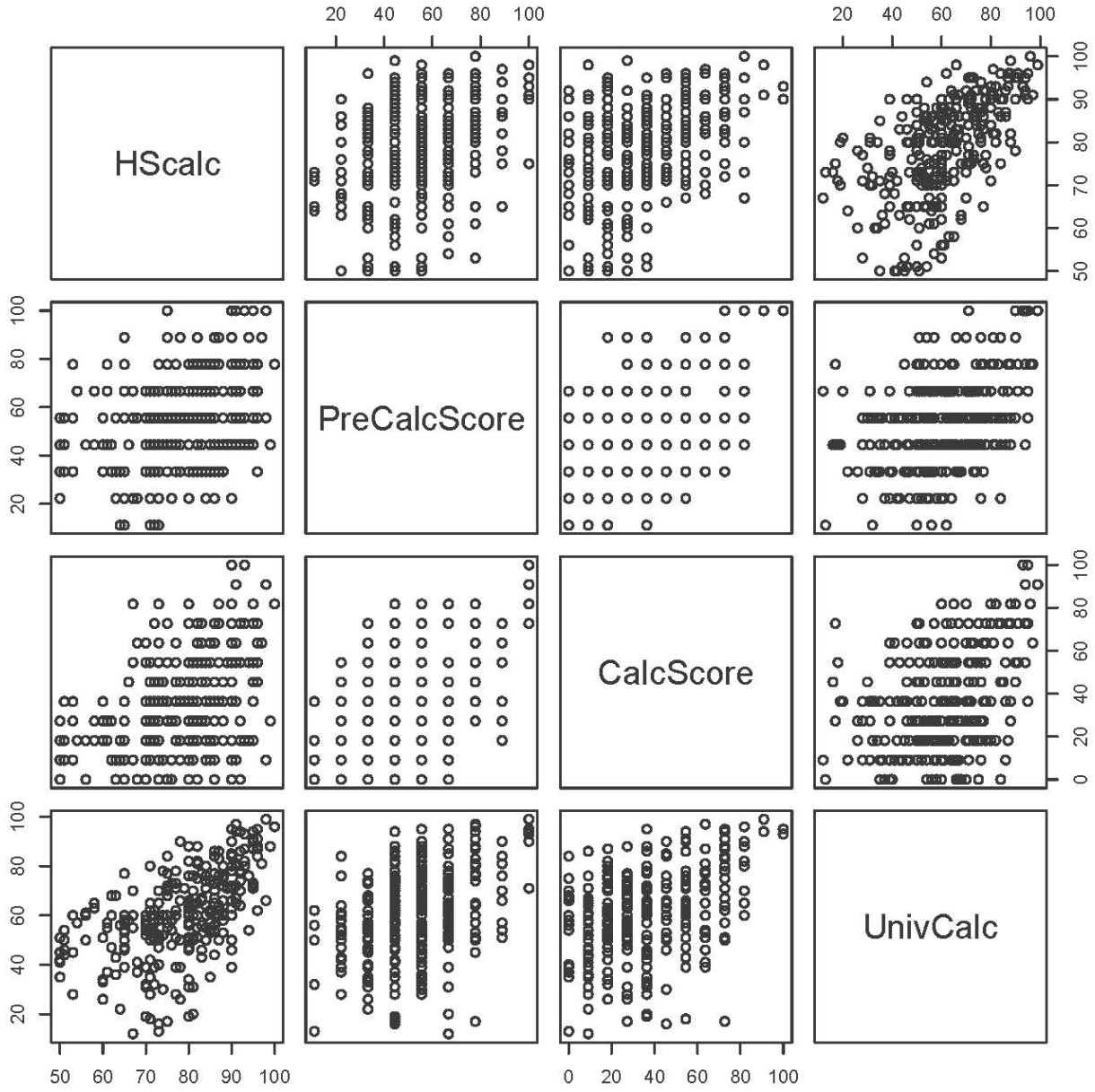
```
> ########## Descriptive stats ##########
> datamat = cbind(HScalc,PreCalcScore,CalcScore,UnivCalc)
> boxplot(datamat); title("Box plots of the math data")
```




```
> # upper whisker = min(max(x), Q_3 + 1.5 * IQR)
> # lower whisker = max(min(x), Q_1 – 1.5 * IQR)
> summary(datamat)
```

```
     HScalc        PreCalcScore      CalcScore        UnivCalc
 Min.   : 50.00   Min.   : 11.11   Min.   :  0.00   Min.   :12.00
 1st Qu.: 71.50   1st Qu.: 44.44   1st Qu.: 18.18   1st Qu.:51.00
 Median : 80.00   Median : 55.56   Median : 36.36   Median :60.00
 Mean   : 78.51   Mean   : 53.75   Mean   : 36.02   Mean   :60.91
 3rd Qu.: 87.00   3rd Qu.: 66.67   3rd Qu.: 54.55   3rd Qu.:73.00
 Max.   :100.00   Max.   :100.00   Max.   :100.00   Max.   :99.00
```

```
> cor(datamat); pairs(datamat)

                HScalc PreCalcScore CalcScore  UnivCalc
HScalc       1.0000000    0.3271197 0.4132186 0.5548571
PreCalcScore 0.3271197    1.0000000 0.4573164 0.3736003
CalcScore    0.4132186    0.4573164 1.0000000 0.3793398
UnivCalc     0.5548571    0.3736003 0.3793398 1.0000000
```



```
> ###########  Fit the full regression model ##########
> fullmodel = lm(UnivCalc ~ HScalc+PreCalcScore+CalcScore)
> sumfull = summary(fullmodel); sumfull

Call:
lm(formula = UnivCalc ~ HScalc + PreCalcScore + CalcScore)

Residuals:
    Min      1Q  Median      3Q     Max
-48.699  -7.954   1.603   9.242  30.260

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -6.32155    6.01019  -1.052  0.29376
HScalc        0.70097    0.08133   8.619  4.4e-16 ***
PreCalcScore  0.16849    0.05182   3.252  0.00128 **
CalcScore     0.08722    0.04282   2.037  0.04257 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 14.16 on 291 degrees of freedom
Multiple R-squared:  0.3583,  Adjusted R-squared:  0.3517
F-statistic: 54.17 on 3 and 291 DF,  p-value: < 2.2e-16
```

$$t = \frac{\ell'\mathbf{b} - \ell'\beta}{s\sqrt{\ell'(X'X)^{-1}\ell}} \sim t(n-k-1)$$

```
> # Reproduce the test for H0: beta1=0
> ell = as.matrix(c(0,1,0,0)); ell
     [,1]
[1,]    0
[2,]    1
[3,]    0
[4,]    0
> b = coefficients(fullmodel)
> MSE.XpXinv = vcov(fullmodel) # Very handy
> dfe = df.residual(fullmodel) # n-k-1
> tstat = t(ell) %*% b / sqrt( t(ell)%*%MSE.XpXinv%*%ell ) # tstat is a 1x1 matrix
> tstat = as.numeric(tstat) # Make it a number
> pvalue = 2 * ( 1-pt(abs(tstat),dfe) )
> c(tstat,pvalue)
[1] 8.618832e+00 4.440892e-16
```

```
> ########### Confidence intervals ###########
```

$$\boldsymbol{\ell}'\mathbf{b} \pm t_{\alpha/2}\, s \sqrt{\boldsymbol{\ell}'(X'X)^{-1}\boldsymbol{\ell}}$$

```
> # Confidence interval for beta_j is just b_j plus or minus t_{alpha/2} * Standard error
> t.025 = qt(0.975,dfe)
> t.025  # Definitely display t_{alpha/2} so you can produce confidence intervals by hand
[1] 1.96815
> # If High School calculus is 10 marks higher, expected university calculus increases by
> # 10*beta1. Estimate the increase and give a 95% confidence interval.
> me95 = 10 * t.025*0.08133 # Number copied from the printout.
> cat("University calculus mark is expected to be",10*b[2],"points higher, plus or
minus",me95,"\n")
University calculus mark is expected to be 7.00972 points higher, plus or minus
1.600696

> # Confidence interval for beta3-beta2
> ell = as.matrix(c(0,0,-1,1)) # Now ell is different
> se = sqrt( t(ell)%*%MSE.XpXinv%*%ell ) # Standard error of the difference
> me95 = as.numeric( t.025*se ) # Now me95 is different
> estdiff = as.numeric( t(ell) %*% b ); estdiff
[1] -0.08127008
> Lower95 = estdiff - me95; Upper95 = estdiff + me95
> c(Lower95, Upper95)
[1] -0.23598624  0.07344609

> ########### General linear test ###########
```

$$F = \frac{(\mathbf{Cb}-\boldsymbol{\gamma})'(\mathbf{C}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{C}')^{-1}(\mathbf{Cb}-\boldsymbol{\gamma})}{m\, s^2}$$

```
> # Reproduce the overall test: F = 54.17
> # First do it the hard way
> C1 = rbind( c(0,1,0,0),
+             c(0,0,1,0),
+             c(0,0,0,1) )
> m = dim(C1)[1] # Number of rows
> F1 = t(C1%*%b-0) %*% solve(C1 %*% MSE.XpXinv %*% t(C1)) %*% (C1%*%b-0) / m
> F1 # It's a 1x1 matrix
         [,1]
[1,] 54.16969
```

```
> # The easy way: Use this function if you like.
> source("http://www.utstat.utoronto.ca/~brunner/Rfunctions/ftest.txt")
> ftest # See the function definition
function(model,L,h=0)
# General linear test of H0: L beta = h
    {
    BetaHat = model$coefficients
    dimL = dim(L)
    if(length(BetaHat) != dimL[2]) stop("Sizes of L and Beta are incompatible")
    r = dimL[1]
    if(qr(L)$rank != r) stop("Rows of L must be linearly independent.")
    out = numeric(4)
    names(out) = c("F","df1","df2","p-value")
    dfe = df.residual(model)
    diff = L%*%BetaHat-h
    fstat = t(diff) %*% solve(L%*%vcov(model)%*%t(L)) %*% diff / r
    # Note vcov = MSE * XtXinv
    fstat = as.numeric(fstat)
    out[1] = fstat; out[2]=r; out[3]=dfe
    out[4] = 1-pf(fstat,r,dfe)
    return(out)
    }
> ftest(fullmodel,C1) # Compare F = 54.17
        F       df1       df2   p-value
 54.16969   3.00000 291.00000   0.00000

> # Test the two subtests simultaneously, controlling for HS Calculus mark
> C2 = rbind( c(0,0,1,0),
+             c(0,0,0,1) )
> ftest(fullmodel,C2)
           F          df1          df2      p-value
1.144444e+01 2.000000e+00 2.910000e+02 1.642604e-05

> round(ftest(fullmodel,C2),4)
       F      df1      df2  p-value
 11.4444   2.0000 291.0000   0.0000

> # Do it with the full-reduced approach
> JustHScalc = lm(UnivCalc ~ HScalc) # The reduced (restricted) model
> anova(JustHScalc,fullmodel)

Analysis of Variance Table

Model 1: UnivCalc ~ HScalc
Model 2: UnivCalc ~ HScalc + PreCalcScore + CalcScore
  Res.Df   RSS Df Sum of Sq      F    Pr(>F)
1    293 62967
2    291 58375  2    4591.5 11.444 1.643e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```
