MAST30025 Assignment 3 2015

#Question 1

setwd("~/Desktop/UNIMELB 2021 Material/UNIMELB S1 2021 (Currently)/MAST30025/Tutorials /Tutorials/Rfile/data")  
douglas = read.csv("douglas.csv")  
str(douglas)

## 'data.frame': 54 obs. of 4 variables:  
## $ Plot : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ RootVolume: chr "RV3" "RV3" "RV1" "RV2" ...  
## $ SeedLot : chr "A567" "J052" "A567" "A567" ...  
## $ Height : num 54.4 56.4 47 54.6 49.3 ...

#Part a: What sort of experimental design has been used? #How has randomisation been used, if at all?

#Complete block design (CRBD)

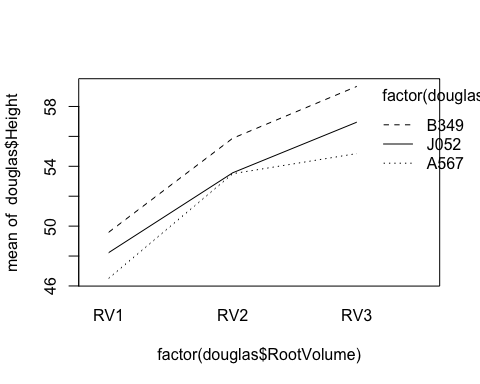
#Not sure if I have seen this in the lectures  
idx = order(douglas$SeedLot, douglas$RootVolume)  
douglas = douglas[idx,]  
head(douglas)

## Plot RootVolume SeedLot Height  
## 3 3 RV1 A567 46.97  
## 13 13 RV1 A567 45.45  
## 19 19 RV1 A567 47.34  
## 29 29 RV1 A567 45.88  
## 43 43 RV1 A567 44.08  
## 53 53 RV1 A567 49.31

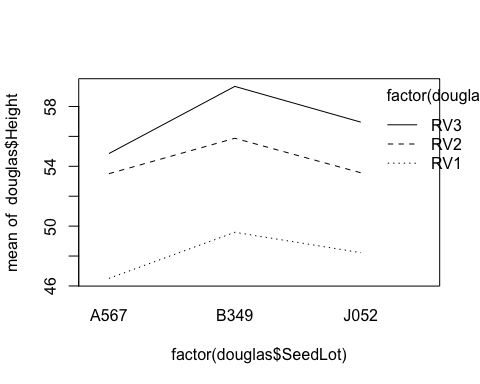
## Including Plots

#Part b) Generate two interaction plots for the data. Is there any evidence of an interaction? #Read from slide 62-67 from the Inference for the less than full rank model.

#Attempt 1  
with(douglas,interaction.plot(factor(douglas$RootVolume),factor(douglas$SeedLot),douglas$Height))

 #From factor J052 and A567 there is a little interaction at the center from this plot.

#Attempt 1  
with(douglas,interaction.plot(factor(douglas$SeedLot),factor(douglas$RootVolume),douglas$Height))

 #All the lines are parallel so there is no interaction from this plot.

#Part c: Specify a model for the data, including main effects and an interaction, and write down the design matrix X. Hence, calculate a solution to the normal equations, and use it to find the fitted means for each combination of factors levels. You may not use the ginv command for this question.

library(MASS)  
library(Matrix)  
y = douglas$Height  
n = length(y)  
X = matrix(c(rep(1,n),rep(0,n\*15)),n,16)  
X[cbind(1:n,as.numeric(factor(douglas$SeedLot))+1)] = 1  
X[cbind(1:n,as.numeric(factor(douglas$RootVolume))+4)] = 1  
X[cbind(1:n,as.numeric(factor(douglas$RootVolume))\*3 + as.numeric(factor(douglas$SeedLot))+4)] = 1  
X

## [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]  
## [1,] 1 1 0 0 1 0 0 1 0 0 0 0 0  
## [2,] 1 1 0 0 1 0 0 1 0 0 0 0 0  
## [3,] 1 1 0 0 1 0 0 1 0 0 0 0 0  
## [4,] 1 1 0 0 1 0 0 1 0 0 0 0 0  
## [5,] 1 1 0 0 1 0 0 1 0 0 0 0 0  
## [6,] 1 1 0 0 1 0 0 1 0 0 0 0 0  
## [7,] 1 1 0 0 0 1 0 0 0 0 1 0 0  
## [8,] 1 1 0 0 0 1 0 0 0 0 1 0 0  
## [9,] 1 1 0 0 0 1 0 0 0 0 1 0 0  
## [10,] 1 1 0 0 0 1 0 0 0 0 1 0 0  
## [11,] 1 1 0 0 0 1 0 0 0 0 1 0 0  
## [12,] 1 1 0 0 0 1 0 0 0 0 1 0 0  
## [13,] 1 1 0 0 0 0 1 0 0 0 0 0 0  
## [14,] 1 1 0 0 0 0 1 0 0 0 0 0 0  
## [15,] 1 1 0 0 0 0 1 0 0 0 0 0 0  
## [16,] 1 1 0 0 0 0 1 0 0 0 0 0 0  
## [17,] 1 1 0 0 0 0 1 0 0 0 0 0 0  
## [18,] 1 1 0 0 0 0 1 0 0 0 0 0 0  
## [19,] 1 0 1 0 1 0 0 0 1 0 0 0 0  
## [20,] 1 0 1 0 1 0 0 0 1 0 0 0 0  
## [21,] 1 0 1 0 1 0 0 0 1 0 0 0 0  
## [22,] 1 0 1 0 1 0 0 0 1 0 0 0 0  
## [23,] 1 0 1 0 1 0 0 0 1 0 0 0 0  
## [24,] 1 0 1 0 1 0 0 0 1 0 0 0 0  
## [25,] 1 0 1 0 0 1 0 0 0 0 0 1 0  
## [26,] 1 0 1 0 0 1 0 0 0 0 0 1 0  
## [27,] 1 0 1 0 0 1 0 0 0 0 0 1 0  
## [28,] 1 0 1 0 0 1 0 0 0 0 0 1 0  
## [29,] 1 0 1 0 0 1 0 0 0 0 0 1 0  
## [30,] 1 0 1 0 0 1 0 0 0 0 0 1 0  
## [31,] 1 0 1 0 0 0 1 0 0 0 0 0 0  
## [32,] 1 0 1 0 0 0 1 0 0 0 0 0 0  
## [33,] 1 0 1 0 0 0 1 0 0 0 0 0 0  
## [34,] 1 0 1 0 0 0 1 0 0 0 0 0 0  
## [35,] 1 0 1 0 0 0 1 0 0 0 0 0 0  
## [36,] 1 0 1 0 0 0 1 0 0 0 0 0 0  
## [37,] 1 0 0 1 1 0 0 0 0 1 0 0 0  
## [38,] 1 0 0 1 1 0 0 0 0 1 0 0 0  
## [39,] 1 0 0 1 1 0 0 0 0 1 0 0 0  
## [40,] 1 0 0 1 1 0 0 0 0 1 0 0 0  
## [41,] 1 0 0 1 1 0 0 0 0 1 0 0 0  
## [42,] 1 0 0 1 1 0 0 0 0 1 0 0 0  
## [43,] 1 0 0 1 0 1 0 0 0 0 0 0 1  
## [44,] 1 0 0 1 0 1 0 0 0 0 0 0 1  
## [45,] 1 0 0 1 0 1 0 0 0 0 0 0 1  
## [46,] 1 0 0 1 0 1 0 0 0 0 0 0 1  
## [47,] 1 0 0 1 0 1 0 0 0 0 0 0 1  
## [48,] 1 0 0 1 0 1 0 0 0 0 0 0 1  
## [49,] 1 0 0 1 0 0 1 0 0 0 0 0 0  
## [50,] 1 0 0 1 0 0 1 0 0 0 0 0 0  
## [51,] 1 0 0 1 0 0 1 0 0 0 0 0 0  
## [52,] 1 0 0 1 0 0 1 0 0 0 0 0 0  
## [53,] 1 0 0 1 0 0 1 0 0 0 0 0 0  
## [54,] 1 0 0 1 0 0 1 0 0 0 0 0 0  
## [,14] [,15] [,16]  
## [1,] 0 0 0  
## [2,] 0 0 0  
## [3,] 0 0 0  
## [4,] 0 0 0  
## [5,] 0 0 0  
## [6,] 0 0 0  
## [7,] 0 0 0  
## [8,] 0 0 0  
## [9,] 0 0 0  
## [10,] 0 0 0  
## [11,] 0 0 0  
## [12,] 0 0 0  
## [13,] 1 0 0  
## [14,] 1 0 0  
## [15,] 1 0 0  
## [16,] 1 0 0  
## [17,] 1 0 0  
## [18,] 1 0 0  
## [19,] 0 0 0  
## [20,] 0 0 0  
## [21,] 0 0 0  
## [22,] 0 0 0  
## [23,] 0 0 0  
## [24,] 0 0 0  
## [25,] 0 0 0  
## [26,] 0 0 0  
## [27,] 0 0 0  
## [28,] 0 0 0  
## [29,] 0 0 0  
## [30,] 0 0 0  
## [31,] 0 1 0  
## [32,] 0 1 0  
## [33,] 0 1 0  
## [34,] 0 1 0  
## [35,] 0 1 0  
## [36,] 0 1 0  
## [37,] 0 0 0  
## [38,] 0 0 0  
## [39,] 0 0 0  
## [40,] 0 0 0  
## [41,] 0 0 0  
## [42,] 0 0 0  
## [43,] 0 0 0  
## [44,] 0 0 0  
## [45,] 0 0 0  
## [46,] 0 0 0  
## [47,] 0 0 0  
## [48,] 0 0 0  
## [49,] 0 0 1  
## [50,] 0 0 1  
## [51,] 0 0 1  
## [52,] 0 0 1  
## [53,] 0 0 1  
## [54,] 0 0 1

#Finding the Condition Inverse according from the lectures!  
XtXc = matrix(0,16,16)  
XtXc[8:16,8:16] = solve((t(X)%\*%X)[8:16,8:16])  
XTXC = XtXc[-(1:6),-(1:6)]  
XTXC

## [,1] [,2] [,3] [,4] [,5] [,6] [,7]  
## [1,] 0 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
## [2,] 0 0.1666667 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
## [3,] 0 0.0000000 0.1666667 0.0000000 0.0000000 0.0000000 0.0000000  
## [4,] 0 0.0000000 0.0000000 0.1666667 0.0000000 0.0000000 0.0000000  
## [5,] 0 0.0000000 0.0000000 0.0000000 0.1666667 0.0000000 0.0000000  
## [6,] 0 0.0000000 0.0000000 0.0000000 0.0000000 0.1666667 0.0000000  
## [7,] 0 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.1666667  
## [8,] 0 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
## [9,] 0 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
## [10,] 0 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000  
## [,8] [,9] [,10]  
## [1,] 0.0000000 0.0000000 0.0000000  
## [2,] 0.0000000 0.0000000 0.0000000  
## [3,] 0.0000000 0.0000000 0.0000000  
## [4,] 0.0000000 0.0000000 0.0000000  
## [5,] 0.0000000 0.0000000 0.0000000  
## [6,] 0.0000000 0.0000000 0.0000000  
## [7,] 0.0000000 0.0000000 0.0000000  
## [8,] 0.1666667 0.0000000 0.0000000  
## [9,] 0.0000000 0.1666667 0.0000000  
## [10,] 0.0000000 0.0000000 0.1666667

library(MASS)  
#Now we can find one of our least sqaures estimator b,  
b = XtXc%\*%t(X)%\*%y  
b[-(1:7),]

## [1] 46.50500 49.58833 48.22667 53.50500 55.87500 53.56667 54.85000 59.34167  
## [9] 56.95000

#46.50500 49.58833 48.22667 53.50500 55.87500 53.56667 54.85000 59.34167 56.95000

I = diag(16)  
b2 = b + (I - XtXc %\*% t(X) %\*% X) %\*% as.vector(c(1,rep(0,14),1))  
b2

## [,1]  
## [1,] 1.00000  
## [2,] 0.00000  
## [3,] 0.00000  
## [4,] 0.00000  
## [5,] 0.00000  
## [6,] 0.00000  
## [7,] 0.00000  
## [8,] 45.50500  
## [9,] 48.58833  
## [10,] 47.22667  
## [11,] 52.50500  
## [12,] 54.87500  
## [13,] 52.56667  
## [14,] 53.85000  
## [15,] 58.34167  
## [16,] 55.95000

#as.vector() is an arbitary vector!  
#b2[-(1:7),]  
#[1] 45.50500 48.58833 47.22667 52.50500 54.87500 52.56667 53.85000 58.34167 55.95000

#In the actual solution  
colMeans(matrix(douglas$Height, nrow = 6))

## [1] 46.50500 53.50500 54.85000 49.58833 55.87500 59.34167 48.22667 53.56667  
## [9] 56.95000

#Part d: Give a 95% confidence interval for the difference in height between a seedling with large root volume (RV3) and a seedling with small root volume (RV1). Suppose that the seedling came from lot A567

library(MASS)  
library(Matrix)  
y = douglas$Height  
n = length(y)  
r = rankMatrix(X)[1]  
s2 = sum((y-X%\*%b)^2)/(n-r)  
s2

## [1] 3.044433

#Fixing the design Matrix from the previous part!

#Testing with our C martix   
  
#C = matrix(0,1,16)   
#C[1,c(2,3,8,10)] = c(1,1,-1,1)  
  
#Actual solution:   
t = c(0,0,0,0,-1,0,1,-1,0,1,0,0,0,0,0,0)  
  
q = qt(0.975,n-r)  
s = sqrt(s2)  
width = s\*q\*sqrt(t(t)%\*%XtXc%\*%t)  
m = t%\*%b  
m

## [,1]  
## [1,] 1.721667

c(m-width,m+width)

## [1] -0.3072973 3.7506307

#CI is wrong

#Fixing the Design Matrix

library(MASS)  
library(Matrix)  
options(width=100)  
X <- matrix(0, nrow = 54, ncol = 16)  
X[,1] = 1  
X[,2] = douglas$SeedLot == "A567"  
X[,3] = douglas$SeedLot == "B349"  
X[,4] <- douglas$SeedLot == "J052"  
X[,5] <- douglas$RootVolume == "RV1"  
X[,6] <- douglas$RootVolume == "RV2"  
X[,7] <- douglas$RootVolume == "RV3"  
X[,8] <- X[,2]\*X[,5]  
X[,9] <- X[,2]\*X[,6]  
X[,10] <- X[,2]\*X[,7]  
X[,11] <- X[,3]\*X[,5]  
X[,12] <- X[,3]\*X[,6]  
X[,13] <- X[,3]\*X[,7]  
X[,14] <- X[,4]\*X[,5]  
X[,15] <- X[,4]\*X[,6]  
X[,16] <- X[,4]\*X[,7]  
X

## [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14] [,15] [,16]  
## [1,] 1 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0  
## [2,] 1 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0  
## [3,] 1 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0  
## [4,] 1 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0  
## [5,] 1 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0  
## [6,] 1 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0  
## [7,] 1 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0  
## [8,] 1 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0  
## [9,] 1 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0  
## [10,] 1 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0  
## [11,] 1 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0  
## [12,] 1 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0  
## [13,] 1 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0  
## [14,] 1 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0  
## [15,] 1 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0  
## [16,] 1 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0  
## [17,] 1 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0  
## [18,] 1 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0  
## [19,] 1 0 1 0 1 0 0 0 0 0 1 0 0 0 0 0  
## [20,] 1 0 1 0 1 0 0 0 0 0 1 0 0 0 0 0  
## [21,] 1 0 1 0 1 0 0 0 0 0 1 0 0 0 0 0  
## [22,] 1 0 1 0 1 0 0 0 0 0 1 0 0 0 0 0  
## [23,] 1 0 1 0 1 0 0 0 0 0 1 0 0 0 0 0  
## [24,] 1 0 1 0 1 0 0 0 0 0 1 0 0 0 0 0  
## [25,] 1 0 1 0 0 1 0 0 0 0 0 1 0 0 0 0  
## [26,] 1 0 1 0 0 1 0 0 0 0 0 1 0 0 0 0  
## [27,] 1 0 1 0 0 1 0 0 0 0 0 1 0 0 0 0  
## [28,] 1 0 1 0 0 1 0 0 0 0 0 1 0 0 0 0  
## [29,] 1 0 1 0 0 1 0 0 0 0 0 1 0 0 0 0  
## [30,] 1 0 1 0 0 1 0 0 0 0 0 1 0 0 0 0  
## [31,] 1 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0  
## [32,] 1 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0  
## [33,] 1 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0  
## [34,] 1 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0  
## [35,] 1 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0  
## [36,] 1 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0  
## [37,] 1 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0  
## [38,] 1 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0  
## [39,] 1 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0  
## [40,] 1 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0  
## [41,] 1 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0  
## [42,] 1 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0  
## [43,] 1 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0  
## [44,] 1 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0  
## [45,] 1 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0  
## [46,] 1 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0  
## [47,] 1 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0  
## [48,] 1 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0  
## [49,] 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1  
## [50,] 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1  
## [51,] 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1  
## [52,] 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1  
## [53,] 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1  
## [54,] 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1

library(MASS)  
library(Matrix)  
XTXc2 = matrix(0, nrow = 16, ncol = 16)  
XTXc2[8:16,8:16] = diag(9)/6  
t <- c(0,0,0,0,-1,0,1,-1,0,1,0,0,0,0,0,0)  
r <- qt(0.975, 45)\*sqrt(s2 \* t(t) %\*% XTXc2 %\*% t)  
m <- t(t) %\*% b  
 cat("(", m - r, ",", m + r, ")\n")

## ( -0.3072973 , 3.750631 )

#Part e: Test for the presence of an interaction at the 5% significance level.Would it be meaningful to check the significance of the main effects? Why? You may not use the lm command for this question.

#Actual solution  
library(Matrix)  
library(MASS)  
C = matrix(c(0,0,0,0,0,0,0, 1,-1,0, -1,1,0, 0,0,0,0,0,0,0,0,0,0, 1,-1,0, 0,0,0,-1,1,0,0,0,0,0,0,0,0, 0,1,-1, 0,0,0,0,-1,1,0,0,0,0,0,0,0, 0,0,0, 1,0,-1, -1,0,1), nrow = 4, byrow =TRUE)  
rankMatrix(C)

## [1] 4  
## attr(,"method")  
## [1] "tolNorm2"  
## attr(,"useGrad")  
## [1] FALSE  
## attr(,"tol")  
## [1] 3.552714e-15

check = C %\*% XtXc %\*% t(X) %\*% X  
round(C - check, 10)

## [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14] [,15] [,16]  
## [1,] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0  
## [2,] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0  
## [3,] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0  
## [4,] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Fstat <- t(b) %\*% t(C) %\*% solve( C %\*% XtXc %\*% t(C), C %\*% b) /4 /s2  
Fstat

## [,1]  
## [1,] 0.8765331

pf(Fstat, 4, 45, lower.tail = FALSE)

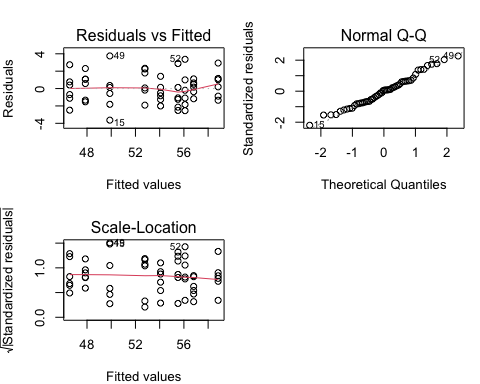
## [,1]  
## [1,] 0.4855229

#Tutor comment: The large p-value indicates that we should retain the null hypothesis that there is no interac- tion. #In the absence of an interaction it does make sense to test for the significance of the main effects (we would have to refit the model without the interaction first)

#Part f: Fit an additive model to the data using the lm command, and produce plots to justify the model assumption that the errors are normal and homoskedastic.

model <- lm(Height ~ SeedLot + RootVolume, data = douglas)  
opar <- par(mfrow=c(2,2),mar=c(4,4,3,1))  
plot(model, which = 1)  
plot(model, which = 2)  
plot(model, which = 3)  
plot(model, which = 5)

## hat values (leverages) are all = 0.09259259  
## and there are no factor predictors; no plot no. 5



par = opar

plot(model, which = 5)

## hat values (leverages) are all = 0.09259259  
## and there are no factor predictors; no plot no. 5



#From the second plot the residuals look normal. In the other three there is no sign of heteroskedasticity or outliers.

#Part g: Suppose that you are to repeat the experiment, except that this time you source seeds from four seed lots, and you only have the resources to plant out 36 seedlings.

#Give a design for this experiment, showing which combination of root volume and seed lot is used for each plot (one seedling is planted in each plot). Explain the experimental design principle(s) you have used.

#Actual Solution  
#We use a complete balanced block design. SeedLot is the blocking factor with four levels, and RootVolume is the treatment factor with three levels. With 36 plots we have three replications. We assign plots at random to each factor combination  
design = data.frame(Plot = sample(36), SeedLot=rep(1:4,c(9,9,9,9)), RootVolume = rep(rep(1:3, c(3,3,3)), 4))  
  
design

## Plot SeedLot RootVolume  
## 1 10 1 1  
## 2 20 1 1  
## 3 22 1 1  
## 4 29 1 2  
## 5 18 1 2  
## 6 13 1 2  
## 7 19 1 3  
## 8 27 1 3  
## 9 6 1 3  
## 10 28 2 1  
## 11 1 2 1  
## 12 9 2 1  
## 13 5 2 2  
## 14 11 2 2  
## 15 4 2 2  
## 16 33 2 3  
## 17 35 2 3  
## 18 2 2 3  
## 19 25 3 1  
## 20 15 3 1  
## 21 26 3 1  
## 22 12 3 2  
## 23 24 3 2  
## 24 14 3 2  
## 25 16 3 3  
## 26 36 3 3  
## 27 23 3 3  
## 28 21 4 1  
## 29 31 4 1  
## 30 34 4 1  
## 31 17 4 2  
## 32 8 4 2  
## 33 3 4 2  
## 34 30 4 3  
## 35 32 4 3  
## 36 7 4 3

#Each factor combination will get three plots, and we need to assign three seedlings to these plots at random. This can be achieved using the randomised plots. We just assign the three seedlings in the order indicated by the plot numbers, e.g. (low, med, high), (low, high, med), (med, low, high), etc.

#This design uses blocking to control the effect of SeedLot, as well as balance and replication to increase the precision. Randomisation has been used to mitigate any effects due to the physical location of the seedlings (as indicated by the plot number).

#Question 2 #From the Latin square slides (Experimental design this is to test your understanding how to define your function and generate different latinsquares)

#Part a: DefineA=(aij)byaij =(i+j−2 modn)+1.

library(magic)

## Loading required package: abind

rlatin(5)

## [,1] [,2] [,3] [,4] [,5]  
## [1,] 1 5 4 3 2  
## [2,] 5 3 2 1 4  
## [3,] 3 2 1 4 5  
## [4,] 4 1 5 2 3  
## [5,] 2 4 3 5 1

#Actual solution

rlatinsquare = function(n){  
 #generate an n\*n latin square by row and col permuations  
 A = matrix(1:n, nrow = n, ncol = n)  
 A = ((A + t(A) - 2)%%n)+1  
 L = A[sample(n),sample(n)]  
 return(L)  
}  
set.seed(30025)  
  
  
rlatinsquare(4) #Obtain B (Trail and Error)

## [,1] [,2] [,3] [,4]  
## [1,] 1 4 2 3  
## [2,] 4 3 1 2  
## [3,] 3 2 4 1  
## [4,] 2 1 3 4

#Obtain B by permuting the rows of A randomly.  
  
  
rlatinsquare(4) #Obtain L (Trail and Error)

## [,1] [,2] [,3] [,4]  
## [1,] 1 3 2 4  
## [2,] 2 4 3 1  
## [3,] 3 1 4 2  
## [4,] 4 2 1 3

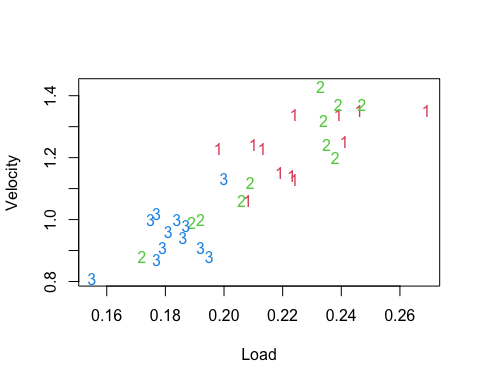
#Obtain L by permuting the columns of B randomly

#Question 3 #Part a: Create a scatterplot of the data, to illustrate the relationships between tree, loading and velocity.

setwd("~/Desktop/UNIMELB 2021 Material/UNIMELB S1 2021 (Currently)/MAST30025/Tutorials /Tutorials/Rfile/data")  
heli = read.csv("heli.csv")  
str(heli)

## 'data.frame': 35 obs. of 3 variables:  
## $ Tree : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ Load : num 0.239 0.208 0.223 0.224 0.246 0.213 0.198 0.219 0.241 0.21 ...  
## $ Velocity: num 1.34 1.06 1.14 1.13 1.35 1.23 1.23 1.15 1.25 1.24 ...

#Actual solution  
plot(Velocity ~ Load, pch=as.character(Tree), col=Tree+1, data=heli)

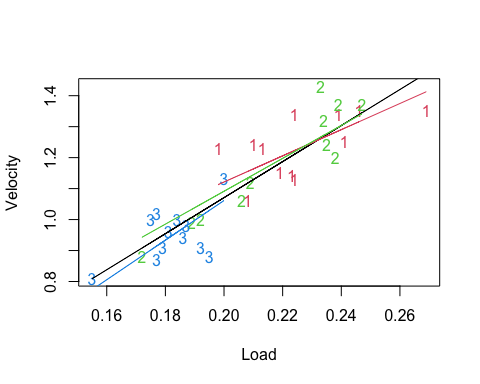


#There appears to be a clear linear relationship between velocity and disk loading. It’s not clear if the tree has an effect, either directly or through interaction.

#Part b: Fit a model to the data. You may use the lm command. You should consider a variety of models and indicate which you prefer and why.

#From my understanding #Using backwards elimation refer from Question 4 Assignment 3 2020, and also it refers from the diagnostic plots from earlier this semester!

#Actual solution   
fullmodel = lm(Velocity~Load + Tree+ Load\*Tree,data = heli)  
model3 = lm(Velocity~Load,data = heli)  
plot(Velocity ~ Load, pch=as.character(Tree), col=Tree+1, data=heli)  
with(heli, lines(Load, fitted(model3)))  
for (i in 1:3) {with(heli, lines(Load[Tree==i], fitted(fullmodel)[Tree==i], col=i+1))}



#Actual solution   
heli$Tree = factor(heli$Tree)  
model3 = lm(Velocity~Load,data = heli)  
summary(model3)

##   
## Call:  
## lm(formula = Velocity ~ Load, data = heli)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.16168 -0.05332 -0.01511 0.05528 0.17086   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.09326 0.10743 -0.868 0.392   
## Load 5.82019 0.51119 11.386 5.7e-13 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.08067 on 33 degrees of freedom  
## Multiple R-squared: 0.7971, Adjusted R-squared: 0.7909   
## F-statistic: 129.6 on 1 and 33 DF, p-value: 5.704e-13

#The test for the interaction term

#Without lm function

str(heli)

## 'data.frame': 35 obs. of 3 variables:  
## $ Tree : Factor w/ 3 levels "1","2","3": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Load : num 0.239 0.208 0.223 0.224 0.246 0.213 0.198 0.219 0.241 0.21 ...  
## $ Velocity: num 1.34 1.06 1.14 1.13 1.35 1.23 1.23 1.15 1.25 1.24 ...

library(MASS)  
library(Matrix)  
y = heli$Velocity  
n = length(heli$Velocity)  
X = matrix(c(rep(1,n),rep(0,n\*7)),n,8)  
X[cbind(1:n,as.numeric(heli$Tree)+1)]=1  
X[,5]=heli$Load  
X[cbind(1:n,as.numeric(heli$Tree)+5)]=heli$Load  
r = rankMatrix(X)[1]  
XtXc = ginv(t(X)%\*%X)  
XtXc

## [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]  
## [1,] 2.6480861 0.3875622 -0.7483181 3.008842 -13.2164161 -0.1136429 4.591751 -17.69452  
## [2,] 0.3875622 8.7193828 -2.2873302 -6.044490 -0.1136429 -39.8765339 8.738308 31.02458  
## [3,] -0.7483181 -2.2873302 6.4476221 -4.908610 4.5917511 8.7383078 -30.465746 26.31919  
## [4,] 3.0088420 -6.0444903 -4.9086100 13.961942 -17.6945243 31.0245832 26.319189 -75.03830  
## [5,] -13.2164161 -0.1136429 4.5917511 -17.694524 67.0244300 -8.0853337 -27.395644 102.50541  
## [6,] -0.1136429 -39.8765339 8.7383078 31.024583 -8.0853337 184.9026225 -31.543453 -161.44450  
## [7,] 4.5917511 8.7383078 -30.4657460 26.319189 -27.3956436 -31.5434527 146.282003 -142.13419  
## [8,] -17.6945243 31.0245832 26.3191892 -75.038297 102.5054073 -161.4445036 -142.134194 406.08410

b = XtXc%\*%t(X)%\*%y  
s2 = sum((y-X%\*%b)^2)/(n-r)  
C = matrix(c(0,0,0,0,0,1,-1,0,0,0,0,0,0,1,0,-1),2,8,byrow = T)  
Fstat = t(b)%\*%t(C)%\*%solve(C%\*%XtXc%\*%t(C))%\*%C%\*%b/2/s2  
pf(Fstat,2,n-r,lower=F)

## [,1]  
## [1,] 0.05010679

#Were given 0.0511 as our p value from the interaction term, going from the first model to the final model using backwards elimation. We have load in our final model. If we consider to keep the interaction term between the Tree and Load we use AIC using stepwise selection.

fullmodel = lm(y~Load + factor(Tree)+ Load\*factor(Tree),data = heli)  
drop1(fullmodel, scope = ~., test = "F")

## Single term deletions  
##   
## Model:  
## y ~ Load + factor(Tree) + Load \* factor(Tree)  
## Df Sum of Sq RSS AIC F value Pr(>F)   
## <none> 0.16549 -175.40   
## Load 1 0.039792 0.20528 -169.85 6.9729 0.01319 \*  
## factor(Tree) 2 0.037553 0.20305 -172.24 3.2903 0.05154 .  
## Load:factor(Tree) 2 0.037949 0.20344 -172.17 3.3250 0.05011 .  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

model2 = lm(y~Load + factor(Tree),data = heli)  
drop1(model2, scope = ~., test = "F")

## Single term deletions  
##   
## Model:  
## y ~ Load + factor(Tree)  
## Df Sum of Sq RSS AIC F value Pr(>F)   
## <none> 0.20344 -172.17   
## Load 1 0.315542 0.51898 -141.39 48.0817 8.884e-08 \*\*\*  
## factor(Tree) 2 0.011322 0.21476 -174.28 0.8626 0.4319   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

model3 = lm(y~Load,data = heli)  
drop1(model3, scope = ~., test = "F")

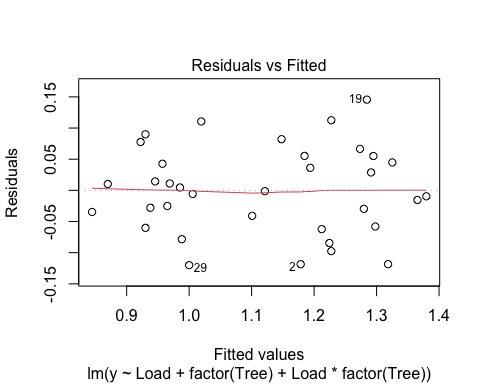
## Single term deletions  
##   
## Model:  
## y ~ Load  
## Df Sum of Sq RSS AIC F value Pr(>F)   
## <none> 0.21476 -174.28   
## Load 1 0.84364 1.05840 -120.45 129.63 5.704e-13 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

summary(model3)

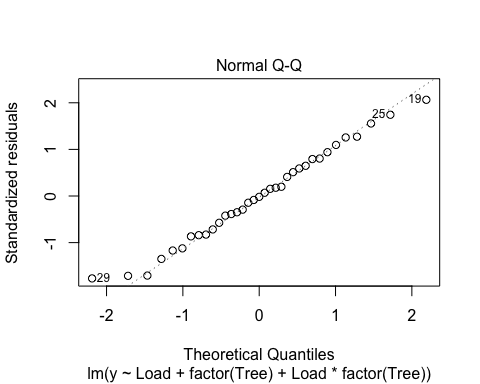
##   
## Call:  
## lm(formula = y ~ Load, data = heli)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.16168 -0.05332 -0.01511 0.05528 0.17086   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.09326 0.10743 -0.868 0.392   
## Load 5.82019 0.51119 11.386 5.7e-13 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.08067 on 33 degrees of freedom  
## Multiple R-squared: 0.7971, Adjusted R-squared: 0.7909   
## F-statistic: 129.6 on 1 and 33 DF, p-value: 5.704e-13

#Using the diagnostic plots using the first model to check if it satisfies our model assumptions

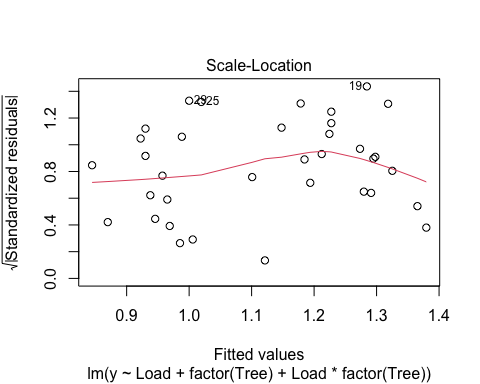
plot(fullmodel,which = 1)



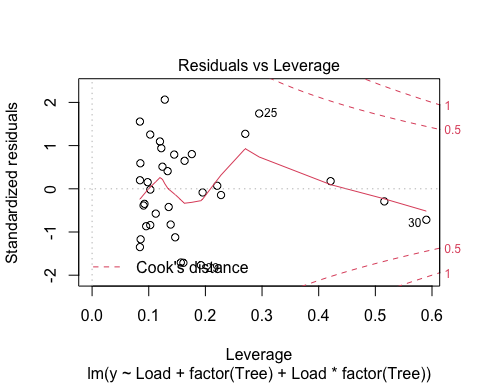
plot(fullmodel,which = 2)



plot(fullmodel,which = 3)

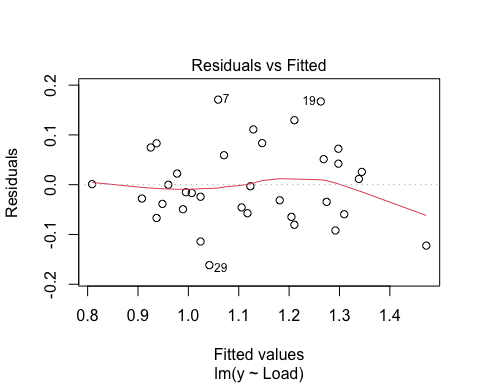


plot(fullmodel,which = 5)

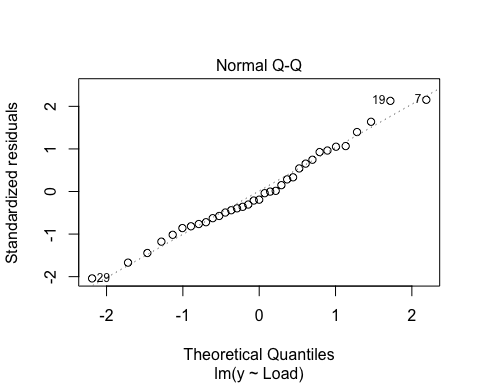


#Similiar thing using the final model

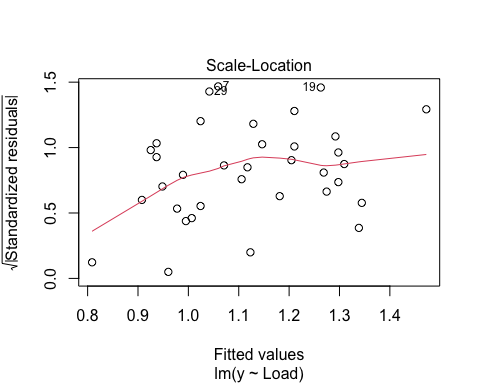
plot(model3,which = 1)



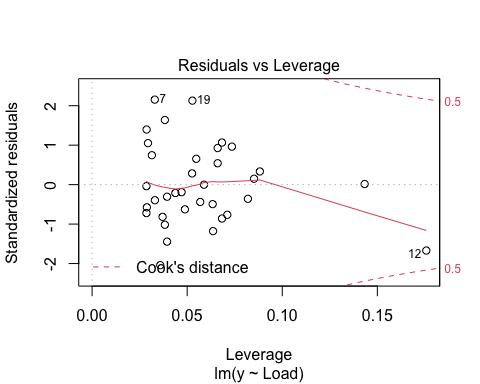
plot(model3,which = 2)



plot(model3,which = 3)



plot(model3,which = 5)



#Demonstrator Comment: The residuals look slightly better for model1, in particular the QQ plot is a little nicer, but the residuals for model3 are still quite acceptable: they are reasonably normal, there are no outliers, and they look homoskedastic. For the purposes of this question, either model1 or model3 would be acceptable. If forced to choose I would go for model1.

#Part c: What assumptions are you making with your model? #Provide some diagnostic plots to indicate that these assumptions are justified

#It seems reasonable that all the plots from above are normally distributed #Model 1: #Residuals vs. fitted values: Most points have large residuals, seems to be unbiased and there is zero correlation. May imply that all the residuals are independant.

#Normal QQ Plot: There consists a small number of outliners, Looks normally distributed

#Absolute Values of standardised residuals against the fitted values: #There are some points with high residuals, variance is not constant hence there is Heteroskedasticity.

#leverge vs. standardarised residuals: #Unequal variance, there is high leverge, Lower Cook’s distance and some points may be correlated!

#Final Model: #Residuals vs. fitted values: #Most points have large residuals, seems to be biased and there may be correlated indiciate that these residuals are not independant.

#Normal QQ Plot:There consists a small number of outliners, Looks normally distributed

#Absolute Values of standardised residuals against the fitted values: #There are some points with high residuals, variance is not constant hence there is Heteroskedasticity.

#leverge vs. standardarised residuals: #Unequal variance, there is high leverge, Lower Cook’s distance and some points may be correlated!

#Part d: Is the effect of the tree significant? #From all three models except the last one, all models with p values are #0.05154 and 0.4319 from model1 and model2,both indeed are not significant above alpha=0.05 (Assume 5% it is always the case and yet very common)!

#Part e: If you want to get a slow samara, which tree would you take it from? Explain how your answer is consistent to your answer to the previous question.

#This tests your understanding from the slopes/gradients in your plot you see in part b: After calcalating each gradient the slowest (lowest magnitude/gradient) would be Tree 3.

#Part f: Give a 95% prediction interval for the velocity of a samura from tree 2 with a loading of 0.2

#My Attempt, what is wrong with my hypothesis?!  
tt = c(0,0,0,0,0.2,0,0,0)  
halfwidth = qt(0.975,df = n-r)\*sqrt(s2)\*sqrt(1+t(tt)%\*%XtXc%\*%tt)  
c(tt%\*%b-halfwidth,tt%\*%b+halfwidth)

## [1] 0.3907443 0.9835919

#Actual solution  
newdata = data.frame(Tree=factor(2,levels=levels(heli$Tree)), Load = 0.2)  
predict(fullmodel, newdata, interval = "prediction")

## fit lwr upr  
## 1 1.060123 0.8951459 1.225101