Chapter 3: CR Design Analysis with R

Jixiang Wu
Associate Professor of Quantitative Genetics/Biostatistics
Plant Science Department
South Dakota State University
Email: Jixiang.wu@sdstate.edu

Phone: (605)688-5947

Introduction

In this document I will provide some R scripts specifically used for this chapter with a completely randomized (CR) design. These R scripts will show you how to generate a CR design and detailed data analyses.

Generate a Completely Randomized Design

It is very easy to use R scripts to generate any completely randomized (CR) experimental design with one factor. There are many ways to write R codes. Suppose there are four treatment levels, named as "A", "B", "C", and "D" with three replications.

The following R scripts are just an example to generate a CR design with four treatment levels and three replications.

```
TRT=rep(c("A","B","C","D"),3)
TRT
```

```
##  [1] "A" "B" "C" "D" "A" "B" "C" "D" "A" "B" "C" "D"
```

```
N=1:12
Plot=sample(N)  # A randomization of 12 experimental units
Plot
```

```
##  [1]  9  3  7  2 12  8  4  6  5 11  1 10
```

The identities in the vector **Plot** are identities of experimental units to be assigned to these treatments.

```
desgn=data.frame(TRT,Plot)
desgn
```

TRT	*Plot*
A	9
B	3
C	7
D	2
A	12
B	8
C	4
D	6
A	5
B	11
C	1
D	10

```
order=order(TRT)
order
```

```
##  [1]  1  5  9  2  6 10  3  7 11  4  8 12
```

```
desgn=desgn[order,]
desgn
```

	TRT	*Plot*
1	A	9
5	A	12
9	A	5
2	B	3
6	B	8
10	B	11
3	C	7
7	C	4
11	C	1
4	D	2
8	D	6
12	D	10

The above table shows that treatment A is assigned to plot 9, 12, and 5; treatment B assigned to plot 3, 8, and 11; treatment C assigned to plot 7, 4, and 1; and treatment D assigned to plot 2, 6, and 10.

Please remember this principle can be applied to another factor with different numbers of treatments.

Analysis for a CR design

Now we will start to run a CR design analysis with an actual data set. Being able to use the data and generate the results in the demonstration. You will need to install an R package, **coursedata**, containing the data set steak and many other data sets. This R package (a very small data set package only developed for this course) is available in D2L now (content→R Resources→coursedata_1.0.zip. Please remember that this R package could be updated so you need update this package as this course goes and you will be notified if any changes are made. It is important to know how to successfully install an updated R package manually. First you need delete any old versions of an R package before you download an updated one. Then following the instructions mentioned in our chapter 2.

As mentioned in our PPT slides and the course book authored by Kuehl, 12 almost identical meat piecies were randomly assigned to four meat protection methods. So certainly this is a CR design. We will use this data to demonstrate various data analyses.

First we will load this data set from this R package: coursedata. Before loading, you need install this package locally. The following R scripts show how to load the data set.

```
require(coursedata)

## Loading required package: coursedata

data(steak)
```

The code require (coursedata) means that the package coursedata will be loaded after installation. If it is not installed, you will not able to load it or you will have an error. If you don't know how to run a specific code or a junk of R codes, you need to get familiar with this before you can run any specific codes.

Once you load the data into your R platform successfully. The data should like as follows. Please compare the following data with those in the book authored by Kuehl.

```
steak
```

Steak	Treatment	Log.count.cm2.
1	Commercial	7.66
6	Commercial	6.98
7	Commercial	7.80

12	Vacuum	5.26
5	Vacuum	5.44
3	Vacuum	5.80
10	Mixed Gas	7.41
9	Mixed Gas	7.33
2	Mixed Gas	7.04
8	CO2	3.51
4	CO2	2.91
11	CO2	3.66

Population Mean and Variance

First we just run a population mean model for this data. You can just use the built-in function *mean* to calculate the population mean. Remember this is the most reduced ANOVA model for this data set, meaning there were no treatment effects. You can also calculate the population variance by using the built-in function *var*.

```
trt=steak$Treatment
y=steak$Log.count.cm2.
mu=mean(y)
v=var(y)
data.frame(mu,v)
```

Mu	v
5.9	3.073

The population mean and variance are 5.9 and 3.073 for this experiment.

Test Variance Homogeneity

ANOVA analysis normally is quite robust. Sometimes we need test if variance is homogeneous among treatments. R has a biult-in function called **fligner.test** to run the test as follows. Please take your own time to learn more information regarding Flinger-Killeen test.

```
fligner.test(y~trt)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  y by trt
## Fligner-Killeen:med chi-squared = 0.51, df = 3, p-value = 0.9167
```

Based on the above Flinger-Killeen chi-square test, the probability value is 0.9167, which indicates the random error variance is homogeneous among these four treatments.

We can calculate the variance for each treatment. The following R scripts show you how to do it easily. You may notice that an R function *tapply* is used here. The *tapply* function can calculate a group of values such as mean and variance. In the following R script, three components normally are required. The first part is a response variable like y used here. The second part is treatment vector like trt used here. Both y and trt should have the same dimensions. The third part is a function part. Here we will use the function *var* to calculate the variance for each of four groups/levels.

```
vt=tapply(y,trt,var)
data.frame(vt)
```

	vt
CO2	0.1575
Commercial	0.1924
Mixed Gas	0.0379
Vacuum	0.0756

Treatment means

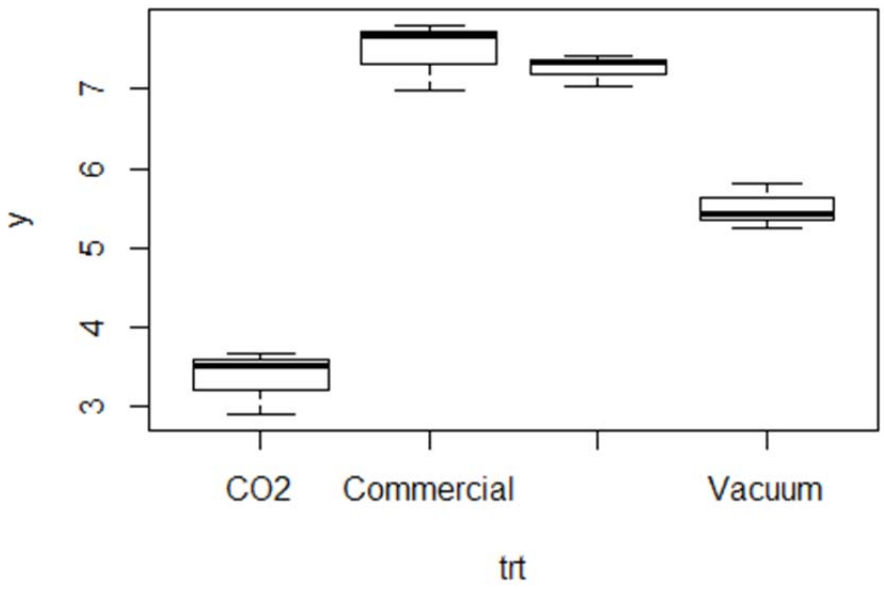
You can calculate treatment means easily with *tapply*. Again we will use the *tapply* to calculate treatment means and you will see that three components are included.

```
mt=tapply(y,trt,mean)
data.frame(mt)
```

	mt
CO2	3.36
Commercial	7.48
Mixed Gas	7.26
Vacuum	5.50

We can now make the boxplots by running the following R script:

```
plot(y~trt)
```



The output of this program is shown above. The plot tells a little clearer information on these treatment means, which look quite different.

Manually calculate SS, MS, and F-test

Now we will learn how to run ANOVA analysis for a CR design data set. ANOVA analysis often requires sums of squares (SS), mean squares (MS), and degrees of freedom (DF) before conducting F tests. It is very easy to manually calculate SS, MS, and to run F-test for a CR design data set. Please pay a little attention that we will frequently use the *tapply* function as mentioned above.

Step 1: Calculate population mean

```
mu=mean(y)  ## Calculate population mean
mu

## [1] 5.9
```

Step 2: Calculate treatment means

```
means=tapply(y,trt,mean)
means

##       CO2 Commercial  Mixed Gas     Vacuum
##      3.36       7.48       7.26       5.50
```

Step 3: Calculate within treatment variance

```
vars=tapply(y,trt,var)
vars
```

```
##         CO2 Commercial  Mixed Gas     Vacuum
##     0.1575     0.1924     0.0379     0.0756
```

Step 4: Measure sample size for each treatment

```
cts=tapply(y,trt,length)
cts
```

```
##         CO2 Commercial  Mixed Gas     Vacuum
##           3          3          3          3
```

Step 5: Calculate Sums of squares for error and treatment

Total sums of square: SSTO

```
SST0=sum((y-mean(y))^2)
SST0
```

```
## [1] 33.8
```

Sums of squares for errors

```
SSE=sum(vars*(cts-1))
SSE
```

```
## [1] 0.9268
```

Sums of squares for treatment

```
SST=sum(cts*(means-mean(y))^2)
SST
```

```
## [1] 32.87
```

Step 6: Calculate degrees of freedom

```
dfe=sum(cts-1)  ## DF for error
dfe
```

```
## [1] 8
```

```
dft=length(cts)-1 ## DF for treatment
dft
```

```
## [1] 3
```

Step 7: Calculate mean squares for treatment and error

```
mst=SST/dft   ## Mean square for treatment
mst

## [1] 10.96

mse=SSE/dfe   ## mean sqaure for error
mse

## [1] 0.1158
```

Step 8: Conduct a F test

```
f.ratio=mst/mse   ## Calculate f value
f.ratio

## [1] 94.58

p_value=1-pf(f.ratio,dft,dfe)  ##calculate p value under H0
p_value

## [1] 1.376e-06
```

Above p value shows that the probability is much less than 0.05, indicating that probability for these four treatment means being the same is 1.376e-06. On the other hand, it is very unlikely that these four treatments are the same → we can reject the null hypothesis and accept the alternative hypothesis→which means that at least two treatments are significantly different.

Develop our own one-way ANOVA function

Though CR design data sets can be quite different from one to another, the data analysis procedure is the same. Therefore we can develop a basic ***R function,*** which can be used to run one-way ANOVA analysis repeatedly. The following R function is designated for one-way ANOVA analysis. All results are stored into a list and can be returned as a list.

```
ss.one.way=function(y,trt){
   means=tapply(y,trt,mean)
   vars=tapply(y,trt,var)
   cts=tapply(y,trt,length)
   SST0=sum((y-mean(y))^2)
   SSE=sum(vars*(cts-1))
   dfe=sum(cts-1)
   SST=sum(cts*(means-mean(y))^2)
   dft=length(cts)-1
   mst=SST/dft
   mse=SSE/dfe
   f.ratio=mst/mse
   p_value=1-pf(f.ratio,dft,dfe)
```

```
   res=list(SST0=SST0,SST=SST,SSE=SSE,dft=dft,dfe=dfe,p_value=p_value)
   return(res)
}
```

If you run the function ss.one.way itself, you will see all the scripts for this tiny R function as shown below.

```
ss.one.way
```

```
## function(y,trt){
##    means=tapply(y,trt,mean)
##    vars=tapply(y,trt,var)
##    cts=tapply(y,trt,length)
##    SST0=sum((y-mean(y))^2)
##    SSE=sum(vars*(cts-1))
##    dfe=sum(cts-1)
##    SST=sum(cts*(means-mean(y))^2)
##    dft=length(cts)-1
##    mst=SST/dft
##    mse=SSE/dfe
##    f.ratio=mst/mse
##    p_value=1-pf(f.ratio,dft,dfe)
##    res=list(SST0=SST0,SST=SST,SSE=SSE,dft=dft,dfe=dfe,p_value=p_value)
##    return(res)
## }
```

Run this tiny R function

We can use the above R function ss.one.way to conduct ANOVA analysis for our steak data set.

```
res=ss.one.way(y,trt)

res
```

```
## $SST0
## [1] 33.8
##
## $SST
## [1] 32.87
##
## $SSE
## [1] 0.9268
##
## $dft
## [1] 3
##
## $dfe
## [1] 8
##
```

```
## $p_value
## [1] 1.376e-06

result=unlist(res)

result

##      SST0       SST       SSE       dft       dfe   p_value
## 3.380e+01 3.287e+01 9.268e-01 3.000e+00 8.000e+00 1.376e-06
```

Now you can try different data sets or assign observation vector to variable *y* and treatment vector to variable *trt*.

Use the Built-in Functions: aov and lm

Above R scripting needs some skills or experience. R platform actually provides built-in R functions to run various ANOVA analyses. There are two powerful built-in R functions: *aov* and *lm*, which can be used for various linear model analyses such as multiple linear regression analysis and almost all experimental designs. The key is we need give a right linear model. Please use ??aov or ??lm to get more detailed information for these two built-in functions. The function *aov* focuses on SS, MS, and F-tests while *lm* focuses on estimations of effects and statistical tests for these effects. You will be amazed by the great functionality and flexibility of these two functions and the model system principles are the same for these two functions. The use of these two functions in CR data analysis is just a simple example in many linear model analyses or design of experiments.

aov

The general form is: aov(response ~ factor, data=data_name), where response represents the response variable and factor the variable that separates the data into groups. Both variables should be contained in the data frame called data_name. Once the ANOVA model is fit, one can look at the results using the summary() function. This produces the standard ANOVA table.

```
aov(y~trt)

## Call:
##    aov(formula = y ~ trt)
##
## Terms:
##                     trt Residuals
## Sum of Squares  32.87      0.93
## Deg. of Freedom     3         8
##
## Residual standard error: 0.3404
## Estimated effects may be unbalanced

summary(aov(y~trt))
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## trt          3   32.9   10.96    94.6 1.4e-06 ***
## Residuals    8    0.9    0.12
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

When you report your results, you can not just print out these outputs. You may need to make appropriate tables to summarize the outputs, especially for many traits being analyzed. Remember there are many possible ways to report or summarize your results based the need and/or the amount of information you obtain.

lm

```
mod1=lm(y~trt)
summary(mod1)
```

```
##
## Call:
## lm(formula = y ~ trt)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -0.500 -0.225  0.110  0.210  0.320
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)      3.360      0.197    17.1  1.4e-07 ***
## trtCommercial    4.120      0.278    14.8  4.2e-07 ***
## trtMixed Gas     3.900      0.278    14.0  6.5e-07 ***
## trtVacuum        2.140      0.278     7.7  5.7e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.34 on 8 degrees of freedom
## Multiple R-squared:  0.973,  Adjusted R-squared:  0.962
## F-statistic: 94.6 on 3 and 8 DF,  p-value: 1.38e-06
```

You could be puzzled with above results: there is no CO2 effect. Actually here the intercept estimate is an estimate of CO2 treatment mean, while all other effects are just deviations from the CO2 mean. The major reason is that the data analysis is matrix based and the matrix is not fully ranked. It is a little beyond the scope of this course.

Use the anova function to test the two models

There is another built-in function call anova to compare two statistical models, for example a reduced model and full model. The following R scripts shows how to compare two linear

models. We still use the above CR data set as our demonstration.The mod0 in the following is a population model which assumes no treatment effects. Thus, it is a reduced model.

```
mod0=lm(y~1) ## population model
summary(mod0)

##
## Call:
## lm(formula = y ~ 1)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -2.99  -1.04   0.49   1.45   1.90
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    5.900      0.506    11.7  1.6e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.75 on 11 degrees of freedom

anova(mod0,mod1)
```

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
11	33.7996	NA	NA	NA	NA
8	0.9268	3	32.87	94.58	0

You may find that the F test results are equivalent to the above results.

Multiple Comparisons

The ANOVA F-test answers the question whether there are significant differences in the K population means. However, it does not provide us with any detailed information about how they differ. Therefore when you reject H0 in ANOVA, additional analyses are normally required to determine what is driving the difference in means. The function **pairwise.t.test** computes the pair-wise comparisons between group means with corrections for multiple testing. The general form is **pairwise.t.test**(reponse, factor, p.adjust = method, alternative = c("two.sided", "less", "greater")). Here response is a vector of observations (the response variable), factor a list of factors and p.adjust is the correction method (e.g., "Bonferroni").

```
a1 <- aov(y~trt)
summary(a1)

##             Df Sum Sq Mean Sq F value  Pr(>F)
## trt          3   32.9   10.96    94.6 1.4e-06 ***
## Residuals    8    0.9    0.12
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
pairwise.t.test(y, trt, p.adj = "none")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  y and trt
##
##            CO2     Commercial Mixed Gas
## Commercial 4.2e-07 -          -
## Mixed Gas  6.5e-07 0.45141    -
## Vacuum     5.7e-05 1.0e-04    0.00022
##
## P value adjustment method: none
```

```r
pairwise.t.test(y, trt, p.adj = "bonf")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  y and trt
##
##            CO2     Commercial Mixed Gas
## Commercial 2.5e-06 -          -
## Mixed Gas  3.9e-06 1.00000    -
## Vacuum     0.00034 0.00060    0.00135
##
## P value adjustment method: bonferroni
```

```r
pairwise.t.test(y, trt, p.adj = "holm")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  y and trt
##
##            CO2     Commercial Mixed Gas
## Commercial 2.5e-06 -          -
## Mixed Gas  3.2e-06 0.45141    -
## Vacuum     0.00023 0.00030    0.00045
##
## P value adjustment method: holm
```

Above pair-wise comparisons suggested that the treatment CO2 is significantly different from all other three treatments. The treatments Mixed Gas and Vacuum also showed significant difference.

Other multiple comparisons

Tukey's method

Another multiple comparisons procedure is Tukey???s method (a.k.a. Tukey's Honest Significance Test). The function TukeyHSD() creates a set of confidence intervals on the differences between means with the specified family-wise probability of coverage. The general form is TukeyHSD(x, conf.level = 0.95). Here x is a fitted model object (e.g., an aov fit) and conf.level is the confidence level.

```
a1 <- aov(y~trt)
TukeyHSD(a1, conf.level = 0.95)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = y ~ trt)
##
## $trt
##                        diff   lwr   upr  p adj
## Commercial-CO2         4.12  3.23  5.01 0.0000
## Mixed Gas-CO2          3.90  3.01  4.79 0.0000
## Vacuum-CO2             2.14  1.25  3.03 0.0003
## Mixed Gas-Commercial -0.22 -1.11  0.67 0.8564
## Vacuum-Commercial    -1.98 -2.87 -1.09 0.0005
## Vacuum-Mixed Gas     -1.76 -2.65 -0.87 0.0010
```

Least significant difference test

This method is built-in the package agricolae. By using this method, you need to provide degrees of freedom and mean square for errors. In the steak example, the DF and MSE are 8 and 0.116 as shown in the following R scripts.

```
require(agricolae)
```

```
## Loading required package: agricolae
```

```
## Warning: package 'agricolae' was built under R version 3.0.3
```

```
res=LSD.test(y, trt, DFerror=8, MSerror=0.116)
res
```

```
## $statistics
##   Mean    CV MSerror    LSD
##    5.9 5.773   0.116 0.6413
##
## $parameters
```

```
##   Df ntr t.value
##    8   4   2.306
##
## $means
##                y    std r   LCL   UCL  Min  Max
## CO2         3.36 0.3969 3 2.907 3.813 2.91 3.66
## Commercial 7.48 0.4386 3 7.027 7.933 6.98 7.80
## Mixed Gas  7.26 0.1947 3 6.807 7.713 7.04 7.41
## Vacuum     5.50 0.2750 3 5.047 5.953 5.26 5.80
##
## $comparison
## NULL
##
## $groups
##          trt means M
## 1 Commercial  7.48 a
## 2 Mixed Gas   7.26 a
## 3 Vacuum      5.50 b
## 4 CO2         3.36 c
```

Linear Mixed Model Analysis

As we will talk later, CR design model is a specific case in the domain of linear mixed models. We may use linear mixed model approaches to analyzing this data set. Before you can use specific linear mixed model approaches, you need install some R packages. You can follow some instructions to download and install the **minque** package which is available online and D2L as well. Before you use the functions in minque, you need load it to your R environment by using the following R code: **require**(minque)

```
require(minque)
```

```
## Loading required package: minque
```

```
## Warning: package 'minque' was built under R version 3.0.3
```

```
## Loading required package: klaR
```

```
## Warning: package 'klaR' was built under R version 3.0.2
```

```
## Loading required package: MASS
##
## Attaching package: 'MASS'
##
## The following object is masked _by_ '.GlobalEnv':
##
##     ginv
##
## Loading required package: agridat
```

```
## Warning: package 'agridat' was built under R version 3.0.2
```

```
## Loading required package: grid
## Loading required package: lattice
## Loading required package: reshape2

## Warning: package 'reshape2' was built under R version 3.0.2

## Loading required package: Matrix

## Warning: package 'Matrix' was built under R version 3.0.2

res=lmm(y~trt)[[1]]
data.frame(res$Var)
```

	y.Est	y.SE	y.Chi_sq	y.P_value
V(e)	0.1158	0.0579	4	0.0228

```
data.frame(res$FixedEffect)
```

	y.Est	y.SE	y.z_value	y.P_value
mu	5.90	0.2686	21.965	0.000
trt(Commercial)	1.58	0.3024	5.224	0.000
trt(Vacuum)	-0.40	0.3024	-1.323	0.186
trt(Mixed Gas)	1.36	0.3024	4.497	0.000
trt(CO2)	-2.54	0.3024	-8.399	0.000

The results generated by minque approach are slightly different from the **lm** analysis. In minque package, all estimated effects are deviations from the population mean, however, in lm analysis, the first estimated effect is forced to 0 and the mu is the first treatment mean, all other effects are deviations from the first treatment mean. The results should be compariable and equivalent to each other. You may compare these results one-by-one to get more detailed understanding.