Chapter 4: Data Analysis for RCB designs with R

Jixiang Wu

Associate Professor of Quantitative Genetics/Biostatistics

Agronomy, Horticulture, and Plant Science Department

South Dakota State University

February 14, 2017

Introduction

One of the three important principals for an experimental design is local control since either experimental units or conditions may not be uniform. Randomized complete block design is one of the most popular experimental designs used for local control.

Generate a RCB Design

It seems very easy to generate any RCB design by using R. The following R scripts show how to generate a RCB deisgn.

```
r=5  # 5 blocks
t=10 # 10 treatments
RCB=matrix(0, r, t)
for(i in 1:r)RCB[i,]=sample(1:t)
#rownames(RCB)=c("Block","Treatment")
data.frame(RCB)
```

X1	X2	X3	X4	X5	X6	X7	X8	X9	X10
8	6	7	10	4	2	5	9	3	1
3	10	4	8	7	9	2	1	6	5
10	3	8	9	6	5	4	2	7	1
10	2	1	7	6	5	4	8	3	9
8	10	5	7	9	6	4	2	3	1

Load an RCB data set

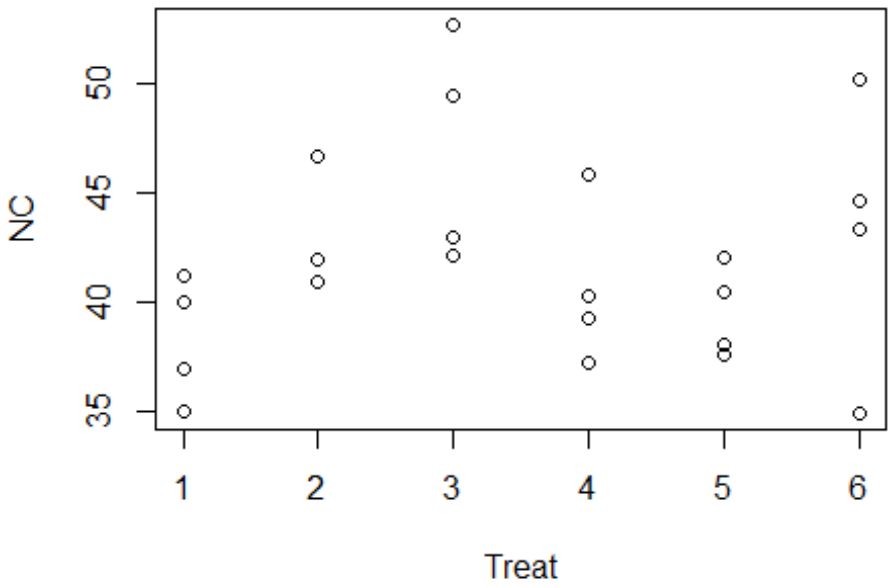
We may read a read data file from a hard drive using the following R codes.

```
require(coursedata)
```

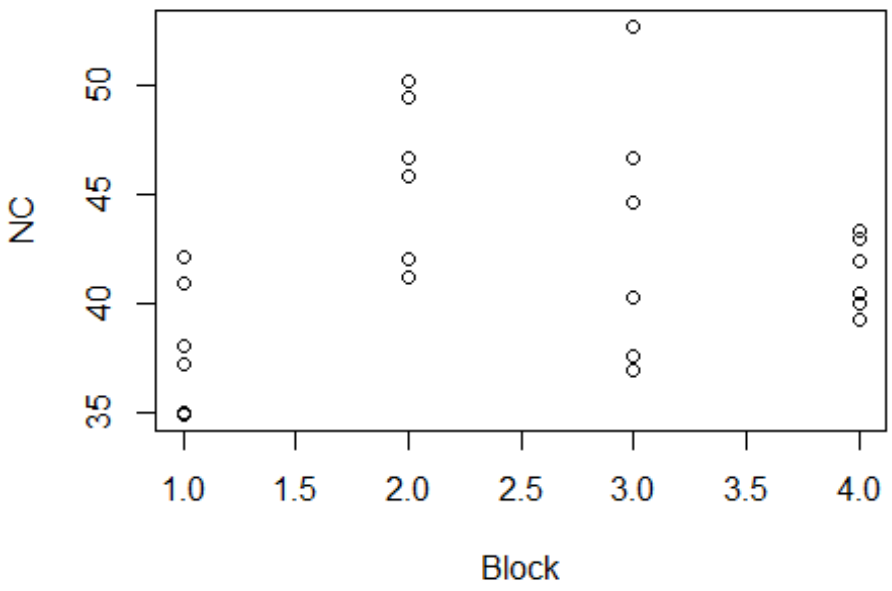
```
## Loading required package: coursedata

data(wheatrcb)
wheat=wheatrcb
```

We can visualize the data by using the following R codes

```
plot(NC~Treat,data=wheat)
```

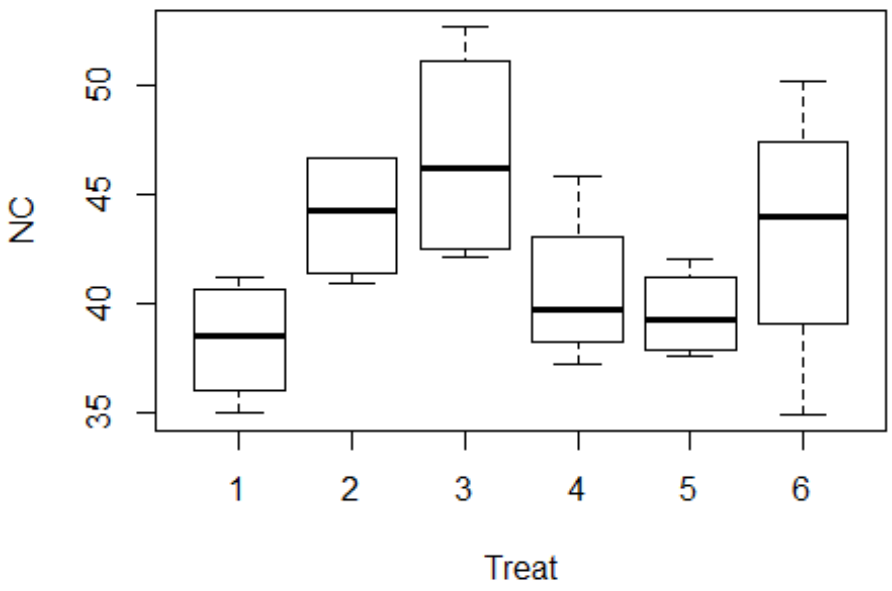



```
plot(NC~Block,data=wheat)
```

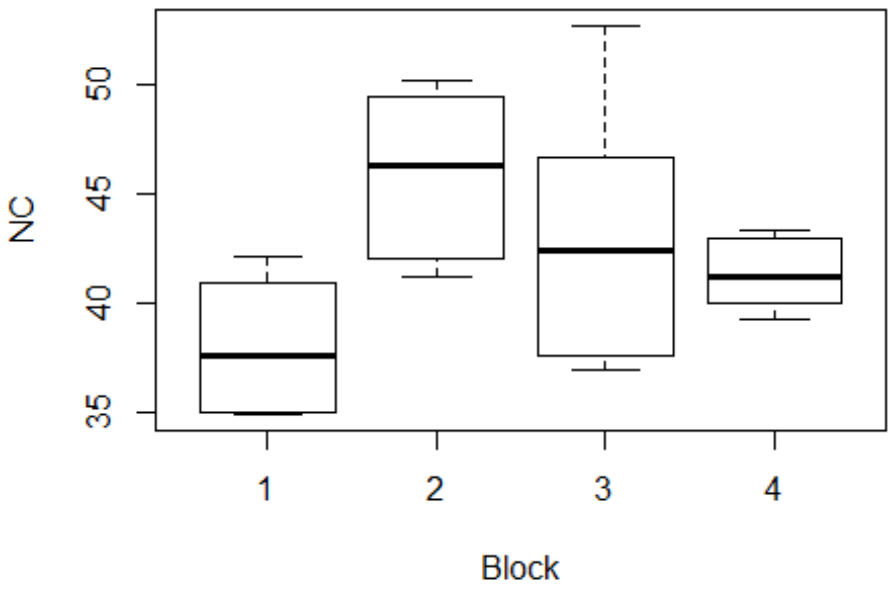


The above R codes treat Treat and Block as numerical values. So we need convert these two variables to categorical variables. Here we are using a very useful function, transform, to change the classes of variables with in a data frame.You will see some different results after variable conversion.

```
wheat=transform(wheat,Treat=factor(Treat),Block=factor(Block))
plot(NC~Treat,data=wheat)
```




```r
plot(NC~Block,data=wheat)
```



Then we use aov to run anova analysis when both Treat and Block are not converted to the categorical variables.

```
require(coursedata)
data(wheatrcb)
wheat=wheatrcb
mod=aov(NC~Treat+Block,data=wheat)
a=summary(mod)
```

If you pay a little attention to the anova results, the degrees of freedom for both treatment and block are 1. It is because the codes used for these two effects are integers or numerical rather than categorical variables. Using str(datafile) you can check the class of each variable in a datafile.

```
str(wheat)
```

```
## 'data.frame':    24 obs. of  3 variables:
##  $ Treat: int  2 5 4 1 6 3 1 3 4 6 ...
##  $ Block: int  1 1 1 1 1 1 2 2 2 2 ...
##  $ NC   : num  40.9 38 37.2 35 34.9 ...
```

In order to obtain appropriate results, we first need to factorize these two factors: treatment and block. Please read the following R scripts as your reference to try. You will see the big differences for block and treatment after they are converted to factors.

```
wheat=transform(wheat,Treat=factor(Treat),Block=factor(Block))
str(wheat)
```

```
## 'data.frame':    24 obs. of  3 variables:
##  $ Treat: Factor w/ 6 levels "1","2","3","4",..: 2 5 4 1 6 3 1 3 4 6 ...
##  $ Block: Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 2 2 2 2 ...
##  $ NC   : num  40.9 38 37.2 35 34.9 ...
```

```
wheat
```

Treat	Block	NC
2	1	40.89
5	1	37.99
4	1	37.18
1	1	34.98
6	1	34.89
3	1	42.07
1	2	41.22
3	2	49.42
4	2	45.85
6	2	50.15
5	2	41.99

2	2	46.69
6	3	44.57
3	3	52.68
5	3	37.61
1	3	36.94
2	3	46.65
4	3	40.23
2	4	41.90
4	4	39.20
6	4	43.29
5	4	40.45
3	4	42.91
1	4	39.97

```
head(wheat)
```

Treat	Block	NC
2	1	40.89
5	1	37.99
4	1	37.18
1	1	34.98
6	1	34.89
3	1	42.07

Now you may see the results look different after data conversion. The SS, MS, DF, and f-tests are totally different compared to the data analysis shown above. Therefore, it is very important to check and/change the type of each variable before you use the built-in functions like aov or lm.

```
mod=aov(NC~Treat+Block,data=wheat)
summary(mod)
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## Treat        5  201.3   40.26   5.592 0.00419 **
## Block        3  197.0   65.67   9.120 0.00112 **
## Residuals   15  108.0    7.20
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Based on the above ANOVA results, we can conclude that both treatment and block effects are significant. Therefore, multiple mean comparsons among treatments are needed, but we will show the multiple comparisons among treatments later.

You may treat this RCB design data set to a CR design data set. By doing so, you can use a reduced anova model to run the data analysis, with only treatment effect included in the model.

```
mod=aov(NC~Treat,data=wheat)  # this is a reduced model with treatment effect
but without blocking
summary(mod)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## Treat        5  201.3   40.26   2.376 0.0802 .
## Residuals   18  305.0   16.95
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Without including block effects in the model, you will see that the MSE increased from 7.2 to 16.9, more than the double. It is very reasonable that the F-test for treatment effect is not significant due to a large MSE using this reduced model for data analysis.

You may also run another reduced model with only block effect included. Again, the MSE for this reduced model slightly increased; however, the F-test for block effect was significant, but not so significant as in the full model.

```
mod=aov(NC~Block,data=wheat)
summary(mod)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## Block        3  197.0   65.67   4.246 0.0178 *
## Residuals   20  309.3   15.47
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Relative efficiency

It is very easy to calculate relative efficient (RE) using R from ANAVA analysis of different models. We just focus on using mean square error (MSE) from a full model and a reduced model.The MSE calculated from the first model is from a full model and the MSE from the second model is a reduced model (a CR design). We can extract MSE from the results obtained by aov analysis for these two models. Please pay a little attention on how I extracted a MSE from an AOV analyis: to find the index to locate MSE from each ANOVA table so that MSE from an ANOVA table can be automatically located.

```
require(coursedata)
data(wheatrcb)
wheat=wheatrcb
wheat=transform(wheat,Treat=factor(Treat),Block=factor(Block))
# A full model
mod1=aov(NC~Treat+Block,data=wheat)
a=summary(mod1)
dfe1=mod$df.residual  ## degree of freedom for MSE
```

```
rn1=nrow(a[[1]])      ## which row where MSE is stored
mse1=a[[1]][rn1,3]  ## Locate the MSE from this ANOVA table

# A reduced model
mod2=aov(NC~Treat,data=wheat)
a=summary(mod2)
dfe2=mod2$df.residual ## degree of freedom for MSE
rn2=nrow(a[[1]]) ## which row where MSE is stored
mse2=a[[1]][rn2,3] ## Locate the MSE from this ANOVA table

# Calculate relative efficiency (RE)
RE=mse2/mse1
RE

## [1] 2.353307
```

The RE calculated from above is 2.35, showing that the RCB design for the wheat experiment is estimated to be slightly more than twice as efficient as a CR design.Or we can conlude that the CR design will require twice replications for each treatment as compared with a RCB design. We can also conclude that an RCB design is highly needed to reduce experimental errors.

Pair-wise mean comparison

Once a F-test for treatment effects is significant, it is very reasonable to conduct pair-wise mean comparisons between treatments. Since the model for an RCB design is different from that for a CR design, the procedures in a CR design may not all work for RCB design or other complicated designs because more factors are used for modelling. Now let us try LSD test among these treatments. Again, we need load the package agricolae before we can use the function LSD.test. Again, it is very very important to provide the right MSE and its degrees of freedom (DF).Using the following R codes can automatically locate the right MSE and its corresponding DF.

LSD test

```
require(agricolae)

## Loading required package: agricolae

require(coursedata)
data(wheatrcb)
wheat=wheatrcb
wheat=transform(wheat,Treat=factor(Treat),Block=factor(Block))  ## convert th
e variables
mod=aov(NC~Treat+Block,data=wheat)
a=summary(mod1)
dfe=mod$df.residual
rn=nrow(a[[1]])
mse=a[[1]][rn,3]
```

```
res=LSD.test(wheat$NC, wheat$Treat, DFerror=dfe, MSerror=mse)

res
```

```
## $statistics
##       Mean       CV  MSerror      LSD
##   42.07167 6.378131 7.200561 4.044299
##
## $parameters
##   Df ntr t.value alpha       test      name.t
##   15   6 2.13145  0.05 Fisher-LSD wheat$Treat
##
## $means
##   wheat$NC      std r      LCL      UCL   Min   Max
## 1  38.2775 2.839324 4 35.41775 41.13725 34.98 41.22
## 2  44.0325 3.073352 4 41.17275 46.89225 40.89 46.69
## 3  46.7700 5.129659 4 43.91025 49.62975 42.07 52.68
## 4  40.6150 3.712811 4 37.75525 43.47475 37.18 45.85
## 5  39.5100 2.078012 4 36.65025 42.36975 37.61 41.99
## 6  43.2250 6.304525 4 40.36525 46.08475 34.89 50.15
##
## $comparison
## NULL
##
## $groups
##   trt   means   M
## 1   3 46.7700   a
## 2   2 44.0325  ab
## 3   6 43.2250 abc
## 4   4 40.6150 bcd
## 5   5 39.5100  cd
## 6   1 38.2775   d
```

Duncan Test

You can also try Duncan test for pair-wise comparisons among treatments.Again, you need to define which factor to be used for pair-wise comparisons.The duncan.test function needs the aov results as a list for data input for the first component. The second component is the variable name of the treatment.

```
require(agricolae)
require(coursedata)
data(wheatrcb)
wheat=wheatrcb
wheat=transform(wheat,Treat=factor(Treat),Block=factor(Block))
mod=aov(NC~Treat+Block,data=wheat)
res=duncan.test(mod, "Treat",main="Nitrogen content with different treatment"
)
res
```

```
## $statistics
##       Mean       CV  MSerror
##   42.07167 6.378131 7.200561
##
## $parameters
##   Df ntr alpha   test name.t
##   15   6  0.05 Duncan  Treat
##
## $Duncan
##      Table CriticalRange
## 2 3.014325      4.044299
## 3 3.159826      4.239517
## 4 3.250248      4.360835
## 5 3.311848      4.443484
## 6 3.356024      4.502754
##
## $means
##        NC      std r   Min   Max
## 1 38.2775 2.839324 4 34.98 41.22
## 2 44.0325 3.073352 4 40.89 46.69
## 3 46.7700 5.129659 4 42.07 52.68
## 4 40.6150 3.712811 4 37.18 45.85
## 5 39.5100 2.078012 4 37.61 41.99
## 6 43.2250 6.304525 4 34.89 50.15
##
## $comparison
## NULL
##
## $groups
##   trt   means   M
## 1   3 46.7700   a
## 2   2 44.0325  ab
## 3   6 43.2250 abc
## 4   4 40.6150 bcd
## 5   5 39.5100  cd
## 6   1 38.2775   d
```

Tukey Test

You can try Tukey test for pair-wise comparisons among treatments as well. The procedure is very similar to the Duncan test.

```
require(agricolae)
require(coursedata)
data(wheatrcb)
wheat=wheatrcb
wheat=transform(wheat,Treat=factor(Treat),Block=factor(Block))
mod=aov(NC~Treat+Block,data=wheat)
res=HSD.test(mod, "Treat",main="NC with different treatment")
res
```

```
## $statistics
##       Mean       CV  MSerror      HSD
##   42.07167 6.378131 7.200561 6.164724
##
## $parameters
##   Df ntr StudentizedRange alpha  test name.t
##   15   6         4.594735  0.05 Tukey  Treat
##
## $means
##        NC      std r   Min   Max
## 1 38.2775 2.839324 4 34.98 41.22
## 2 44.0325 3.073352 4 40.89 46.69
## 3 46.7700 5.129659 4 42.07 52.68
## 4 40.6150 3.712811 4 37.18 45.85
## 5 39.5100 2.078012 4 37.61 41.99
## 6 43.2250 6.304525 4 34.89 50.15
##
## $comparison
## NULL
##
## $groups
##   trt   means  M
## 1   3 46.7700  a
## 2   2 44.0325 ab
## 3   6 43.2250 ab
## 4   4 40.6150 ab
## 5   5 39.5100  b
## 6   1 38.2775  b
```

Linear mixed model approaches to analyzing one-factor experimental designs

Linear mixed model approaches are much complicated and but they are very powerful and suitable for unbalaced and/or missing data. You need install and load the minque R package before you can use some important R functions.

Install and load minque

You can use the function of install.packages such as install.packages ("minque") and then the function require like require(minque).

```
require(minque)
```

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

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

```
## Loading required package: MASS
```

```
## Loading required package: Matrix
```

With the minque package you can run various kinds of data analyses

Case 1: Both treatment and block effects fixed

In this case we consider treatment effects as fixed.

```
require(coursedata)
data(wheatrcb)
wheat=wheatrcb
wheat=transform(wheat,Treat=factor(Treat),Block=factor(Block))

res1=lmm(NC~Treat+Block,data=wheat)[[1]]
res1$Var

## $NC
##           Est       SE Chi_sq    P_value
## V(e) 7.200561 2.629273    7.5 0.00308495

res1$FixedEffect

## $NC
##                 Est        SE    z_value      P_value
## mu       42.071667 0.6247409 67.3425872 0.000000e+00
## Treat(2)  1.960833 1.2360804  1.5863316 1.126641e-01
## Treat(5) -2.561667 1.2360804 -2.0724111 3.822712e-02
## Treat(4) -1.456667 1.2360804 -1.1784563 2.386148e-01
## Treat(1) -3.794167 1.2360804 -3.0695145 2.144070e-03
## Treat(6)  1.153333 1.2360804  0.9330569 3.507906e-01
## Treat(3)  4.698333 1.2360804  3.8009934 1.441171e-04
## Block(1) -4.071667 0.9811066 -4.1500758 3.323653e-05
## Block(2)  3.815000 0.9811066  3.8884664 1.008796e-04
## Block(3)  1.041667 0.9811066  1.0617263 2.883600e-01
## Block(4) -0.785000 0.9811066 -0.8001169 4.236430e-01

res1$RandomEffect ##no results shown

## NULL
```

Case 2: Random block effects

In this case we consider block effects as random.

```
res1=lmm(NC~Treat|Block,data=wheat)[[1]]
res1$Var

## $NC
##                Est       SE   Chi_sq    P_value
## V(Block) 9.744569 8.947018 1.186229 0.13804572
## V(e)     7.200561 2.629273 7.500000 0.00308495
```

```
res1$FixedEffect

## $NC
##                Est       SE    z_value      P_value
## mu       42.071667 1.662511 25.3060951 0.0000000000
## Treat(2)  1.960833 1.236080  1.5863316 0.1126641103
## Treat(5) -2.561667 1.236080 -2.0724111 0.0382271197
## Treat(4) -1.456667 1.236080 -1.1784563 0.2386147539
## Treat(1) -3.794167 1.236080 -3.0695145 0.0021440697
## Treat(6)  1.153333 1.236080  0.9330569 0.3507905836
## Treat(3)  4.698333 1.236080  3.8009934 0.0001441171

res1$RandomEffect

## $NC
##                 Pre       SE    z_value   P_value
## Block(1) -3.8419557 2.550892 -1.5061224 0.1320358
## Block(2)  3.5997693 2.550892  1.4111806 0.1581914
## Block(3)  0.9828990 2.550892  0.3853158 0.7000035
## Block(4) -0.7407127 2.550892 -0.2903740 0.7715301
```

Case 3: Random treatment and block effects model

In this case we consider both treatment effects and block effects as random.

```
res1=lmm(NC~1|Treat+Block,data=wheat)[[1]]
res1$Var

## $NC
##                Est       SE   Chi_sq    P_value
## V(Treat) 8.265679 6.400028 1.667990 0.09826401
## V(Block) 9.744569 8.947018 1.186229 0.13804572
## V(e)     7.200561 2.629273 7.500000 0.00308495

res1$FixedEffect

## $NC
##         Est       SE  z_value P_value
## mu 42.07167 2.028245 20.74289       0

res1$RandomEffect

## $NC
##                 Pre       SE    z_value    P_value
## Treat(2)  1.7768689 2.378282  0.7471229 0.45498939
## Treat(5) -2.3213323 2.378282 -0.9760543 0.32903756
## Treat(4) -1.3200029 2.378282 -0.5550237 0.57887844
## Treat(1) -3.4381998 2.378282 -1.4456653 0.14827110
## Treat(6)  1.0451282 2.378282  0.4394467 0.66033790
## Treat(3)  4.2575380 2.378282  1.7901737 0.07342598
## Block(1) -3.8419557 2.550892 -1.5061224 0.13203576
## Block(2)  3.5997693 2.550892  1.4111806 0.15819138
```

```
## Block(3)  0.9828990 2.550892  0.3853158 0.70000346
## Block(4) -0.7407127 2.550892 -0.2903740 0.77153015
```

Missing data case

A nice feature of linear model approaches is flexibity to deal with missing data. Now let us manually create a missing data set from the above wheat data.

```
n=nrow(wheat)
id=sample(n,2)
wheat1=wheat[-id,]
res1=lmm(NC~1|Treat+Block,data=wheat1)[[1]]
res1$Var

## $NC
##                Est       SE   Chi_sq     P_value
## V(Treat) 8.674201 6.724680 1.663856 0.098541772
## V(Block) 7.826003 7.400057 1.118433 0.145128078
## V(e)     6.511680 2.552020 6.510551 0.005361813

res1$FixedEffect

## $NC
##         Est       SE  z_value P_value
## mu 41.67471 1.921438 21.68933       0

res1$RandomEffect

## $NC
##                 Pre       SE    z_value    P_value
## Treat(2)  2.2042876 2.460433  0.8958941 0.37030934
## Treat(5) -2.0270947 2.460433 -0.8238771 0.41000938
## Treat(4) -0.9932246 2.460433 -0.4036787 0.68644897
## Treat(1) -3.1802575 2.460433 -1.2925599 0.19616330
## Treat(6) -0.7692829 2.296273 -0.3350136 0.73761481
## Treat(3)  4.7655722 2.460433  1.9368833 0.05275961
## Block(1) -3.4789198 2.262582 -1.5375886 0.12414924
## Block(2)  2.9676854 2.244452  1.3222318 0.18609096
## Block(3)  1.3595394 2.262582  0.6008797 0.54792011
## Block(4) -0.8483050 2.244452 -0.3779565 0.70546294
```

Then we can try one more time by repeating the above process.

```
n=nrow(wheat)
id=sample(n,2)
wheat1=wheat[-id,]
res1=lmm(NC~1|Treat+Block,data=wheat1)[[1]]
res1$Var

## $NC
##                Est       SE   Chi_sq     P_value
```

```
## V(Treat)  9.839131 7.505821 1.718372 0.094951390
## V(Block) 10.585735 9.728398 1.184021 0.138269462
## V(e)      6.982848 2.738113 6.503731 0.005382418

res1$FixedEffect

## $NC
##         Est      SE  z_value P_value
## mu 42.29713 2.14301 19.73725       0

res1$RandomEffect

## $NC
##                 Pre       SE    z_value    P_value
## Treat(2)  1.6403982 2.634150  0.6227428 0.53345356
## Treat(5) -2.1764320 2.572333 -0.8460927 0.39750101
## Treat(4) -1.5574578 2.634150 -0.5912562 0.55434877
## Treat(1) -3.7447259 2.634150 -1.4216066 0.15514047
## Treat(6)  0.8847965 2.634150  0.3358944 0.73695046
## Treat(3)  4.9534210 2.572333  1.9256533 0.05414768
## Block(1) -4.1087280 2.669230 -1.5392933 0.12373270
## Block(2)  3.7472254 2.645998  1.4161861 0.15672104
## Block(3)  0.8006381 2.669230  0.2999510 0.76421455
## Block(4) -0.4391356 2.645998 -0.1659622 0.86818671
```

You can compare the these results among different missing data cases and compare with the results from the complete data set.

This is some idea to lead jackknife process which are integrated into minque package.

```
res1=lmm.jack(NC~1|Treat+Block,data=wheat)[[1]]
res1$Var

## $NC
##          Estimate       SE       PValue   2.5%LL  97.5%UL
## V(Treat) 8.195099 1.289060 0.0005267433 3.641791 12.74841
## V(Block) 9.715641 1.527788 0.0005255852 4.319083 15.11220
## V(e)     7.296507 1.046848 0.0002614985 3.598758 10.99426

res1$PVar

## $NC
##              Estimate         SE       PValue    2.5%LL   97.5%UL
## V(Treat)/VP 0.3254295 0.05087590 5.030255e-04 0.1457221 0.5051369
## V(Block)/VP 0.3858190 0.05606868 2.886427e-04 0.1877693 0.5838687
## V(e)/VP     0.2887515 0.02908983 1.522258e-05 0.1859984 0.3915046

res1$FixedEffect

## $NC
##    Estimate        SE PValue  2.5%LL  97.5%UL
## mu 42.07862 0.2263466      0 41.2791 42.87813
```

```
res1$RandomEffect

## $NC
##                  Pre        SE       PValue     2.5%LL     97.5%UL
## Treat(2)  1.7714842 0.2699423 4.145948e-04  0.8179755  2.72499288
## Treat(5) -2.2382754 0.6034653 1.924481e-02 -4.3698771 -0.10667375
## Treat(4) -1.2911194 0.3468661 1.885902e-02 -2.5163436 -0.06589519
## Treat(1) -3.4257074 0.3512336 1.761293e-05 -4.6663589 -2.18505584
## Treat(6)  1.0349789 0.5963467 3.817324e-01 -1.0714778  3.14143560
## Treat(3)  4.1486390 0.4829286 4.990068e-05  2.4428053  5.85447277
## Block(1) -3.7963571 0.3561667 8.394372e-06 -5.0544334 -2.53828076
## Block(2)  3.5858482 0.2946617 2.733097e-06  2.5450238  4.62667257
## Block(3)  0.9768081 0.3405839 7.178237e-02 -0.2262257  2.17984200
## Block(4) -0.7662993 0.2060491 1.895259e-02 -1.4941200 -0.03847858
```

Please compare these results with the results without jackknife. The power with jackknife has been greatly improved, yet other results remain almost the same. This is the beauty of linear mixed model approaches integrated with jackknife methods.

You may also try a model with fixed treatment effects and random block effects.

```
res1=lmm.jack(NC~Treat|Block,data=wheat)[[1]]
res1$Var

## $NC
##          Estimate       SE       PValue   2.5%LL  97.5%UL
## V(Block) 9.732036 1.422840 0.0003023116 4.706182 14.75789
## V(e)     7.145094 1.108034 0.0004733303 3.231219 11.05897

res1$PVar

## $NC
##               Estimate         SE       PValue    2.5%LL   97.5%UL
## V(Block)/VP 0.5760467 0.05209507 6.162131e-06 0.3920329 0.7600605
## V(e)/VP     0.4239533 0.05209507 7.720345e-05 0.2399395 0.6079671

res1$FixedEffect

## $NC
##           Estimate        SE       PValue     2.5%LL     97.5%UL
## mu       42.106171 0.2888862 8.881784e-16 41.0857477 43.12659520
## Treat(2)  1.946825 0.3414294 1.173674e-03  0.7408051  3.15284576
## Treat(5) -2.577648 0.4941621 2.205821e-03 -4.3231614 -0.83213508
## Treat(4) -1.459995 0.4038456 2.223398e-02 -2.8864866 -0.03350406
## Treat(1) -3.797184 0.5131014 1.640453e-04 -5.6095961 -1.98477239
## Treat(6)  1.202675 0.8239168 5.263815e-01 -1.7076206  4.11297069
## Treat(3)  4.685327 0.7432622 5.613028e-04  2.0599257  7.31072903

res1$RandomEffect

## $NC
##                  Pre        SE       PValue     2.5%LL   97.5%UL
```

```
## Block(1) -3.7811056 0.3805671 1.510512e-05 -5.1253707 -2.436841
## Block(2)  3.5356619 0.3925568 3.394277e-05  2.1490460  4.922278
## Block(3)  1.0145633 0.5359516 3.107230e-01 -0.8785617  2.907688
## Block(4) -0.7691197 0.5270475 5.266079e-01 -2.6307930  1.092554
```

Conclusions

Please remember several key functions that can be used for one-factor ANOVA analysis:lm,aov,LSD.test,HSD.test, and duncan.test. Actually these functions can be used in more complicated experimental deisgn analysis.

You may also use minque package to analyze one-factor experimental designs. It can estimate all variance components and fixed effects and predict random effects.