Lab 9 - Experimental Design

April 17, 2017

Lab Goals

In this lab we will explore experimental design functions in R found in the *dae* package. In particular, we will determine how to:

1. design a completely randomized factorial experiment
2. estimate the sample size needed to achieve a desired level of power

The *dae* Package

The *dae* package in R provides randomization algorithms and will perform power and sample size calculations for several basic experimental designs. You will need to install this package on your personal computer, but it should already be installed in the labs in Mann library. After the package is installed, you can access it using the command `library(dae)`.

Part I - Designing a Completely Randomized Factorial Experiment

Suppose a researcher wants to design a factorial experiment consisting of a factor, `Poison` (3 levels: P1, P2, and P3), and a factor, `Medication` (4 levels: M1, M2, M3, and M4). For this factorial experiment, there are 3*4 = 12 treatment combinations replicated 4 times each. Thus one needs a total of 48 experimental units. Because the design structure is a completely randomized design, we need one single randomization in this experiment. Note that in this experiment there is no blocking.

Here is a table of the 12 treatment combinations in standard order.

Treatment	Poison Level	Medication Level
1	P1	M1
2	P1	M2
3	P1	M3
4	P1	M4
5	P2	M1
6	P2	M2

Treatment	Poison Level	Medication Level
7	P2	M3
8	P2	M4
9	P3	M1
10	P3	M2
11	P3	M3
12	P3	M4

The researcher first obtains a random sample of 48 subjects from the desired population. Subjects are referenced by their randomly assigned numbers.

Now, the researcher wants to randomly assign each of the 48 subjects one of the 12 treatment combinations where each treament combination is replicated 4 times. Using the *dae* package, this randomization can be done in 3 steps:

1. Create a list of the 48 subject numbers.

```
# n = number of observations
n=48

# Creates a list named "subject" denoting the number of subjects to be observed
subject.list = list(subject=n)
```

2. Generate a data frame of the 48 treatment combinations to be randomly assigned to the subjects using the function `fac.gen()`.

```
# Accessing the "dae" package
library(dae)
```

```
## Loading required package: ggplot2
```

```
# Creating a data frame of the 48 treatments before randomization
treatments = fac.gen(list(Poison=c("P1","P2","P3"), Medication = c("M1","M2","M3","M4")),times = 4)
```

For a completely randomized factorial design, `fac.gen()` contains 2 arguments.

i) a list of the factor levels for each factor

ii) the number of replicates (`times`) for each combination of factor levels

Here is the 48 treatment combinations in standard order created by `fac.gen()`.

```
treatments
```

```
##    Poison Medication
## 1      P1         M1
## 2      P1          M2
## 3      P1          M3
## 4      P1          M4
## 5      P2          M1
## 6      P2          M2
## 7      P2          M3
## 8      P2          M4
## 9      P3          M1
## 10     P3          M2
## 11     P3          M3
## 12     P3          M4
## 13     P1          M1
## 14     P1          M2
## 15     P1          M3
## 16     P1          M4
## 17     P2          M1
## 18     P2          M2
## 19     P2          M3
## 20     P2          M4
## 21     P3          M1
## 22     P3          M2
## 23     P3          M3
## 24     P3          M4
## 25     P1          M1
## 26     P1          M2
## 27     P1          M3
## 28     P1          M4
## 29     P2          M1
## 30     P2          M2
## 31     P2          M3
## 32     P2          M4
## 33     P3          M1
## 34     P3          M2
## 35     P3          M3
## 36     P3          M4
## 37     P1          M1
## 38     P1          M2
## 39     P1          M3
## 40     P1          M4
## 41     P2          M1
## 42     P2          M2
## 43     P2          M3
## 44     P2          M4
## 45     P3          M1
## 46     P3          M2
## 47     P3          M3
## 48     P3          M4
```

3. Randomly assign the treatments to the 48 subjects using the function `fac.layout()`.

```
# Randomly assigns the 48 treatments to the 48 subjects
treatment.random = fac.layout(unrandomized = subject.list, randomized = treatments)
```

For a completely randomized factorial design, `fac.layout()` requires 3 arguments.

i) the list of observation numbers (`subject.list` in this example)

ii) the data frame of factor level combinations to be randomly assigned (`treatments` in this example)

The following is the output from the `fac.layout()` function for this example. Our focus is on the last 3 columns of this output. These three columns include in order: the subject number (`subject`), the level of the first factor randonmly assigned to this subject (`Poison`), and the level of the second factor randomly assigned to this subject (`Medication`). These columns dictate how the reseacher will assign the treatments to the subjects.

The first two columns give details of the randomization. The permutation number indicates which subject was assigned the treatment associated with the `Units` number when the treatments are listed in the standard order.

```
treatment.random
```

```
##    .Units .Permutation subject Poison Medication
## 1       1           20       1     P3         M2
## 2       2           24       2     P1         M3
## 3       3            8       3     P3         M2
## 4       4            5       4     P2         M1
## 5       5            4       5     P1         M4
## 6       6           33       6     P3         M1
## 7       7           21       7     P2         M3
## 8       8           31       8     P1         M3
## 9       9            6       9     P1         M1
## 10     10           15      10     P1         M4
## 11     11           39      11     P3         M1
## 12     12           23      12     P3         M2
## 13     13           48      13     P2         M2
## 14     14           22      14     P1         M4
## 15     15           27      15     P3         M2
## 16     16           10      16     P2         M2
## 17     17           41      17     P2         M1
## 18     18           13      18     P2         M3
## 19     19           18      19     P2         M4
## 20     20           40      20     P1         M1
## 21     21           38      21     P2         M3
## 22     22            3      22     P1         M2
## 23     23           28      23     P3         M4
## 24     24           37      24     P1         M2
## 25     25           36      25     P1         M3
## 26     26           42      26     P3         M3
## 27     27            2      27     P1         M3
## 28     28           47      28     P3         M3
## 29     29           17      29     P3         M1
## 30     30           16      30     P1         M2
## 31     31           45      31     P2         M4
## 32     32           19      32     P2         M2
## 33     33           29      33     P2         M2
## 34     34            1      34     P2         M1
## 35     35           26      35     P3         M3
## 36     36           46      36     P1         M1
## 37     37            9      37     P3         M4
## 38     38           30      38     P3         M1
## 39     39           25      39     P3         M3
## 40     40           14      40     P2         M4
## 41     41           34      41     P2         M1
## 42     42              32       42     P1           M2
## 43     43               7       43     P2           M4
## 44     44              43       44     P3           M4
## 45     45              11       45     P2           M3
## 46     46              12       46     P3           M4
## 47     47              35       47     P1           M4
## 48     48              44       48     P1           M1
```

Part II - Sample Size and Power Plots

For the experiemental designs considered in this lab (and many others) the *dae* package includes both a function to calculate the power of a test to detect whether there is a difference between the treatment means (`power.exp()`) and also a function to calculate the sample size needed to obtain a specified power level for this test (`no.reps()`). Here we will explore these functions for

1. a *Single Factor Completely Randomized Design* where the factor has 6 levels (i.e. 6 treatments)
2. the *Completely Randomized Factorial Design* described in Part I

Power Calculations

The `power.exp()` function also found in the *dae* package will calculate the power of a F test designed to detect differences between treatment means. The basic structure of this function is:

`power.exp(rm=` , `df.num=` , `df.denom=` , `delta=` , `sigma=`, `alpha=)`

The following arguments need to be specified:

1. `rm` = number of replicates used to estimate the treatment means
2. `df.num` = degrees of freedom for the numerator of the F test
3. `df.denom` = degrees of freedom for the denominator of the F test
4. `delta` = the minimum difference between means to be detected
5. `sigma` = estimated population standard deviation shared by all treatment groups
6. `alpha` = significance level of the test

Single Factor Completely Randomized Design

Here we will assume we are using a single factor completely randomized design where the factor of interest has 6 levels (i.e. 6 treatments), and there are 8 observations per treatment. Based on a previous study, the population standard deviation is estimated to be 2. The minimum absolute difference between means that we would like to be able to detect is 5. We would like to determine the power of an F test to determine whether there is a difference between any of the treatment means at a significance level of 0.01.

The following table indicates how the first three arguments of the `power.exp()` function are determined for a *Single Factor Completely Randomized Design.*

`rm`	`df.num`	`df.denom`
r	t-1	t(r-1)

where

t = number of treatments

r = number of replicates per treatment

1. Fill in the arguments below for the `power.exp()` function for this study:

i) `rm` = 8

ii) `df.num` = 5

iii) `df.denom` = 6(8-1) = 42

iv) `delta` = 5

v) `sigma` = 2

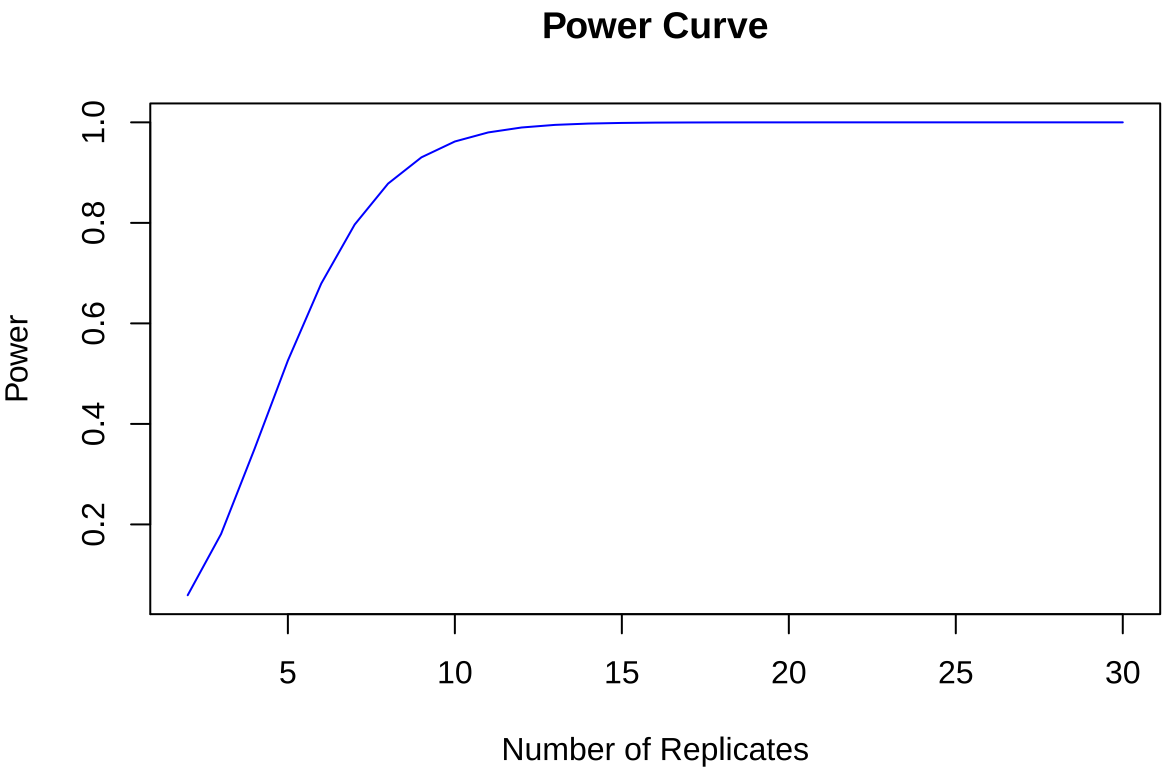
vi) `alpha` = 0.01

2. Calculate the power of the test in the code chunk below.

3. How does sample size affect the power calculation? Re-run the power calculation with the number of replicates increased to 10. What is the total sample size in this case?

4. Here we will calculate the power for a range of sample sizes and use these values to create a plot of power by sample size. Does the plot agree with the result from question 3?

```
# Vector of replicates (sample size per treatment) for which the power will be calculated
rep.range = 2:30

# Creating a vector of the power for each value of "rep.range"
power.range = as.vector(NA,mode='numeric')
num.treat=6

for ( i in 1:length(rep.range)) {
  power.range[i] = power.exp(rm=rep.range[i], df.num=num.treat-1, df.denom=num.treat*(rep.range[i]-1), d
}

# Creating the "Power Curve"
plot(rep.range,power.range,type = 'l',col=4,xlab = 'Number of Replicates',ylab = 'Power',main="Power Cu
```

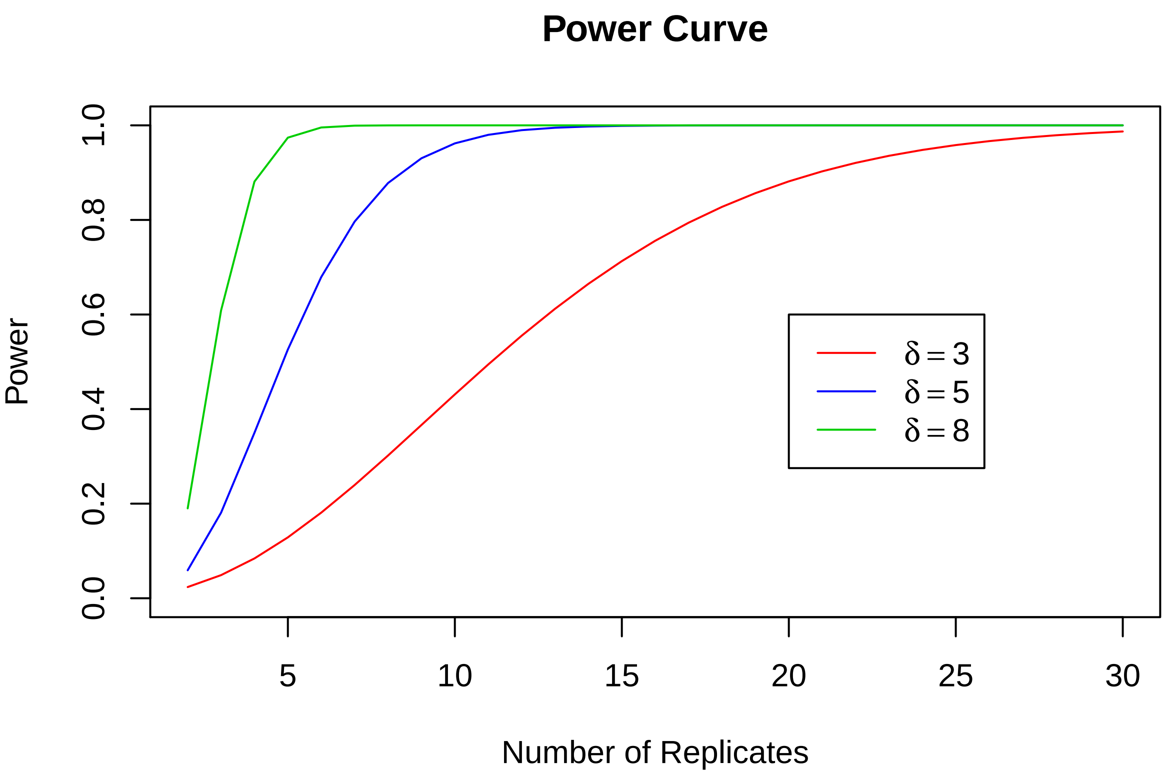


5. We can also look at how `delta` = *minimum difference between means to be detected* affects the power of the test for different sample sizes. First we will create two new arrays of power calculations where in the first set of calculations, `delta = 3`, and in the second set of calculations, `delta = 8`. How does `delta` affect the power calculation?

```
# delta = 3

power.range2 = as.vector(NA,mode='numeric')
num.treat=6

for ( i in 1:length(rep.range)) {
  power.range2[i] = power.exp(rm=rep.range[i], df.num=num.treat-1, df.denom=num.treat*(rep.range[i]-1),
}


# delta = 8

power.range3 = as.vector(NA,mode='numeric')
num.treat=6

for ( i in 1:length(rep.range)) {
  power.range3[i] = power.exp(rm=rep.range[i], df.num=num.treat-1, df.denom=num.treat*(rep.range[i]-1),
}
```

Plotting all three curves together.

```
plot(rep.range,power.range,type = 'l',col=4,xlab = 'Number of Replicates',ylab = 'Power',main="Power Cu
lines(rep.range,power.range2,type = 'l',col=2)
lines(rep.range,power.range3,type = 'l',col=3)

legend(20,.6,c(expression(delta==3),expression(delta == 5),expression(delta==8)),col=c(2,4,3),lty=c(1,1
```



Completely Randomized Factorial Design

Here we will again consider the completely randomized factorial design described in Part I. The two factors are `Poison` (3 levels) and `Medication` (4 levels). There are 4 replicates for each combination of `Poison` and `Medication` resulting in 48 observations in total. For the power calculations, we will assume the standard deviation of the observed responses is estimated to be 4. The following 3 questions may be of interest:

i) Are there differences between the mean responses of the 3 different levels of `Poison`?

ii) Are there differences between the mean responses of the 4 different levels of `Medication`?

iii) Does the mean response of `Poison` depend on the level of `Medication`?

For each of these questions, we can calculate the power of the F test to discern differences between the treatment means when they exist.

The following table indicates how to specify the first three arguments of the `power.exp()` function for testing for differences between the mean responses of different levels of two main effects (A & B) as well as the levels of the interaction between the two main effects (AB).

Factor	`rm`	`df.num`	`df.denom`
`A`	br	a-1	ab(r-1)
`B`	ar	b-1	ab(r-1)
`AB`	r	(a-1)(b-1)	ab(r-1)

where

r = number of replicate observations of each combination of the the levels of A and B

a = number of levels of A

b = number of levels of B

Note: `rm` is equal to the number of observations used to determine the sample treatment mean for each level of the given factor.

1. Determine the power of a test performed at the 0.05 significance level to detect a minimum difference of 3 between the effects of the different levels of `Poison`.
2. Determine the power of a test performed at the 0.05 significance level to detect a minimum difference of 3 between the effects of the different levels of `Medication`.
3. Determine the power of a test performed at the 0.05 significance level to detect a minimum difference of 3 between the effects of the different levels of the interaction between `Poison` and `Medication`.

Similar to the single factor completely randomized design, power curves looking at the relationships between sample size, delta and power can also be determined.

Sample Size Calculations

The `no.reps()` function, also found in the *dae* package, will calculate the sample size needed (of the replicates) to achieve a minimum power level for a F test designed to detect differences between treatment means. The basic structure of this function is:

`no.reps(multiple=`, `df.num=`, `df.denom=expression()`, `delta=` ,`sigma=`, `alpha=` , `power=`)

The following arguments need to be specified:

1. `multiple` = multiplier, m, which when multiplied by the number of replicates per treatment combination results in the total number of observations (rm) used to calculate the mean for each level of the factor being tested
2. `df.num` = degrees of freedom for the numerator of the F test
3. `df.denom` = degrees of freedom for the denominator of the F test as an `expression` in terms of `r`, the unknown number of replicates
4. `delta` = the minimum difference between means to be detected
5. `sigma` = estimated standard deviation of the response for all treatment groups
6. `alpha` = significance level of the test
7. `power` = desired minimum power of the test

Single Factor Completely Randomized Design

Here we will again assume we are using a single factor completely randomized design where the factor of interest has 6 levels (i.e. 6 treatments). Based on a previous study, the standard deviation of the responses is estimated to be 2. The minimum absolute difference between means that we would like to be able to detect is 5.

The following table indicates how the first three arguments of the `no.rep()` function are determined for a *Single Factor Completely Randomized Design.*

`multiple`	`df.num`	`df.denom`
1	t-1, t = number of treatments	expression((df.num + 1) * (r - 1))

1. Determine the replicate sample size necessary for an F test to determine whether any of the treatment means are different at a significance level of 0.01 to have a power of at least .80.
2. Note: This function gives the correct number of replications needed for the desired level of power and significance. However, by default the "power" calculated in the output uses a significance level of 0.05. Calculate the power of the test using the number of replicates determined in (1) and also calculate the power of the test where the number of replicates is one less than the number determined in (1). Verify the sample size determined in (1) is correct.

Completely Randomized Factorial Design

We can also perform sample size calculations to achieve a desired level of power for the tests related to a completely randomized factorial design.

The following table indicates how the first three arguments of the `power.exp()` function for testing for differences between the different levels of two main effects (A & B) as well as the interaction between the two main effects (AB).

Factor	`multiple`	`df.num`	`df.denom`
`A`	`b`	a-1	expression(multiple*(df.num + 1)* (r - 1))
`B`	`a`	b-1	expression(multiple*(df.num + 1)* (r - 1))
`AB`	`1`	(a-1)(b-1)	expression(multiple*(df.num + 1)* (r - 1))

Here we will determine the sample size per combination of factor levels A and B needed to achieve a minimum level of power to detect differences between the mean responses for the levels of `Poison`, `Medication`, and the interaction between `Poison` and `Medication`.

As a reminder, `Poison` has 3 levels and `Medication` has 4 levels. Assume, as before, the standard deviation of the observed responses is estimated to be 4.

1. Assume `delta=3` and that we would like to perform a test for the main effect of `Poison` at the 0.05 significance level. What is the replicate sample size necessary to achieve a minimum power of 0.75 for this test?
2. Using the result from (1), what is the total sample size needed for this experimental design?
3. Assume `delta=3` and that we would like to perform a test for the main effect of `Medication` at the 0.05 significance level. What is the replicate sample size necessary to achieve a minimum power of 0.75 for this test?
4. Assume `delta=3` and that we would like to perform a test for the interaction effect of `Poison` and `Medication` at the 0.05 significance level. What is the replicate sample size necessary to achieve a minimum power of 0.75 for this test?
5. Why is the required number of replicates significantly larger for the interaction terms in comparison to the main effects to achieve the same level of power?