A Basic Manual

How to conduct power analysis with simr to plan your study

Outline

1. **Packages and Functions**
2. **Power Analysis when a pilot study is available**
3. **Power Analysis when no pilot study**

Finch, 2019/1/31

1. Packages and Functions

1.1 Packages required

```
library(lme4)
library(lmerTest)
library(simr)
```

1.2 Functions frequently used

powerSim(): conduct power analysis with simulation

```
powerSim(fit, test = fixed(getDefaultXname(fit)), sim = fit,
        fitOpts = list(), testOpts = list(), simOpts = list(), seed, ...)
```

powerCurve(): conduct power analysis along a variable

```
powerCurve(fit, test = fixed(getDefaultXname(fit)), sim = fit,
        along = getDefaultXname(fit), within, breaks, seed, fitOpts = list(),testOpts
        = list(), simOpts = list(), ...)
```

extend (): extend your data along a variable. This function is used to add

subjects/items/observations in your pilot study

```
extend (object, along, within, n, values)
```

getData(): when extend() is applied to your data, check it

```
nrows (getData (object))
```

Modify functions: a series of functions to adjust parameters in lmm model

```
·    fixef (object) <- value      # adjust the fixed effect size
·    VarCorr (object) <- value # adjust the random effect
·    sigma (object) <- value      # adjust the residual
```

makeLmer(): create a linear mixed model according to your design

2. Power Analysis when a pilot study is available

A brief description of the work flow is as follows:

Step 1: Analyze your pilot data with linear mixed model
Step 2: Adjust the parameters in your model, like fixed effects, random intercepts and random sloples
Step 3: Extend your data to different numbers of subject/items/observations
Step 4: Run 1000 simulations on the extended data
Step 5: Repeat step2-4 until you get an appropriate design to obtain a 0.8 power

Now let's see the details:

Suppose we have conducted a pilot study with **6** subjects, to study the influences of word frequency and regularity on word recognition. The design is a 2*2 within-subject deign. Frequency (high, low) and regularity (regular, irregular) are the two independent variables. We have only 12 words (items) in each condition (48 in total).

```
> head(data,10)
   subject item frequency regularity  rt
1        1    1      high    regular 438
2        1    2      high    regular 488
3        1    3      high    regular 612
4        1    4      high    regular 697
5        1    5      high    regular 615
6        1    6      high    regular 405
7        1    7      high    regular 329
8        1    8      high    regular 732
9        1    9      high    regular 546
10       1   10      high    regular 641
```

```
> str(data)
'data.frame':   288 obs. of  5 variables:
 $ subject   : int  1 1 1 1 1 1 1 1 1 1 ...
 $ item      : int  1 2 3 4 5 6 7 8 9 10 ...
 $ frequency : Factor w/ 2 levels "high","low": 1 1 1 1
 $ regularity: Factor w/ 2 levels "irregular","regular":
 $ rt        : num  438 488 612 697 615 ...
```

Through the pilot study, we have the mean latencies and standard deviations for each condition.

```
> (mean<-aggregate(data$rt,by=list(data$frec
  Group.1   Group.2   x
1    high irregular 545
2     low irregular 549
3    high   regular 524
4     low   regular 526
> (sd<-aggregate(data$rt,by=list(data$freque
  Group.1   Group.2     x
1    high irregular 104.4
2     low irregular  94.8
3    high   regular  96.5
4     low   regular  94.4
```

Then we conduct a linear mixed model analysis on the pilot data

```
> d.fit<-lmer(rt~regularity*frequency+(1+regularity+frequency|subject)+
+ (1|item),data=data)
> anova(d.fit)
Type III Analysis of Variance Table with Satterthwaite's method
                     Sum Sq Mean Sq NumDF DenDF F value Pr(>F)
regularity            18875   18875     1   5.0    2.02   0.21
frequency               561     561     1  23.2    0.06   0.81
regularity:frequency    162     162     1  44.0    0.02   0.90
```

Check the parameters of fixed and random effect in the model, with summary ()

```
> summary(d.fit)
Linear mixed model fit by REML. t-tests use Satterthwaite's method ['lmerMo
Formula: rt ~ regularity * frequency + (1 + regularity + frequency | subjec
   Data: data

REML criterion at convergence: 3423

Scaled residuals:
    Min      1Q  Median      3Q     Max
-2.6000 -0.6593  0.0089  0.6227  2.9848

Random effects:
 Groups   Name              Variance Std.Dev. Corr
 item     (Intercept)         10.2     3.20
 subject  (Intercept)        445.9    21.12
          regularityregular  693.9    26.34   -1.00
          frequencylow        37.3     6.11   -1.00  1.00
 Residual                   9344.0    96.66
Number of obs: 288, groups:  item, 48; subject, 6

Fixed effects:
                              Estimate Std. Error     df t value Pr(>|t|)
(Intercept)                     545.06      14.32   7.15   38.07  1.6e-09
regularityregular               -20.80      19.41  10.73   -1.07     0.31
frequencylow                      4.37      16.35  36.34    0.27     0.79
regularityregular:frequencylow   -3.01      22.86  44.00   -0.13     0.90
```

Suppose we are interested in the main effect of regularity. Do we have enough confidence to detect it with 6 subjects and 48 items?

Note:

In simr, if there are interactions in the model, main effects cannot be tested

Remove the interaction

```
> d.fit<-lmer(rt~regularity+frequency+(1+regularity+frequency|subject)+
+ (1|item),data=data)
> anova(d.fit)
Type III Analysis of Variance Table with Satterthwaite's method
           Sum Sq Mean Sq NumDF DenDF F value Pr(>F)
regularity  18898   18898     1   5.2    2.03   0.21
frequency     564     564     1  39.0    0.06   0.81
```

Check the parameters again

```
> summary(d.fit)
Linear mixed model fit by REML. t-tests use Satterthwaite's metl
Formula: rt ~ regularity + frequency + (1 + regularity + freque:
   Data: data

REML criterion at convergence: 3432

Scaled residuals:
    Min      1Q  Median      3Q     Max
-2.6187 -0.6617  0.0028  0.6160  2.9965

Random effects:
 Groups   Name              Variance Std.Dev. Corr
 item     (Intercept)          0.0    0.00
 subject  (Intercept)        447.0   21.14
          regularityregular  695.5   26.37    -1.00
          frequencylow        37.4    6.12    -1.00  1.00
 Residual                   9320.9   96.54
Number of obs: 288, groups:  item, 48; subject, 6

Fixed effects:
                  Estimate Std. Error     df t value Pr(>|t|)
(Intercept)         545.82      13.10   5.38   41.67  5.8e-08 *:
regularityregular   -22.30      15.66   5.17   -1.42     0.21
frequencylow          2.87      11.65  38.99    0.25     0.81
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

And run 100 simulations to calculate the power

Note:
1000 simulations by default. The more simulations, the narrow confidence interval of the calculated power

```
> powerSim(d.fit,fixed('regularity'),nsim=100,alpha=0.05)
Power for predictor 'regularity', (95% confidence interval):
      26.00% (17.74, 35.73)

Test: Likelihood ratio

Based on 100 simulations, (18 warnings, 0 errors)
alpha = 0.05, nrow = 288

Time elapsed: 0 h 0 m 35 s

nb: result might be an observed power calculation
Warning message:
In observedPowerWarning(sim) :
  This appears to be an "observed power" calculation
```

Note:
In the powerSim (), we need to specify which fixed effect we want to test (fixed('regularity')), the number of simulations we want (nsim=100)

The power to detect regularity effect is pretty small.

Before moving on, we can see there is a warning message saying that this is an 'observed power'. We can ignore it since this is a pilot study.
Or, we might wonder the effect size of regularity is too large. According to previous studies, the effect size is typically -15ms.

We can adjust the effect size to 15ms with the following steps:

- Check the names of fixed effect

```
> fixef(d.fit)
      (Intercept) regularityregular      frequencylow
           545.82            -22.30              2.87
```

- Adjust the effect size of regularity

```
> fixef(d.fit)["regularityregular"]<--15

> fixef(d.fit)
      (Intercept) regularityregular      frequencylow
           545.82            -15.00              2.87
```

Note:
Other parameters can be adjusted as you need. For example, you can adjust the residual to 90 by: VarCorr (d.fit) ['Residual'] < -'90'.

Now run simulations again:

```
> fixef(d.fit)["regularityregular"]<--15
> powerSim(d.fit,fixed('regularity'),nsim=100,alpha=0.05)
Power for predictor 'regularity', (95% confidence interval):
      13.00% ( 7.11, 21.20)

Test: Likelihood ratio

Based on 100 simulations, (16 warnings, 0 errors)
alpha = 0.05, nrow = 288

Time elapsed: 0 h 0 m 36 s
```

Smaller power, which is a reasonable result.

How many subjects might be appropriate? We can try 24, which is a typical number of subjects in cognitive psychology

Adding subjects can be achieved with the extend() function.

```
> d2.fit<-extend(d.fit,along='subject',n=24)
> str(getData(d2.fit))
'data.frame':   1152 obs. of  5 variables:
 $ subject   : Factor w/ 24 levels "a","b","c","d",..: 1 1 1
 $ item      : Factor w/ 48 levels "1","2","3","4",..: 1 2 3
 $ frequency : Factor w/ 2 levels "high","low": 1 1 1 1 1 1 1
 $ regularity: Factor w/ 2 levels "irregular","regular": 2 2
 $ rt        : num  438 488 612 697 615 ...
```

We can see there are 24 subjects already. How about its power?

```
> powerSim(d2.fit,fixed('regularity'),nsim=100)
Power for predictor 'regularity', (95% confidence interval):
      58.00% (47.71, 67.80)

Test: Likelihood ratio

Based on 100 simulations, (69 warnings, 0 errors)
alpha = 0.05, nrow = 1152

Time elapsed: 0 h 1 m 32 s
```

Try 48 subjects:

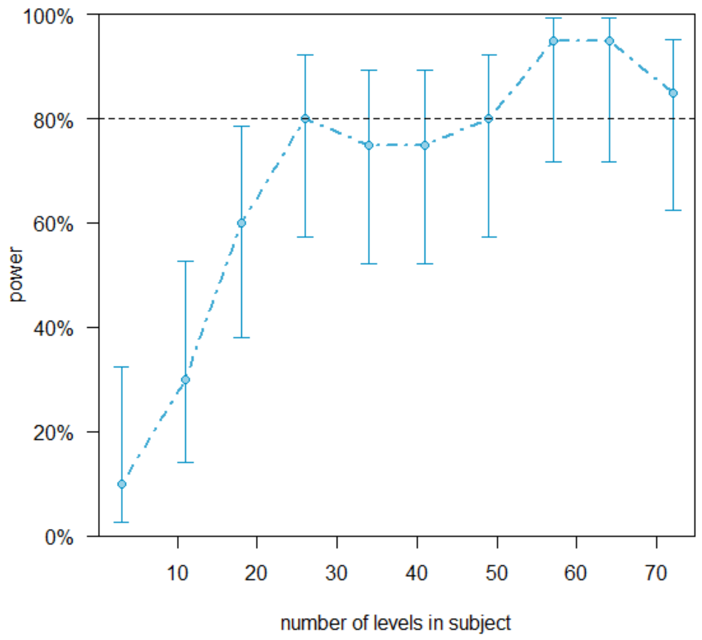
```
> d2.fit<-extend(d.fit,along='subject',n=48)
> str(getData(d2.fit))
'data.frame':   2304 obs. of  5 variables:
 $ subject   : Factor w/ 48 levels "a","b","c","d",..: 1 1 1
 $ item      : Factor w/ 48 levels "1","2","3","4",..: 1 2 3
 $ frequency : Factor w/ 2 levels "high","low": 1 1 1 1 1 1
 $ regularity: Factor w/ 2 levels "irregular","regular": 2 2
 $ rt        : num  438 488 612 697 615 ...
> powerSim(d2.fit,fixed('regularity'),nsim=100)
Power for predictor 'regularity', (95% confidence interval):
      78.00% (68.61, 85.67)

Test: Likelihood ratio

Based on 100 simulations, (76 warnings, 0 errors)
alpha = 0.05, nrow = 2304

Time elapsed: 0 h 2 m 45 s
```

Better, but not enough. Try 72 subjects:

```
> d2.fit<-extend(d.fit,along='subject',n=72)
> str(getData(d2.fit))
'data.frame':   3456 obs. of  5 variables:
 $ subject   : Factor w/ 72 levels "a","b","c","d",..: 1 1 1
 $ item      : Factor w/ 48 levels "1","2","3","4",..: 1 2 3
 $ frequency : Factor w/ 2 levels "high","low": 1 1 1 1 1 1 1
 $ regularity: Factor w/ 2 levels "irregular","regular": 2 2
 $ rt        : num  438 488 612 697 615 ...
> powerSim(d2.fit,fixed('regularity'),nsim=100)
Power for predictor 'regularity', (95% confidence interval):
      94.00% (87.40, 97.77)

Test: Likelihood ratio

Based on 100 simulations, (89 warnings, 0 errors)
alpha = 0.05, nrow = 3456

Time elapsed: 0 h 3 m 53 s
```

72 subjects lead to an overpowered study. We can draw a curve to see how power changes along numbers of subjects:

```
> d2.curve<-powerCurve(d2.fit,fixed('regularity'),along='subject',nsim=20)
Calculating power at 10 sample sizes along subject
> print(d2.curve)
Power for predictor 'regularity', (95% confidence interval),
by number of levels in subject:
      3: 10.00% ( 1.23, 31.70) - 144 rows
     11: 30.00% (11.89, 54.28) - 528 rows
     18: 60.00% (36.05, 80.88) - 864 rows
     26: 80.00% (56.34, 94.27) - 1248 rows
     34: 75.00% (50.90, 91.34) - 1632 rows
     41: 75.00% (50.90, 91.34) - 1968 rows
     49: 80.00% (56.34, 94.27) - 2352 rows
     57: 95.00% (75.13, 99.87) - 2736 rows
     64: 95.00% (75.13, 99.87) - 3072 rows
     72: 85.00% (62.11, 96.79) - 3456 rows
```



We need no less than 50 subjects to detect the 15ms regularity effect

If we only have 6 subjects, then more items may also improve the power of our study. The procedure is similar:

64 items instead of 48:

```
> d3.fit<-extend(d.fit,along='item',n=64)
> powerSim(d3.fit,fixed('regularity'),nsim=100)
Power for predictor 'regularity', (95% confidence interval):
      16.00% ( 9.43, 24.68)

Test: Likelihood ratio

Based on 100 simulations, (19 warnings, 0 errors)
alpha = 0.05, nrow = 384

Time elapsed: 0 h 0 m 40 s
```

96 items instead of 48:

```
> d3.fit<-extend(d.fit,along='item',n=96)
> powerSim(d3.fit,fixed('regularity'),nsim=100)
Power for predictor 'regularity', (95% confidence interval):
      18.00% (11.03, 26.95)

Test: Likelihood ratio

Based on 100 simulations, (31 warnings, 0 errors)
alpha = 0.05, nrow = 576

Time elapsed: 0 h 0 m 55 s
```

It seems that adding items contributes little to the power. Not a good strategy.

How about adding items and subjects simultaneously? For example, we can have 64 items and 48 subjects:

```
> d3.fit<-extend(d.fit,along='item',n=64)
> d4.fit<-extend(d3.fit,along='subject',n=48) #### extend d3.fit ####
> powerSim(d4.fit,fixed('regularity'),nsim=100)
Power for predictor 'regularity', (95% confidence interval):
      81.00% (71.93, 88.16)

Test: Likelihood ratio

Based on 100 simulations, (79 warnings, 0 errors)
alpha = 0.05, nrow = 3072

Time elapsed: 0 h 3 m 35 s
```

We can also define a range of subject numbers and a range of item numbers, and run the power analysis

```
sub_level<-seq (12,20,2) ###### subject number from 12 to 24 (step: 2)
item_level<-seq (48,56,4)
powers<-NULL

for (i in 1: length (sub_level)) {
    for (j in 1: length (item_level)) {
        d.extention<-extend(d.fit,along='item',n=item_level[j])
        d.extention<-extend(d.extention,along='subject',n=sub_level[i])
        simulation<-powerSim(d.extention,fixed('regularity'),nsim=50)
        powers<-rbind (powers, data.frame(subject=sub_level[i],item=item_level[j],
        power=simulation$x/simulation$n))
        print(paste(100*((i-1) * length(item_level) + j)/(length(sub_level) *
        length(item_level)),'% finished',sep=''))
```

```
> powers
   subject item power
1       12   48  0.30
2       12   52  0.26
3       12   56  0.34
4       14   48  0.46
5       14   52  0.44
6       14   56  0.44
7       16   48  0.40
8       16   52  0.52
9       16   56  0.30
10      18   48  0.38
11      18   52  0.46
12      18   56  0.46
13      20   48  0.50
14      20   52  0.38
15      20   56  0.60
```

What if we have no pilot data?

3. Power Analysis when no pilot study

When a pilot study is not available, a brief description of the work flow is as follows:

Step 1: Specify your design with a data frame, including subjects, items, conditions
Step 2: Specify the parameters required according to your design
Step 3: Create a linear mixed effect model with the data frame and parameters
Step 4: Run simulations to calculate the power
Step 5: Adjust numbers of subjects/items/observations until power is OK

Now let's see the details:

Suppose we want to conduct an experiment, with 10 subjects, 6 items. The only independent variable is condition (homogeneous vs. heterogeneous)

```
> subject <- 1:10
> condition <- c('homo','hetero')
> item<-letters[1:6]
>
> data <- expand.grid(subject=subject, item=item, condition=condition)
> data$condition<-as.factor(data$condition)
> data$subject<-as.factor(data$subject)
> data$item<-as.factor(data$item)
> str(data)
'data.frame':	120 obs. of  3 variables:
 $ subject  : Factor w/ 10 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8
 $ item     : Factor w/ 6 levels "a","b","c","d",..: 1 1 1 1 1 1 1 1 1
 $ condition: Factor w/ 2 levels "homo","hetero": 1 1 1 1 1 1 1 1 1 1
```

For this design, we will have fixed and random effects as follows:

- Fixed effect of condition, including its intercept and slope
- Random effects on subjects, including intercept, slope and their covariance
- Random effects on items, including intercept, slope and their covariance

These effects must be specified as follows:

```
> # fixed intercept and slope
> fixed <- c(2, -0.1)
> # random intercept and slope variance-covariance matrix
> randMat1 <- matrix(c(0.5,0.05,0.05,0.1), 2) # on subjects
> randMat2 <- matrix(c(0.5,0.05,0.05,0.1), 2) # on items
> # residual standard deviation
> res <- 1
```

Now the linear mixed effect model can be created:

```
> d.fit <- makeLmer(y ~ condition + (1+condition|subject)+(1+condition|item),
+ fixef=fixed, VarCorr=list(randMat1,randMat2),sigma=res,data=data)
```

Check the model

```
> print(d.fit)
Linear mixed model fit by REML ['lmerMod']
Formula: y ~ condition + (1 + condition | subject) + (1 + condition |      item)
   Data: data
REML criterion at convergence: 372
Random effects:
 Groups   Name             Std.Dev. Corr
 subject  (Intercept)      0.707
          conditionhetero  0.316    0.22
 item     (Intercept)      0.707
          conditionhetero  0.316    0.22
 Residual                  1.000
Number of obs: 120, groups:  subject, 10; item, 6
Fixed Effects:
    (Intercept)  conditionhetero
            2.0             -0.1
```

We can adjust the parameters with the Modify functions as before:

```
> fixef(d.fit)
    (Intercept) conditionhetero
            2.0            -0.1
> fixef(d.fit)["conditionhetero"]<--0.3
> fixef(d.fit)
    (Intercept) conditionhetero
            2.0            -0.3
```

Now let's conduct power analysis with this model:

```
> powerSim(d.fit, nsim=100)
Power for predictor 'condition', (95% confidence interval):
      17.00% (10.23, 25.82)

Test: Likelihood ratio

Based on 100 simulations, (2 warnings, 0 errors)
alpha = 0.05, nrow = 120

Time elapsed: 0 h 0 m 20 s
```

Note:
The fixed effect does not need to be specified in this simulation, as there is only one fixed effect.

How about experiments with 2 factors:

Suppose we have conducted a pilot study with **24** subjects. The design is a 2*2 within-subject deign. Frequency (high, low) and regularity (regular, irregular) are the two independent variables. We have only 12 items in each condition (48 in total).

```
> subject<-rep(rep(c(1:24),each=12),4)
> item<-c(rep(1:12,24),rep(13:24,24),rep(25:36,24),rep(37:48,24))
> frequency<-rep(c('high','low'),each=576)
> regularity<-rep(rep(c('regular','irregular'),each=288),2)
> data<-data.frame(subject,item,frequency,regularity)
> str(data)
'data.frame':   1152 obs. of  4 variables:
 $ subject   : int  1 1 1 1 1 1 1 1 1 1 ...
 $ item      : int  1 2 3 4 5 6 7 8 9 10 ...
 $ frequency : Factor w/ 2 levels "high","low": 1 1 1 1 1 1 1 1 1 1 .
 $ regularity: Factor w/ 2 levels "irregular","regular": 2 2 2 2 2 2
```

For this design, we will have fixed and random effects as follows:

- Fixed effects of frequency and regularity, including intercept and slopes
- Random effects on subjects, including intercept, slopes and their covariances
- Random effects on items, just its intercept

```
> # fixed intercept and slope
> fixed <- c(2, 15,10)
> # random intercept for item
> randMat1 <- 0.2
> # random intercepts and slopes for subject
> randMat2 <- matrix(c(27,0.2,0.2,0.2,36,0.2,0.2,0.2,36), 3)
> # standard deviation of residual
> res <- 1
```

Note:

The random effects must be a square matrix. Values on the main diagonal are the standard deviations of intercept, standard deviations of slopes of frequency and regularity. The other values are the correlations between intercept and slopes.

```
> randMat2
     [,1] [,2] [,3]
[1,] 27.0  0.2  0.2
[2,]  0.2 36.0  0.2
[3,]  0.2  0.2 16.0
```

We can change the effect size of regularity to 2 and run power analysis with 100 simulations

```
> fixef(d.fit)['regularityregular']<-2
> powerSim(d.fit, fixed('regularity'),nsim=100)
Power for predictor 'regularity', (95% confidence interval):
      68.00% (57.92, 76.98)

Test: Likelihood ratio

Based on 100 simulations, (0 warnings, 0 errors)
alpha = 0.05, nrow = 1152

Time elapsed: 0 h 5 m 49 s
```

The model we created can also be extended to different number of subjects/items. For example, we can try 36 subjects:

```
> d2.fit<-extend(d.fit,along='subject',n=36)
> str(getData(d2.fit))
'data.frame':   1728 obs. of  5 variables:
 $ subject    : Factor w/ 36 levels "a","b","c","d",..: 1 1 1
 $ item       : Factor w/ 48 levels "1","2","3","4",..: 1 2 3
 $ frequency  : Factor w/ 2 levels "high","low": 1 1 1 1 1 1
 $ regularity: Factor w/ 2 levels "irregular","regular": 2 2
 $ y          : num  17 17.1 19.1 18.1 18.6 ...
```

Run the simulation again:

```
> powerSim(d2.fit, fixed('regularity'),nsim=100)
Power for predictor 'regularity', (95% confidence interval):
      86.00% (77.63, 92.13)

Test: Likelihood ratio

Based on 100 simulations, (0 warnings, 0 errors)
alpha = 0.05, nrow = 1728

Time elapsed: 0 h 8 m 24 s
```

Now it’s OK.