Statistical Consulting Project

Michael Sampson

12/6/2018

Power Analysis of Simulated Treatment Effect of Exercise on Swallowing Performance using *simstudy*

> As we age, we lose muscles mass and function in the throat. This can cause difficulty swallowing, pneumonia (when material is misdirected into the lungs during swallowing), and can harm nutrition, hydration, and quality of life. Can we prevent and or reverse this phenomenon?” (Molfenter and Wolfe)

Study Background

(insert background info here)

The primary research question driving this project is,

- **Can we exercise the muscles in the throat to strengthen them for improved swallowing function?**

The goal of this project is to generate and test a set of candidate study designs that will be used to estimate the treatment effect of exercise on phrayngeal muscle loss. We’ll use simulation to conduct a power analysis over a range of possible treatment effects, intra-class correlations, and sample sizes to determine the final sample size for a sufficiently powered study. We plan to use the *simstudy* package in R to generate simulated data for each candidate study design. Once the data has been simulated, we’ll fit mixed effects models to estimate the treatement effect, running the simulation many times to estimate power.

Study population

Randomization will occur at the person level within sites. Sites are places such as senior centers, naturally occurring retirement communities (NORCs) and community centers. In this simulation, we will assume recruitment will occurs across 10 and that randomization will be *balanced* within sites.

Sample size

We plan to vary the total sample size by varying the number of recruited persons within each site, ranging from 15 to 30 by 3. The unit of inference is the individual. Results from this study will be used to make generalizations to the target population aging adults.

Interventions

There are two interventions that will be included in the study:

- `Exercise` : consists of 4 swallowing exercises 3 times weeks for 8 weeks
- `Exercise + Protein` : combined exercise regimen and supplemental protein for 8 weeks

As is standard in a randomized control trial there will be a group of participants who will be assigned to the `Control` group. Participants in the `Control` group will not receive an active intervention but will participate in the study in a yet to be determined fashion to that is as similiar to the internvention groups except for actually receiving the treatment.

Outcomes

In this simulation, the primary outcome is `peak pressure` , a measure of swallowing performance in adults. According to the researchers and the supporting literature in the field, a typical mean value for `peak pressure` in healthy, older adults is 364 (mmHg/s/cm), with a standard deviation of 97 (mmHg/s/cm). We'll use these values throughout our simulation.

There are a battery of throat-related secondary outcome measures that this simulation will not address but that may be included in future analyses. These secondary outcome include measures of pharyngeal shortening, pharyngeal constriction, pre-albumin levels, etc.

Effect Size

Currently there is no available estimate of the effect size of exercise on `peak pressure` . We therefore plan to simulate **low** (0.1), **medium** (0.5), and **high** (0.9) effect sizes when estimating power. The variation in the outcome measure is very high (sd = 97) which initially suggests that the effect size will need to be relatively large to be detected.

Intra-class correlation (ICC)

There is reason to believe that individuals recruited from the same site may have correlated outcomes, given the nature of the recruitment sites (e.g. naturally occuring senior centers). To address this concern, we will simulate different levels of the correlation among individuals at any given site by varying the intra-class corrleation (ICC) across simulations. We plan to simulate **low** (0.1), **medium** (0.5), and **high** (0.9) values of ICC.

To review, we plan to simulate a set of candidate designs to estimate the effect of exercise and exercise plus protein on swallowing performance in healthy aging adults. In our first simulation we will be generating data for a standard Randomized Control Trial (RCT) with two treatment arms. We'll fit a mixed effect model to estimate the effect and estimate the power by simulating the study repeatedly a large number of times. Finally, we plan to vary **sample size**, **effect size**, and **ICC** to understand how these parameters affect overall power.

Simulating study data with simstudy

We'll start by simulating data for one run of the study. We will then conduct a power analysis using the data definitions created for this single run. Finally, we'll run the study many times, varying the paramters outlined above.

Initially, we'll set our variance within and between sites using the variance estimates provided by the researches. We'll also set the ICC to 0.5.

```
varWithin <- sdOutcome^2
varBetween <- iccRE(ICC = 0.5, varWithin = varWithin , dist = 'normal')
varTotal <- varWithin + varBetween

cat(paste0("total variance = ",varTotal),"\n",
    paste0("between variance = ",varBetween),"\n",
    paste0("within variance = ",varWithin))
```

```
>R total variance = 18818
>R  between variance = 9409
>R  within variance = 9409
```

Next we'll define the data generating processes. We start by defining **site-level data** ("site"). The site level definitions include:

- random variation at the site level; (mean = 0, variance = between site variance)
- number of people per site

```
# how many participants per site?
n_per_site <- 15

# add variance of the outcome measure that is attributable to the cluster
siteDef <- defData(varname = "site_RE", formula = 0, variance = varBetween, dist = "normal", id = "site_id")
siteDef <- defData(siteDef, varname = "n_per_site", formula = n_per_site, dist = "nonrandom")

# head(siteDef)
```

Next we generate site level data using our site-level definitions. The code below will generate 10 with 15 people per site and a site-level variation of 9409.

```
# set a seed for reporducibility
set.seed(10031)

# create N sites with characteristics defined above and take a peek
dt.sites <- genData(n_sites, siteDef)
head(dt.sites)
```

```
>R    site_id    site_RE n_per_site
>R 1:       1  139.27769         15
>R 2:       2 -150.60252         15
>R 3:       3  -26.38527         15
>R 4:       4  -12.88243         15
>R 5:       5   13.16583         15
>R 6:       6  -71.40512         15
```

Now we add individuals to each site. This should be a dataset that contains 150 records.

```
# Add individuals to clusters
dt.person <- genCluster(dt.sites, cLevelVar = "site_id", numIndsVar = "n_per_site", l
evel1ID = "person_id")
head(dt.person)
```

```
>R    site_id  site_RE n_per_site person_id
>R 1:       1 139.2777         15         1
>R 2:       1 139.2777         15         2
>R 3:       1 139.2777         15         3
>R 4:       1 139.2777         15         4
>R 5:       1 139.2777         15         5
>R 6:       1 139.2777         15         6
```

```
# check number of records
nrow(dt.person)
```

```
>R [1] 150
```

Because we are randomizing the treatment arms within sites, we now need to assign each person to a treatment group that is balanced **within** each site. This means that each site should have 5 people assigned to each treatment.

```
# Assign intervention randomly to each person, balanced within each site
dt.person <- trtAssign(dt.person, nTrt = 3, grpName = "treatment",  balanced = T, str
ata = "site_id")
head(dt.person)
```

```
>R    person_id treatment site_id  site_RE n_per_site
>R 1:         1         3       1 139.2777         15
>R 2:         2         1       1 139.2777         15
>R 3:         3         3       1 139.2777         15
>R 4:         4         2       1 139.2777         15
>R 5:         5         1       1 139.2777         15
>R 6:         6         3       1 139.2777         15
```

Examine balance across treatment groups: Are there an equal number of sites assigned to each treatment?

```
>R    treatment sites
>R 1:         1    10
>R 2:         2    10
>R 3:         3    10
```

Examine balance of treatment arms within sites: Are there an equal number of people assigned to each treatment within each site?

```
>R     site_id treatment N
>R  1:       1         1 5
>R  2:       1         2 5
>R  3:       1         3 5
>R  4:       2         1 5
>R  5:       2         2 5
>R  6:       2         3 5
>R  7:       3         1 5
>R  8:       3         2 5
>R  9:       3         3 5
>R 10:       4         1 5
>R 11:       4         2 5
>R 12:       4         3 5
>R 13:       5         1 5
>R 14:       5         2 5
>R 15:       5         3 5
>R 16:       6         1 5
>R 17:       6         2 5
>R 18:       6         3 5
>R 19:       7         1 5
>R 20:       7         2 5
>R 21:       7         3 5
>R 22:       8         1 5
>R 23:       8         2 5
>R 24:       8         3 5
>R 25:       9         1 5
>R 26:       9         2 5
>R 27:       9         3 5
>R 28:      10         1 5
>R 29:      10         2 5
>R 30:      10         3 5
>R     site_id treatment N
```

Now we generate outcomes for each person prior to receiving their assigned treatment (`Y_pre`) and after (`Y_post`). The treatment takes on values 1 (control), 2 (exercise), and 3 (exercise+protein). For this initial set up we are going to assume that **exercise has an effect of 20 (mmHg/s/cm)** and **exercise+ has an effect of 24 (mmHg/s/cm)**.

```
trtDef <- defDataAdd(varname = "Y_pre",
                     dist = "normal",
                     formula = "site_RE",
                     variance = varWithin)

trtDef <- defDataAdd(trtDef,
                     varname = "Y_post",
                     dist = "normal",
                     formula = "site_RE  + 20 * (treatment == 2) + 24 * (treatment ==
3)",
                     variance = varWithin)

# head(trtDef)
```

With our treatment definitions in place, we generate pre- and post-treatment outcome values for each individual in our dataset

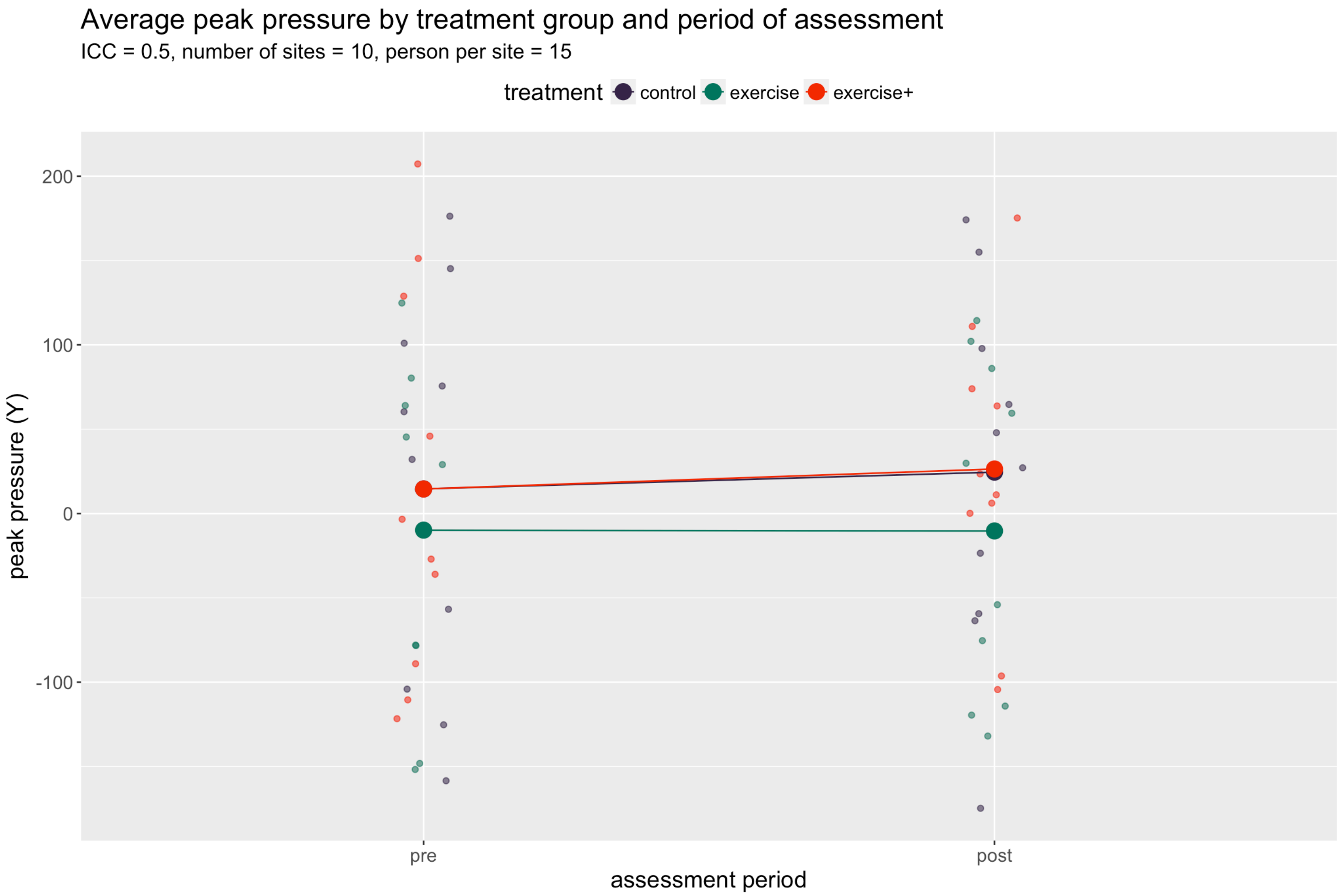
```
# Generate outcome measures for each person pre treatment and post treatment
dt.person <- addColumns(trtDef, dt.person)
round(head(dt.person))
```

```
>R    person_id treatment site_id site_RE n_per_site Y_pre Y_post
>R 1:         1         3       1     139         15   377    167
>R 2:         2         1       1     139         15    41    144
>R 3:         3         3       1     139         15   254    -48
>R 4:         4         2       1     139         15    12     74
>R 5:         5         1       1     139         15    53    379
>R 6:         6         3       1     139         15   149    218
```

For each site, calculate mean of outcome at both time points across all three treatment conditions

```
>R     treatment site_id   Y_bar_pre   Y_bar_post
>R  1:         1       1  145.185548  174.1258704
>R  2:         1       2 -158.438234 -174.7553384
>R  3:         1       3   32.081537  -23.5382824
>R  4:         1       4  -56.757735   27.1302526
>R  5:         1       5   60.351163   47.9219762
>R  6:         1       6 -125.260798  -63.5605052
>R  7:         1       7  100.974564   64.6670959
>R  8:         1       8  176.274921  154.9540330
>R  9:         1       9 -104.109881  -59.3734158
>R 10:         1      10   75.602149   97.8447202
>R 11:         2       1   29.021792  114.3849669
>R 12:         2       2 -148.166556 -114.1635918
>R 13:         2       3  -77.986348 -131.9365912
>R 14:         2       4  -78.320593  -54.0507956
>R 15:         2       5   45.360307   29.8001377
>R 16:         2       6   14.202174 -119.5190747
>R 17:         2       7   80.304842   59.4733071
>R 18:         2       8  124.790655  102.0756929
>R 19:         2       9 -151.731854  -75.3650705
>R 20:         2      10   64.026717   85.9900918
>R 21:         3       1  207.234167  110.9306976
>R 22:         3       2 -121.640131 -104.3595997
>R 23:         3       3  -27.015321    0.1166144
>R 24:         3       4   -3.403255    6.1557313
>R 25:         3       5   45.877759   23.4928443
>R 26:         3       6 -110.478117   11.1125237
>R 27:         3       7  -36.017544   63.7574877
>R 28:         3       8  151.217858  175.2084798
>R 29:         3       9  -89.059840  -96.2849350
>R 30:         3      10  128.865412   73.9487618
>R     treatment site_id   Y_bar_pre   Y_bar_post
```

Now plot mean value of `peak pressure` for each treatment condition along with individual outcomes. (This plot is a replication of that found on rdatagen.com)



Now we compute the pre and post difference for all three treatment groups. Remember – the true treatment effects are:

- control = 0
- exercise = 20
- exercise+ = 24

```
>R    treatment Y_pre Y_post Y_post - Y_pre
>R 1:   control    15     25             10
>R 2:  exercise   -10    -10              0
>R 3: exercise+    15     26             12
```

We'll calculate the treatment effect using regression. We'll start with a simple linear model that does not take into account cluster level variation. The first model ignores the first assessstment (pre), the second includes pre- scores as a term in the linear model.

```
# run a simple linear model, ignoring pre-scores
tidy(lm(Y_post ~ treatment, data = dt.person))[c("term","estimate","p.value")]
```

```
>R # A tibble: 3 x 3
>R   term               estimate p.value
>R   <chr>                 <dbl>   <dbl>
>R 1 (Intercept)           24.5    0.189
>R 2 treatmentexercise    -34.9    0.187
>R 3 treatmentexercise+     1.87   0.944
```

```
# run a simple linear model, including pre-scores
tidy(lm(Y_post ~ treatment + Y_pre, data = dt.person))[c("term","estimate","p.value")
]
```

```
>R # A tibble: 4 x 3
>R   term               estimate       p.value
>R   <chr>                 <dbl>         <dbl>
>R 1 (Intercept)          18.3   0.273
>R 2 treatmentexercise   -24.5   0.301
>R 3 treatmentexercise+    1.88  0.936
>R 4 Y_pre                 0.424 0.00000000898
```

Now run a mixed effects model to account for site-level variation. Remember:

- site-level variance = 9409
- person-level variance = 9409

```
# run a mixed effects model, ignoring pre-scores
summary(lmerTest::lmer(Y_post ~ treatment + (1|site_id), data = dt.person))
```

```
>R Linear mixed model fit by REML. t-tests use Satterthwaite's method [
>R lmerModLmerTest]
>R Formula: Y_post ~ treatment + (1 | site_id)
>R    Data: dt.person
>R
>R REML criterion at convergence: 1804.5
>R
>R Scaled residuals:
>R      Min       1Q   Median       3Q      Max
>R -2.32120 -0.63879  0.02667  0.69237  2.45540
>R
>R Random effects:
>R  Groups   Name        Variance Std.Dev.
>R  site_id  (Intercept) 8032     89.62
>R  Residual             9900     99.50
>R Number of obs: 150, groups:  site_id, 10
>R
>R Fixed effects:
>R                    Estimate Std. Error      df t value Pr(>|t|)
>R (Intercept)          24.542     31.642  11.923   0.776   0.4531
>R treatmentexercise   -34.873     19.900 138.000  -1.752   0.0819 .
>R treatmentexercise+    1.866     19.900 138.000   0.094   0.9254
>R ---
>R Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>R
>R Correlation of Fixed Effects:
>R             (Intr) trtmnt
>R tretmntxrcs -0.314
>R trtmntxrcs+ -0.314  0.500
```

```
# run a mixed effects model, including pre-scores
summary(lmerTest::lmer(Y_post ~ treatment + Y_pre + (1|site_id), data = dt.person))
```

```
>R Linear mixed model fit by REML. t-tests use Satterthwaite's method [
>R lmerModLmerTest]
>R Formula: Y_post ~ treatment + Y_pre + (1 | site_id)
>R    Data: dt.person
>R 
>R REML criterion at convergence: 1807.5
>R 
>R Scaled residuals: 
>R      Min       1Q   Median       3Q      Max 
>R -2.29204 -0.64756  0.02788  0.70042  2.48127 
>R 
>R Random effects:
>R  Groups   Name        Variance Std.Dev.
>R  site_id  (Intercept)  7327     85.6   
>R  Residual             10009    100.0   
>R Number of obs: 150, groups:  site_id, 10
>R 
>R Fixed effects:
>R                      Estimate Std. Error        df t value Pr(>|t|)  
>R (Intercept)          23.92771   30.56528  10.56667   0.783   0.4509  
>R treatmentexercise  -33.84433   20.10332 135.91906  -1.684   0.0946 .
>R treatmentexercise+   1.86758   20.00920 135.70458   0.093   0.9258  
>R Y_pre                0.04208    0.07950 145.53787   0.529   0.5974  
>R ---
>R Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>R 
>R Correlation of Fixed Effects:
>R             (Intr) trtmnt trtmn+
>R tretmntxrcs -0.329              
>R trtmntxrcs+ -0.327  0.498       
>R Y_pre       -0.038  0.097  0.000
```

Power Analysis

```
iter <- 100
p.results <- data.table()

print("...running power analysis...")
```

```
>R [1] "...running power analysis..."
```

```
tic("single power analysis")
for (i in 1:iter) {

  dt.sites <- genData(n_sites, siteDef)

  dt.person <- genCluster(dt.sites, cLevelVar = "site_id", numIndsVar = "n_per_site",
level1ID = "person_id")

  dt.person <- trtAssign(dt.person, nTrt = 3, grpName = "treatment",  balanced = T, s
trata = "site_id")

  dt.person <- addColumns(trtDef, dt.person)

  dt.person[, treatment := factor(treatment, levels = c(1,2,3), labels = c("control",
"exercise","exercise+"))]

  # store model fit
  mod.fit <- lmerTest::lmer(Y_post ~ treatment + Y_pre + (1|site_id), data = dt.perso
n)

  # extract p-values for each intervention
  p.exercise <- coef(summary(mod.fit))["treatmentexercise","Pr(>|t|)"]
  p.exercise_protein <- coef(summary(mod.fit))["treatmentexercise+","Pr(>|t|)"]

  # store p-values into a data.table
  p.results <- rbind(p.results, data.table(p.exercise, p.exercise_protein))
}
toc()
```

```
>R single power analysis: 10.389 sec elapsed
```

```
# glance at results different iterations
head(p.results)
```

```
>R      p.exercise p.exercise_protein
>R 1: 0.5017580119       0.8204347589
>R 2: 0.0029876395       0.0007978566
>R 3: 0.0014428871       0.1552837281
>R 4: 0.0006841345       0.0558721775
>R 5: 0.4916302197       0.5372613187
>R 6: 0.2591182175       0.0659470600
```

Now find the proportion of iterations where the p-value for each treatment is less that a specificed alpha value.

```
>R    p.exercise p.exercise_protein
>R 1:       0.22               0.23
```

Power Anlayis for varying sample sizes, effect sizes and ICCs

Calculate power analysis for different sample sizes (150 to 300 by), effect sizes (0.1, 0.5, 0.9), and ICCs (0.1, 0.5, 0.9). Start by creating function to run power for N iterations with given arguments.

```r
get_power <- function(n_per_site, eff_size, icc, protein_eff = 0.05, iters = 100, n_sites = 10, block = T, alpha = 0.05) {

  # compute the absolute effect based on the effect size passed into function
  # for both treatment arms
  eff_exercise <- round(eff_size*sdOutcome)
  eff_exercise_protein <- round((eff_size+protein_eff)*sdOutcome)

  # create a text based formula to be based to simstudy functions a mean value
  # for treatment arms
  f.pre <- "site_RE"
  f.post <- paste("site_RE +", eff_exercise, "* (treatment == 2) +", eff_exercise_protein, "* (treatment == 3)")

  # initialize a data table to store different variables
  power.results <- data.table(ss = n_per_site * n_sites,
                              eff_size = eff_size,
                              icc = icc)

  p.results <- data.table()

  for (i in 1:iters) {
    varWithin <- sdOutcome^2
    varBetween <- iccRE(ICC = icc, varWithin = varWithin , dist = 'normal')
    # varTotal <- varWithin + varBetween

    # add variance of the outcome measure that is attributable to the cluster
    siteDef <- defData(varname = "site_RE", formula = meanOutcome, variance = varBetween, dist = "normal", id = "site_id")
    siteDef <- defData(siteDef, varname = "n_per_site", formula = n_per_site, dist = "nonrandom")

    # Generate site data
    dt.sites <- genData(n_sites, siteDef)

    # Add individuals to each site
    dt.person <- genCluster(dt.sites, cLevelVar = "site_id", numIndsVar = "n_per_site", level1ID = "person_id")

    # Randomly assign intervention at person level
    if(block == T){
      dt.person <- trtAssign(dt.person, nTrt = 3, grpName = "treatment", balanced = TRUE, strata = "site_id")
    } else {
      dt.person <- trtAssign(dt.person, nTrt = 3, grpName = "treatment", balanced = T
RUE)
    }

    # Generate outcome measures for each person pre- and post- treatment
    # control (treatmment == 1), exercise (treatment == 2), exercise+ (treatment == 3
)
    trtDef <- defDataAdd(varname = "Y_pre",
                         dist = "normal",
                         formula = f.pre,
                         variance = varWithin)

    trtDef <- defDataAdd(trtDef,
                         varname = "Y_post",
                         dist = "normal",
                         formula = f.post,
                         variance = varWithin)

    # Generate outcome measures for each person pre treatment and post treatment
    dt.person <- addColumns(trtDef, dt.person)
    dt.person[, treatment := factor(treatment, levels = c(1,2,3), labels = c("control
","exercise","exercise+"))]

    # store model fit
    mod.fit <- lmerTest::lmer(Y_post ~ treatment + Y_pre + (1|site_id), data = dt.per
son)

    # extract p-values for each intervention
    p.exercise <- coef(summary(mod.fit))["treatmentexercise","Pr(>|t|)"]
    p.exercise_protein <- coef(summary(mod.fit))["treatmentexercise+","Pr(>|t|)"]

    # store p-values into a data.table
    p.results <- rbind(p.results, data.table(p.exercise, p.exercise_protein))
  }
  power.results <- cbind(power.results, p.results[, lapply(.SD, function(x) {mean(x <
alpha)})])
  return(power.results)
}
```

Determine parameters to go into function. First determine vector of **sample sizes**

```
n.vec <- seq(15, 30, 1)[seq(15, 30, 1) %% 3 == 0]
print(n.vec)
```

```
>R [1] 15 18 21 24 27 30
```

Next determine vector of **effect sizes**

```
eff.vec <- seq(0.1, 0.9, by = 0.1)
print(eff.vec)
```

```
>R [1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
```

Next determine vector of **ICCs**

```
icc.vec <- seq(0.1, 0.9, by = 0.1)
print(icc.vec)
```

```
>R [1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
```

Now generate a data.table with the different combinations of parameter values of sample size and treatment effects

```
>R    n_per_site eff_sizes icc
>R 1:         15       0.1 0.1
>R 2:         18       0.1 0.1
>R 3:         21       0.1 0.1
>R 4:         24       0.1 0.1
>R 5:         27       0.1 0.1
>R 6:         30       0.1 0.1
```

```
>R [1] 486
```

Now run power analysis for randomized control trial for different sample sizes, effect sizes, and ICCs. Keep track of time using *tictoc* package.

```
iters = 250

print("...Running power analysis for study design with different sample sizes, effect
sizes, and ICCs...")
```

```
>R [1] "...Running power analysis for study design with different sample sizes, effec
t sizes, and ICCs..."
```

```
tic(msg = "run power analysis",log = T)
power_run <- mapply(get_power, n_per_site = param.cols$n_per_site, eff_size = param.c
ols$eff_size, param.cols$icc, iters = iters, n_sites = n_sites, block = T)
toc()
```

```
>R run power analysis: 16948.24 sec elapsed
```

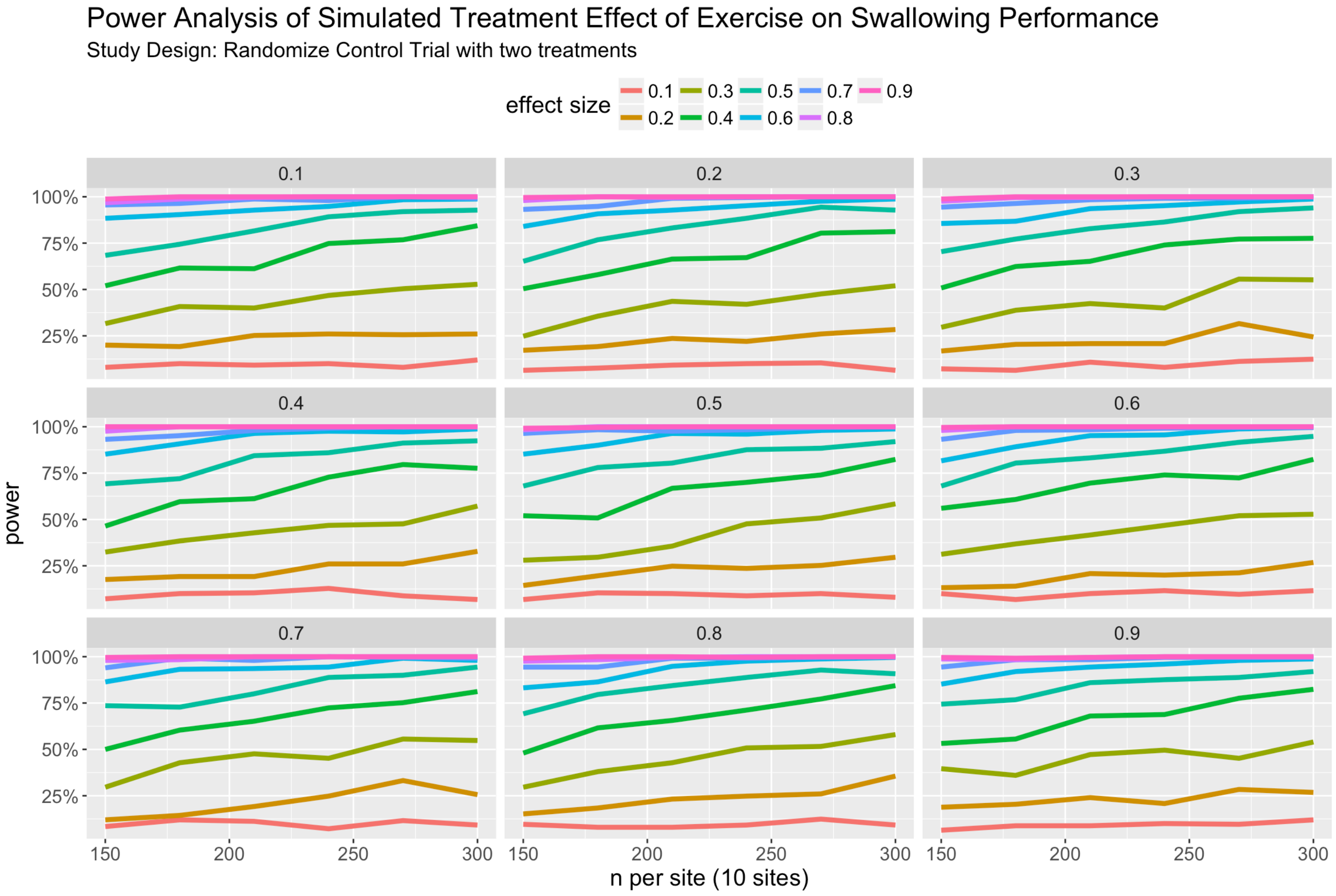
Now convert results from power analyis into a data table

```
>R [1] "...Finished power analysis. Now reshaping and converting to data.table..."
```

```
>R [1] "...Finished cleaning and reshaping results."
```

```
>R convert to data.table: 0.023 sec elapsed
```

Plot results of power analysis. We'll plot the power of each study (Y) against the sample size (x) for different levels of the effect size and ICC



Power Analysis of Simulated Treatment Effect of Exercise on Swallowing Performance

Study Design: Randomize Control Trial with two treatments