SimulPower: Simultaneous Power Analysis for a Set of Statistical Tests

Joel M Le Forestier¹

¹ Department of Psychology, University of Toronto

Abstract

This document includes an introduction to, and set of vignettes for, the SimulPower R package. SimulPower is a package for simulating power to detect multiple effects simultaneously (“simultaneous power”). For a detailed discussion of simultaneous power, see Le Forestier, Page-Gould, & Chasteen (Forthcoming). The present version of this document is up-to-date as of July 2021 and SimulPower Version 0.8.0.

Keywords: Statistical power, power analysis, probability, sample size, simulation

SimulPower: Simultaneous Power Analysis for a Set of Statistical Tests

SimulPower

SimulPower is an R package for simulating power to detect multiple effects simultaneously (“simultaneous power”). For a detailed discussion of simultaneous power, see Le Forestier, Page-Gould, & Chasteen (Forthcoming). The present document contains instructions for the use of SimulPower, beginning with installation, and including vignettes and commentary for each of its functions.

Installing SimulPower

Before beginning, you will have to install the SimulPower package. SimulPower is hosted on GitHub. To install packages from GitHub, you must first install the devtools package.

```
install.packages("devtools")  
devtools::install_github("joelleforestier/SimulPower")
```

Load SimulPower into your R environment

Once installed, packages from GitHub are loaded into your environment the same was as any other package.

```
library(SimulPower)
```

Once loaded, you are ready to begin using SimulPower. To help you do so, the following sections of this document will present vignettes and commentary for each of SimulPower’s main functions, beginning with those that perform the simultaneous power analyses.

pwrMultivars

pwrMultivars is one of main power analysis functions in SimulPower. It simulates simultaneous power for a set of tests contained in a single linear model with between two

and ten predictor variables. This function simulates data based on a correlation matrix imposed using the `mvrnorm` function from the MASS package (Venables & Ripley, 2002). To speed up processing time, `pwrMultivars` uses parallel processing using the `parallel`, `doParallel`, and `foreach` packages (Microsoft Corporation & Weston, 2019; Microsoft Corporation & Weston, 2020; R Core Time, 2020).

`pwrMultivars` estimates power by running the same multiple regression in random samples drawn from a population with a user-specified covariance matrix. By default, `pwrMultivars` applies a default alpha-level of 0.05, assumes that you aim to detect significant effects in all of your tests, uses a seed of 1, specifies no covariances between the predictor variables, and bases the power analysis off of 5,000 iterations. All of these parameters are customizable.

You must provide `pwrMultivars` with the desired sample size for which power is to be calculated, the units in which you are providing effect sizes, and the effect sizes for the relationships between each of the predictor variables and the dependent variable. All other arguments are optional.

Arguments taken by `pwrMultivars`:

n. Set the size of each sample to be drawn from the population. This is the sample size for which you are estimating statistical power. In other words, setting `n` to equal 100 will estimate statistical power at $n = 100$. Accepts any positive number. This argument has no default.

es_units. Set the units in which you are specifying your effect sizes. Accepts “d” for Cohen’s d, “r” for correlation coefficients, and “r2” for percent of variance accounted for. This argument has no default.

null_effect. For which, if any, of your predictors are you computing “null power?” If you want to compute “power” to NOT detect an effect, use this argument to specify which

effects are predicted nulls by setting this argument equal to the number(s) corresponding to the predictors you hypothesize to be null. If you predict predictor 3 and predictor 4 to have null effects, you should specify `null_effect = c(3, 4)`. Accepts either a single whole number between 1 and the number of predictors you have specified or a vector of numbers between 1 and the the number of predictors you have specified. Default = no null effects.

iterations. How many times you would like to run your model in random samples drawn from your population? One model will be run in each random sample. Accepts any whole number greater than 0. Default = 5000.

alpha. Set your alpha level. This is the threshold below which p-values will be considered significant. Accepts any number greater than 0 and less than 1. Default = 0.05.

bonferroni. Apply a bonferroni correction? This is suggested if you intend on interpreting the results of multiple tests individually, but not if you intend on assessing a single research question by triangulating across multiple tests (Le Forestier, Page-Gould, & Chasteen, Forthcoming). Accepts TRUE or FALSE. Default = FALSE.

seed. Set a seed to make your results reproducible. Accepts any number. Default = 1.

es1...es10. The effect size, expressed in units specified in the `es_units` argument, for the relationship between each predictor and the dependent variable. You should always specify these in order, beginning with `es1`, and not skipping any. Accepts any number. These arguments have no defaults.

iv1iv2_cov...iv9iv10_cov. The relationships between each set of predictors, specified in correlation coefficients. Specifying relationships between predictors is optional unless your predictors, together, account for more than 100% of the variance in your DV, in which case you must specify relationships between your predictors to make that possible. Accepts any number between -1 and 1. Default = 0.

print_result. Should power analysis results be printed to the console? Accepts TRUE or FALSE. Default = TRUE.

The above information is also summarized in `pwrMultivars`'s help page.

`?pwrMultivars`

`pwrMultivars` vignettes

A simple vignette. Imagine that you aim to answer the research question: What are the benefits of eating fruit? Specifically, you aim to test the hypotheses that (1) Apple consumption is positively associated with health; and (2) Orange consumption is positively associated with health. You have conducted a literature review and estimate the population effect size for the relationship between apple consumption and health to be $r = .15$ and the population effect size for the relationship between orange consumption and health to be $r = .25$. You expect that apple consumption and orange consumption do not covary.

You now wish to conduct a power analysis to determine the sample size at which you will achieve 80% power to detect both those effects simultaneously, when entered into the same multiple regression.

In this simple case, you may leave all of `pwrMultivars`'s default arguments in place. You must only specify effect sizes for the two predictor variables, the units in which those effect sizes are provided, and the sample size for which you wish to estimate power. While the effect sizes are known ($r = .15$ and $r = .25$), as are the units in which you will specify them (r), the sample size that will yield adequate power is not. For this reason, estimating power with `pwrMultivars` is an iterative process wherein you first estimate power for a sample size that seems reasonable to you, and then correct upward or downward depending on the result of the power analysis. You will then repeat the process until your desired level of power has been achieved. See also the section on the `pwrCurve` function for a way to guide this process.

You decide to begin by estimating power at $n = 100$.

```
pwrMultivars(n = 100,
```

```
        es_units = "r",  
        es1 = .15,  
        es2 = .25)  
  
## Predictor 1 Power: 33.72%  
## Predictor 2 Power: 71.96%  
## Simultaneous Power: 24.26%
```

Because that power analysis yielded simultaneous power of only 24.18%, far below your desired 80% power, you next revise your sample size upward and run the function again.

```
pwrMultivars(n = 400,  
             es_units = "r",  
             es1 = .15,  
             es2 = .25)  
  
## Predictor 1 Power: 87.6%  
## Predictor 2 Power: 99.96%  
## Simultaneous Power: 87.58%
```

This round, simultaneous power was 86.80%. Because this was substantially higher than your goal of 80% simultaneous power and you do not wish to spend resources recruiting more participants than you need, you decide to run a third power analysis with a smaller sample size to bring your power closer to 80%.

```
pwrMultivars(n = 330,  
             es_units = "r",  
             es1 = .15,  
             es2 = .25)  
  
## Predictor 1 Power: 81.1%
```

```
## Predictor 2 Power: 99.6%
```

```
## Simultaneous Power: 80.84%
```

This power analysis yielded 80.10% simultaneous power. Satisfied with this, you settle on $n = 330$ as the target sample size for your study.

A more complex vignette. Suppose that you now wish to expand the scope of your research question and, accordingly, the number of tests in your hypothesis set. You now aim to answer the research question: What are the benefits of eating fruit and vegetables? Specifically, you aim to test the following six hypotheses: (1) Apple consumption is positively associated with health; (2) Orange consumption is positively associated with health; (3) Broccoli consumption is strongly positively associated with health; and (4) Brussels sprout consumption is strongly positively associated with health. You have performed a literature review and anticipate the effect sizes corresponding to the previous four effects to be $R^2 = .02$, $R^2 = .06$, $R^2 = .20$, and $R^2 = .25$. You expect that broccoli consumption and brussels sprout consumption will covary at $r = .65$.

You now wish to conduct a power analysis to determine the sample size at which you will achieve 80% power to detect significant effects for the four effects simultaneously when entered into the same multiple regression.

You additionally wish to run 10,000 iterations in order to achieve a more reliable power estimate and you lower your alpha level to .01 just for fun. Furthermore, you want to set a seed of 100 instead of the default 1. You decide to first estimate power at $n = 400$. Note that covariances between predictors are always specified in correlation coefficients, regardless of the units in which your main effect sizes are specified.

```
pwrMultivars(n = 400,  
             es_units = "r2",  
             es1 = .02,
```



```
      es2 = .06,  
      es3 = .20,  
      es4 = .25,  
      iv3iv4_cov = .65,  
      alpha = .01,  
      iterations = 10000,  
      seed = 100)  
  
## Predictor 1 Power: 82.22%  
## Predictor 2 Power: 99.97%  
## Predictor 3 Power: 91.59%  
## Predictor 4 Power: 99.99%  
## Simultaneous Power: 75.25%
```

At $n = 400$, simultaneous power was 74.46%. So, you increase your sample size to 440 and try again

```
pwrMultivars(n = 440,  
      es_units = "r2",  
      es1 = .02,  
      es2 = .06,  
      es3 = .20,  
      es4 = .25,  
      iv3iv4_cov = .65,  
      alpha = .01,  
      iterations = 10000,  
      seed = 100)  
  
## Predictor 1 Power: 86.11%
```

```
## Predictor 2 Power: 99.99%  
## Predictor 3 Power: 94.45%  
## Predictor 4 Power: 100%  
## Simultaneous Power: 81.29%
```

You see that your simultaneous power is now 81.33%, which is above your minimum threshold of 80%. You therefore decide on $n = 440$ as your target sample size.

pwrMultimodels

`pwrMultivars` is another power analysis function in `SimulPower`. It simulates simultaneous power for a set of tests contained in between 2 and 10 separate models. Like `pwrMultivars`, it simulates data based on a correlation matrix imposed using the `mvrnorm` function from the `MASS` package (Venables & Ripley, 2002) and uses parallel processing using the `parallel`, `doParallel`, and `foreach` packages (Microsoft Corporation & Weston, 2019; Microsoft Corporation & Weston, 2020; R Core Time, 2020) to speed up processing.

`pwrMultimodels` estimates statistical power by running between 2 and 10 linear models in random samples drawn a population with a user-specified covariance matrix. In general, `pwrMultimodels` works the same way as `pwrMultivars`, preserving default values for arguments and argument names.

Arguments taken by `pwrMultimodels`:

n. Set the size of each sample to be drawn from the population. This is the sample size for which you are estimating statistical power. In other words, setting `n` to equal 100 will estimate statistical power at $n = 100$. Accepts any positive number. This argument has no default.

es__units. Set the units in which you are specifying your effect sizes. Accepts “d” for Cohen’s d, “r” for correlation coefficients, and “r2” for percent of variance accounted for. This argument has no default.

null_effect. For which, if any, of your models are you computing “null power?” If you want to compute “power” to NOT detect an effect, use this argument to specify which models are predicted nulls by setting this argument equal to the number(s) corresponding to the models you hypothesize to be null. Accepts either a single whole number between 1 and the number of models you have specified or a vector of numbers between 1 and the the number of models you have specified. Default = no null effects.

iterations. How many times you would like to run your models in random samples drawn from your population? One model will be run in each random sample. Accepts any whole number greater than 0. Default = 5000.

alpha. Set your alpha level. This is the threshold below which p-values will be considered significant. Accepts any number greater than 0 and less than 1. Default = 0.05.

bonferroni. Apply a bonferroni correction? This is suggested if you intend on interpreting the results of multiple tests individually, but not if you intend on assessing a single research question by triangulating across multiple tests (Le Forestier, Page-Gould, & Chasteen, Forthcoming). Accepts TRUE or FALSE. Default = FALSE.

seed. Set a seed to make your results reproducible. Accepts any number. Default = 1.

es1...es10. The effect size, expressed in units specified in the es_units argument, for each model. You should always specify these in order, beginning with es1, and not skipping any. Accepts any number. These arguments have no defaults.

iv1iv2_cov...iv9iv10_cov. The relationships between each set of predictors, specified in correlation coefficients. Specifying relationships between predictors is optional. Accepts any number between -1 and 1. Default = 0.

dv1dv2_cov...dv9dv10_cov. The relationships between each set of dependent variables, specified in correlation coefficients. Specifying relationships between DVs is optional. Accepts any number between -1 and 1. Default = 0.

iv1dv2_cov...iv9dv10_cov. The relationships between each set of predictors and dependent variables from separate models, specified in correlation coefficients. Specifying relationships between separate models' predictors and DVs is optional. Accepts any number between -1 and 1. Default = 0.

print_result. Should power analysis results be printed to the console? Accepts TRUE or FALSE. Default = TRUE.

The above information is also summarized in `pwrMultimodels`'s help page.

`?pwrMultimodels`

pwrMultimodels vignettes

With few exceptions, `pwrMultimodels` works in the same way as `pwrMultivars`.

A vignette. Continuing your now-burgeoning line of food research, you now want to expand your set of variables. You now wonder if brussels sprouts, milk chocolate, and dark chocolate might be associated with happiness.

You therefore aim to test the following specific hypotheses: (1) Milk chocolate consumption will be positively correlated with happiness; (2) Dark chocolate consumption will be positively correlated with happiness; and (3) Brussels sprouts consumption will be negatively associated with happiness. You want to test these hypotheses in separate models instead of a single model.

You have conducted a literature review and anticipate the effect sizes corresponding to the previous three hypotheses to be $d = 0.40$ (happiness ~ milk chocolate), $d = 0.45$ (happiness ~ dark chocolate), and $d = -0.75$ (happiness ~ brussels sprouts).

You now wish to conduct a power analysis to determine the sample size at which you will achieve 80% power to detect all three effects simultaneously. You decide to perform your first power analysis at $n = 200$.

```
pwrMultimodels(n = 200,  
               es_units = "d",  
               es1 = .40,  
               es2 = .45,  
               es3 = -.75)  
  
## Model 1 Power: 80.56%  
## Model 2 Power: 88.18%  
## Model 3 Power: 99.84%  
## Simultaneous Power: 70.62%
```

Simultaneous power for this set of tests is 71.40% at $n = 200$. Note that even though single power for each test is greater than 80%, simultaneous power is nonetheless lower than 80%. You increase your sample size to $n = 240$ and try again.

```
pwrMultimodels(n = 240,  
               es_units = "d",  
               es1 = .40,  
               es2 = .45,  
               es3 = -.75)  
  
## Model 1 Power: 87.16%  
## Model 2 Power: 93.62%  
## Model 3 Power: 99.96%  
## Simultaneous Power: 81.52%
```

You settle on $n = 240$ as your target sample size.

pwrCurve

As was previously described, the process of performing simultaneous power analysis with SimulPower is an iterative one wherein the user must first estimate what they believe to be a reasonable sample size and calculate simultaneous power at that sample size, then adjust the sample size upward or downward and repeat the process until they have achieved their desired level of power. To expedite this process and to provide the user with a more comprehensive understanding of the relationship between sample size and power for their hypothesis set, the `pwrCurve` function can be used in conjunction with any of the main power analysis functions in SimulPower to compute power estimates across a range of sample sizes and to visualize the corresponding power curve.

A suggested workflow for SimulPower involves first using the `pwrCurve` function to visualize the effect of sample size on simultaneous power across an range of sample sizes, then to use the information yielded from that procedure to inform which specific power analyses to conduct using the main power analysis functions. This workflow allows the user to efficiently conduct a large number of power analyses to guide them toward more informed estimates for final power analyses.

`pwrCurve` runs SimulPower's power analysis functions at a range of sample sizes to estimate simultaneous power at each size. To speed up this process, `pwrCurve` uses less intense defaults for its arguments than the other power analysis functions (i.e., fewer iterations per power analysis). However, with the exception of the “n” argument, which is replaced with “min” and “max” in `pwrCurve`, `pwrCurve` accepts all of the main power analysis functions' arguments, so this can be changed by the user if desired.

Arguments taken by `pwrCurve`:

method. Specify which SimulPower function you would like to use to computer simultaneous power. Accepts either “`pwrMultivars`” or “`pwrMultimodels`”. This argument

has no default.

min. Set the minimum sample size for which you would like to estimate simultaneous power. Accepts any positive number. This argument has no default.

max. Set the maximum sample size for which you would like to estimate simultaneous power. Accepts any positive number. This argument has no default.

increment. Set the increments by which you would like the sample size to increase between power analyses. Accepts any positive whole number. Default = 20.

thresholds. Set the power thresholds for which you would like sample size estimates printed. Accepts either any single number between 0 and 100 or a vector of numbers between 0 and 100. Default = `c(80, 90, 95)`.

es_units. Set the units in which you are specifying your effect sizes. Accepts “d” for Cohen’s d, “r” for correlation coefficients, and “r2” for percent of variance accounted for. This argument has no default.

null_effect. For which, if any, of your predictors or models are you computing “null power?” If you want to compute “power” to NOT detect an effect, use this argument to specify which effects are predicted nulls by setting this argument equal to the number(s) corresponding to the predictors you hypothesize to be null. Accepts either a single whole number between 1 and the number of predictors you have specified or a vector of numbers between 1 and the the number of predictors you have specified. Default = no null effects.

iterations. How many times you would like to run your model in random samples drawn from your population? One model will be run in each random sample. Accepts any whole number greater than 0. Default = 1000.

alpha. Set your alpha level. This is the threshold below which p-values will be considered significant. Accepts any number greater than 0 and less than 1. Default = 0.05.

bonferroni. Apply a bonferroni correction? This is suggested if you intend on interpreting the results of multiple tests individually, but not if you intend on assessing a single research question by triangulating across multiple tests (Le Forestier, Page-Gould, &

Chasteen, Forthcoming). Accepts TRUE or FALSE. Default = FALSE.

seed. Set a seed to make your results reproducible. Accepts any number. Default = 1.

es1...es10. The effect size, expressed in units specified in the `es_units` argument, for the relationship between each predictor and the dependent variable. You should always specify these in order, beginning with `es1`, and not skipping any. Accepts any number. These arguments have no defaults.

iv1iv2_cov...iv9iv10_cov. The relationships between each set of predictors, specified in correlation coefficients. Specifying relationships between predictors is optional unless your predictors, together, account for more than 100% of the variance in your DV, in which case you must specify relationships between your predictors to make that possible. Accepts any number between -1 and 1. Default = 0.

dv1dv2_cov...dv9dv10_cov. The relationships between each set of dependent variables, specified in correlation coefficients. Specifying relationships between DVs is optional. Accepts any number between -1 and 1. Default = 0.

iv1dv2_cov...iv9dv10_cov. The relationships between each set of predictors and dependent variables from separate models, specified in correlation coefficients. Specifying relationships between separate models' predictors and DVs is optional. Accepts any number between -1 and 1. Default = 0.

The above information is also summarized in `pwrCurve`'s help page.

?pwrCurve

pwrCurve vignettes

A vignette using pwrCurve with pwrMultivars. Presently, we return to the first vignette wherein you performed a simultaneous power analysis for two effects: (1) Apple consumption and health are associated at $r = .15$; and (2) orange consumption and health are associated at $r = .25$. This time, instead of immediately performing power analysis at a

specific sample size, you will produce a simultaneous power curve to estimate simultaneous power at a range of sample sizes, then use this information to inform the specific simultaneous power analyses you will run afterwards.

After reflecting on your effect sizes and the number of effects you aim to detect, you decide to visualize power between $n = 100$ and $n = 600$. You leave all of `pwrCurve`'s default arguments in place.

```
pwrCurve(method = "pwrMultivars",  
          min = 100,  
          max = 600,  
          es_units = "r",  
          es1 = .15,  
          es2 = .25)
```

```
## The 80% simultaneous power threshold was crossed between n = 300 and n = 320  
## The 90% simultaneous power threshold was crossed between n = 420 and n = 440  
## The 95% simultaneous power threshold was crossed between n = 560 and n = 580
```

The `pwrCurve` function visualized simultaneous power and also produced estimates for the sample sizes at which 80%, 90%, and 95% power were reached.

Because your goal is to power your study at 80%, you are able to narrow down the range of sample sizes you are interested in testing to between $n = 320$ and $n = 340$. Your next step is then to run a single power analysis using `pwrMultivars` with a larger number of iterations using a sample size between $n = 320$ and $n = 340$ to achieve a more stable power estimate. See the `pwrMultivars` section of this document for details.

A vignette using `pwrCurve` with `pwrMultimodels`. `pwrCurve` works much the same way with `pwrMultimodels` as it does with `pwrMultivars`. For example, let us return to the `pwrMultimodels` vignette (above) wherein you aimed to detect the following three effects: You therefore aim to test the following specific hypotheses: (1) Milk chocolate consumption will be positively correlated with happiness; (2) Dark chocolate consumption will be positively correlated with happiness; and (3) Brussels sprouts consumption will be negatively

associated with happiness.

Retaining the same hypothesized effect sizes from the original vignette ($d = 0.40$, $d = 0.45$, and $d = -0.75$), you decide to visualize a power curve for the set of tests at sample sizes between $n = 10$ and $n = 610$. Because of how wide this range is, you decide to increase the sample size between power analyses in increments of $n = 100$ (instead of the default $n = 20$).

```
pwrCurve(method = "pwrMultimodels",  
          min = 10,  
          max = 610,  
          increment = 100,  
          es_units = "d",  
          es1 = .40,  
          es2 = .45,  
          es3 = -.75)
```

```
## The 80% simultaneous power threshold was crossed between n = 210 and n = 310  
## The 90% simultaneous power threshold was crossed between n = 210 and n = 310  
## The 95% simultaneous power threshold was crossed between n = 310 and n = 410
```

Results indicate that 80% power was reached between $n = 210$ and $n = 310$. You now wish to achieve a more fine-grained estimate. So, you now visualize a second set of power curves, zooming in on the range of sample sizes between $n = 210$ and $n = 310$, increasing sample size in smaller increments.

```
pwrCurve(method = "pwrMultimodels",  
         min = 210,  
         max = 310,  
         increment = 10,  
         es_units = "d",  
         es1 = .40,  
         es2 = .45,  
         es3 = -.75)
```

The 80% simultaneous power threshold was crossed between $n = 220$ and $n = 230$

The 90% simultaneous power threshold was crossed between $n = 280$ and $n = 290$

The 95% simultaneous power threshold was not crossed.

You have now narrowed down the range of sample sizes at which 80.00% power may be achieved to between $n = 240$ and $n = 250$. At this stage, you elect to perform a single power analysis at a sample size within that range, using the `pwrMultimodels` function and a larger number of iterations. See the `pwrMultimodels` section of this document, above, for details.

Conclusion

This document has endeavoured to provide users with a walkthrough of using the basic features of the `SimulPower` R package to conduct simultaneous power analysis. For a detailed discussion of simultaneous power analysis, see Le Forestier, Page-Gould, & Chasteen (Forthcoming).

References

Le Forestier, J. M., Page-Gould, E., & Chasteen, A. (Forthcoming). Statistical power for a set of tests.

Microsoft Corporation & Weston, S. (2019). doParallel: Foreach parallel adaptor for the “parallel” package. [Computer Software].

Microsoft Corporation & Weston, S. (2020). foreach: Provides foreach looping construct. [Computer Software].

R Core Team. (2020). R: A language and environment for statistical computing. [Computer Software].

Venables, W. N. & Ripley, B. D. (2002). Modern applied statistics with S. Springer.