# Power analysis through simulations in Stata: a guide for dummies

Pol Campos-Mercade*

This version: September 2018

You want to run an experiment for which you expect to find a treatment effect. How big should the sample size be for you to have a reasonable chance of finding significant results? In this manuscript, I share and explain the Stata code that I use to answer this question. The code uses simulations to perform power analyses. In contrast to the already widely used analytical tools to perform power analyses, this method can accommodate any experimental design and statistical test that Stata can perform. The code is very simple, easy to use and will be useful for researchers without much coding experience.

**Keywords:** Power analysis, Simulations, Stata.

**JEL classification:** C15, C90.

* Lund University, Department of Economics, Tycho Brahes väg 1, Lund, Sweden. E-mail: pol.campos@nek.lu.se.

***Disclaimer**: Using simulations to do power analyses has been used for a long time. Feiveson (2002) and Arnold et al. (2011) explain in detail how to apply this method. There are even Stata packages (powersim, Luedicke 2013; powerBBK, Bellemare et al. 2016) and R packages (SIMR, Green and MacLeod 2016) to do specific types of power analyses through simulations. Despite the widespread use of such methods by statisticians, my experience is that most experimental economists still use analytical tools to perform power analyses. While such tools are useful for simple experimental designs, they are rigid and cannot accommodate many possible experiments and statistical tests. My purpose with this manuscript is to disseminate the code with which I often do power analyses in Stata and which many colleagues have found useful.*

## 1. Introduction

Doing a power analysis is a fundamental part of any experimental design. For most basic experimental designs, there exist equations that can perform a power calculation analytically (see for example List et al. 2011).[1] There are however many more complex designs in which analytical solutions do not exist or are very difficult to derive. For example, you might consider different clusters, treatment interactions, covariates, or non-normal distributions (for which you want to perform non-parametric tests). In these cases, using simulations is a very useful tool to perform your power calculations.

In this manuscript, I share the code that I use to perform such power calculations. The code is written in simple programing language and my goal is that any reader with basic notions of programing can understand it. It is also very flexible and

[1] G-Power (Faul et al. 2007) is a very useful tool to perform such basic power calculations. You can download it here: http://www.gpower.hhu.de.

should be able to accommodate any experimental design and statistical test that the reader would like to run. For any comments or questions, please feel free to contact me on pol.campos@nek.lu.se.

In Section 2, I define power and argue why doing power analyses is important for your research. In Section 3, I summarize the way in which the code works. Section 4 provides an example of how to use the code with a very simple example (for which there are analytical solutions). Section 5 discusses a more complex example (for which there are not analytical solutions).

## 2. Power

This section provides an overview of the concept of power. Please, skip it if you already feel comfortable with this concept.

### 2.1 What is power?

Power is the probability of detecting an effect, given that this effect is true. In other words, if the treatment that we want to test has an actual effect, the power tells us the probability that our experimental results will say that the treatment effect is statistically significant.

Experimenters typically use power calculations for two purposes. First, to calculate the sample size needed to have a reasonable chance of rejecting the null hypothesis of no effect (if the effect was really true). To calculate this sample size, you need first to specify an effect size for your alternative hypothesis (if your treatment will actually have an effect, how large will it be?). There are three main ways to choose this effect. First, based on your best guess. Second, based on results

from previous research.[2] Third, based on the minimum effect size that it would be interesting to find.[3]

Second, researchers may do a power analysis to understand how big the true effect should be to be able to capture it given a sample size. This is usually done when experimenters face a fixed sample size with which they can experiment (for example, the number of students in a class). Given this sample size, they can calculate how big the true effect should be to have, say, an 80% chance to find a significant result. Doing this kind of analysis is fundamental to understand whether an experiment is well designed or it is too ambitious.[4]

### 2.2 Why is it important to do power analyses?

I can think of at least five reasons on why doing power analyses is important:

1. It increases the chances that you will get statistically significant results. Even if there is a true effect, having only 40% power will result in a null result 60% of the times. Realizing this in time and increasing the sample size until you have, say, 80% power will both help you detect true effects and increase the credibility of your research.
2. It prevents you from overusing resources in the experiment. Experiments are typically expensive and time-consuming. By performing a power analysis,

[2] Some researchers also use results from pilot experiments to specify this effect. In general, I do not like this approach. Pilot experiments are usually run with very small sample sizes, leading to very noisy estimates. In many cases, I believe that these estimates are noisier than our guesses, and blindly following them can lead to big mistakes. I think however that pilot experiments are great to get an idea about the data generating process, which is also necessary to do the power analysis.

[3] Most treatments that we try will have *some* true effect. However, if this effect is 0.001 standard deviations, in most cases this will not be of any practical relevance.

[4] During the first year of my PhD studies, in which I hardly knew anything about what a power analysis was, I run a class experiment with 200 students. I divided them into a control group and two treatment groups of about 65 students each. I did not find any significant result. Afterwards, when I calculated the true effect that I had power to find, I realized that it was way too high. Had I done a power analysis, I would have only done one treatment and I would have had a higher chance to find something interesting. Right now, since I did not have power to find a reasonable effect, not even the null results that I found are interesting (they would only be interesting to be used in a meta-analysis). The data is in my drawer.

you can be sure that you do not have too big of a sample for the kind of effect that you expect to detect.

3. It increases the credibility of your null results. Obviously, finding a null result in an experiment with 20% power is not interesting; one does not learn much about whether the effect does not exist or whether it exists but was not found. Being able to argue that you had an 80% chance of finding a reasonable effect and that you did not find it makes your null result interesting.
4. It increases the precision of your estimates, which makes the treatment effect of your experiment more meaningful.
5. It benefits science. Since significant results from low power studies are less likely to reflect true effects, they will also be less likely to be replicated. This creates replication crises that erode the credibility of scientific knowledge.

## 3. Method

The simulation consists of the following steps:

1. Set the sample size of the simulated experiment.
2. Give a baseline value to each observation. This observation must be what we expect to observe in absence of a treatment effect. To choose it, you can use either of two options.
    a. Simulate a distribution using Stata's tools (for example, with the command *gen x = rnormal()*). To decide what distribution to draw from, you can either use previous results, use a dataset of the subjects that you are going to experiment with, use a pilot study, or just guess.
    b. Use data that already exists. If you happen to have data, you can bootstrap this data to fill out your simulated sample. For example, imagine that we want to test whether an intervention affects

students' GPA. If we have data from the previous cohort, we can fill out the observations with these data. If we have data on 100 students and we have set our simulated sample size to 200 observations, for each of the 200 observations we can randomly allocate the data of one of the 100 students (also called *boostrap*).

3. Randomly assign each observation to each treatment.
4. For those who have been assigned to the treatment group, add the expected treatment effect.
5. Run your test on the created dataset and store the p-value of the test.
6. Repeat steps 2-5 many times.
7. Count the number of times in which the p-value was lower than 0.05. This is your power. For example, if you repeated steps 2-5 a thousand times and you found that there were six hundred cases in which $p<0.05$, you can conclude that you have 60% power to find the effect size that you introduced in step 4.

I now present two different examples of how to apply this method. Note that the code is *not* the most efficient nor the most elegant. I have written it so that people with basic coding experience can clearly understand and learn to manipulate it.

## 4. Example 1

Now let us put this method to work with the simplest case. Imagine that you design a lab experiment in which subjects must solve a task as quickly as possible. You want to test whether incentivizing them to solve the task quickly (by paying them more the earlier they solve it) will reduce the time they need to complete it. For the sake of the example, imagine that:

1. You know (or guess) that the time they need to complete the task follows a normal distribution with a mean of 100 seconds and standard deviation of 20

seconds (note that in this case a lognormal distribution seems more reasonable, but now we are assuming the simplest of the cases).

2. You expect the true effect of incentivizing subjects to reduce their completion time by 5 seconds.
3. You want to have the same number of subjects in the control and in the treatment group.

How many subjects should you have in your study for you to have 80% power (i.e., for your experiment to reject the null hypothesis with 80% probability if the true effect was actually 5 seconds)? Let's code it!

```
**************************************
* This Stata code is for Stata 15. It should work with minor modifications in other Stata versions.
**************************************
* This code performs a power analysis for an experiment in which: the observations from the control group follow a normal(100,20); the true treatment effect is -5; the significance level is 0.05; the sample is 200, equally divided between control and treatment group. We will calculate what power this experiment has to find a significant result. You just have to copy and paste this in your Stata dofile.
**************************************
clear
set matsize 1000
mat estimates = J(1000,1,.)
* Creates a matrix of 1000 rows by 1 column. In each row, we will store the p-value of each simulation.
local subjects=200
local teffect=5
* Defines the number of subjects in our experiment and the treatment effect.
quietly forvalues j=1(1)1000 {
* Repeats the following code 1000 times.
clear
set obs `subjects'
gen id=_n
* Sets 200 observations and assign an ID to each one of them from 1 to 200.
gen treatment=0
replace treatment=1 if id >= `subjects'/2
* Assigns half of the subjects to the treatment group.
gen time=rnormal(100,20)
replace time=time-`teffect' if treatment==1
* Assigns an observation to each of the subjects. Those who are treated perform the task 5 seconds earlier.
ttest time, by(treatment)
```

```
scalar pvalue = r(p)
* Tests differences between the control and treated group. Stores the p-value of the test.
matrix estimates[`j',1] = pvalue
* Adds the p-value of the test to the row number "j" of column 1 on the 1000x1 matrix that we created.
noisily display `j'
* Shows you on which simulation we are at (personal preference)
}
* This experiment is repeated 1000 times.
svmat estimates, names(pvalues)
* Retrieves (and adds to our dataset) the 1000x1 matrix with the p-values of all the simulated experiments.
gen significant=0
replace significant=1 if pvalues<0.05
* Creates a variable with value 1 if the experiment was significant.
ci means significant
* It displays the percentage of experiments in which the test was significant (power) and its confidence interval. In this case, I obtained 41% power.
* Now you can play around with the number of observations and true treatment effect to see how power changes with a different sample size and treatment effect.
```

## 5. Example 2

Now let us take a slightly more complex example. Imagine that you would like to test whether incentivizing university students to attain a given grade boosts their GPA. You want to test this during the students' second semester, so that you can control for their first semester's GPA. For the sake of the example, imagine that:

1. There are 1000 students in this cohort and you have to decide how many of them to incentivize. Therefore, your goal now is *not* to decide your sample size but rather what percentage you treat.
2. You have data on the first semester GPA of the 1000 students of this cohort.
3. You have data on the first and second semester GPA of 1000 students of the previous cohort.
4. Students are divided in 4 groups of 250 students, for which there might be GPA fixed effects in the second semester.

5. We expect the treatment effect not to be constant. The treatment effect for each student will come from a normal distribution. Because we expect that a treatment effect of less than 2 (on average) would not be relevant, we will perform a power analysis assuming that the treatment effect comes from a normal distribution with mean 2 and standard deviation 2.
6. We want to perform three tests: a t-test, a Wilcoxon rank sum test, and a class fixed effects regression in which we control for the students' first semester GPA.

How many students should you incentivize to have 80% power? We will use the dataset of the previous cohort and simulate 1000 experiments with it. We will randomly pick some of the students in the sample and add a hypothetical treatment effect to their second semester GPA. We will then check the frequency in which the tests turned out to be significant (in other words, the power). This will help us decide how many students we should incentivize this year. Let's code it!

```
**************************************
* This Stata code is for Stata 15. It should work with minor modifications in other Stata versions.
**************************************
* This code performs a power analysis for an experiment in which we test whether incentivizing students boosts their GPA. We have: 1000 students to experiment with from which we know their first semester GPA; 1000 observations about the first and second semester GPA of students in the previous cohort (which go from 0, lowest grade, to 100, highest grade); the true treatment effect is distributed following a normal (2,2); there are four class groups with class fixed effects. Out of 1000 students, we have to decide how many students to incentivize to have 80% power the hypothesized effect.
**************************************
clear
set matsize 1000
mat estimates = J(1000,3,.)
* Creates a matrix of 1000 rows by 3 columns. In each row, we will store the p-value of each simulation. In each column, we will store the p-value of each different test.
* Now we would use the dataset of the previous cohort, in which each row corresponds to each student and it shows the students' first semester GPA (gpa1), second semester GPA (gpa2) and class group. Instead of importing a dataset, in this example we generate it (and we imagine that it is real).
set obs 1000
gen id=_n
gen classgroup=.
replace classgroup=1 if id<=250
```

```
replace classgroup=2 if id>250 & id<=500
replace classgroup=3 if id>500 & id<=750
replace classgroup=4 if id>750 & id<=1000
scalar fe1 = rnormal(0,10)
scalar fe2 = rnormal(0,10)
scalar fe3 = rnormal(0,10)
scalar fe4 = rnormal(0,10)
gen gpa1=rnormal(50,15)
gen gpa2r=rnormal(50,15)
gen gpa2=0.7*gpa1+0.3*gpa2r+fe1
replace gpa2=0.7*gpa1+0.3*gpa2r+fe2 if id>250 & id<=500
replace gpa2=0.7*gpa1+0.3*gpa2r+fe3 if id>500 & id<=750
replace gpa2=0.7*gpa1+0.3*gpa2r+fe4 if id>750 & id<=1000
drop gpa2r
* Generates the fake dataset in which gpa1 explains 70% of gpa2 and there are 4 class groups with their own fixed effects.
local treatedsubjects=500
local teffectscalar=1
* Defines the number of subjects that we are going to treat in this simulation and the treatment effect (these are the values to play around with).
quietly forvalues j=1(1)1000 {
* Repeats the following code 1000 times.
capture drop randomnumber
capture drop treatment
capture drop teffect
capture drop idnew
* We drop variables from the previous loop, if there are any.
gen randomnumber=runiform()
sort randomnumber
gen idnew=_n
* We generate a random number for each observation to sort all observations randomly. This is to randomly allocate the treatment (although, obviously, in this kind of design it would be better to stratify).
gen treatment=0
replace treatment=1 if idnew <= `treatedsubjects'
* Assigns the treatment to the number of subjects that we previously set as treated.
gen teffect=rnormal(`teffectscalar',2)
replace gpa2=gpa2+teffect if treatment==1
* Assumes that the treatment effect is normally distributed around what we decided in "teffectscalar" with a standard deviation of 2 (contrary to the previous example, we no longer assume that the treatment effect is constant across all subjects).
ttest gpa2, by(treatment)
scalar pvalue1 = r(p)
matrix estimates[`j',1] = pvalue1
* Performs a t-test and stores it in the first row of the matrix.
```

```
ranksum gpa2, by(treatment)
scalar pvalue2=2 * normprob(-abs(r(z)))
matrix estimates[`j',2] = pvalue2
```

* Performs a Wilcoxon rank sum test. The pvalue for this test has to be computed with the formula above (cannot be extracted with r(p) as with the t-test). Then stores it in the second row of the matrix.

```
areg gpa2 gpa1 treatment, absorb(classgroup)
        local q = _b[treatment]/_se[treatment]
        scalar pvalue3 = 2*ttail(e(df_r),abs(`q'))
matrix estimates[`j',3] = pvalue3
```

* Performs a regression with class fixed effects and controlling for gpa1. The pvalue for "treatment" is extracted from the formula above. Then it stores the pvalue in the third row of the matrix.

```
replace gpa2=gpa2-teffect if treatment==1
```

* Recall that we added the treatment effect to the dataset to make our test. For each simulation, we want to start from the basic dataset and the treatment effect to be different. So now we take the treatment effect away so that in the next loop we start with the same dataset as before.

```
noisily display `j'
```

* Shows you on which simulation we are at (personal preference).

```
}
```

* This experiment is repeated 1000 times.

```
svmat estimates, names(pvalues)
```

* Retrieves (and adds to our dataset) the 1000x3 matrix with the p-values of all the simulated experiments.

```
gen significant1=0
replace significant1=1 if pvalues1<0.05
```

* Creates a variable with value 1 if the test was significant.

```
gen significant2=0
replace significant2=1 if pvalues2<0.05
gen significant3=0
replace significant3=1 if pvalues3<0.05
ci means significant1
ci means significant2
ci means significant3
```

* It displays the percentage of experiments in which the test was significant (power) and its confidence interval for each of the tests. In my example, with 200 treated students, I got 13% power with the t-test, 12% power with the rank sum test, and 80% power with the regression test. If we wanted the t-test or the rank sum test to be our main test, we do not have enough power with 200 incentivized students. But if we want the regression test to be our main test, then we do have 80% power to find a significant result.

## 6. References


Arnold, B. F., Hogan, D. R., Colford, J. M., & Hubbard, A. E. (2011). Simulation methods to estimate design power: an overview for applied research. BMC medical research methodology, 11(1), 94.

Bellemare, C., Bissonnette, L., & Kröger, S. (2016). Simulating power of economic experiments: the powerBBK package. Journal of the Economic Science Association, 2(2), 157-168.

Feiveson, A. H. (2002). Power by simulation. Stata J, 2(2), 107-124.

Green, P., & MacLeod, C. J. (2016). SIMR: an R package for power analysis of generalized linear mixed models by simulation. Methods in Ecology and Evolution, 7(4), 493-498.

List, J. A., Sadoff, S., & Wagner, M. (2011). So you want to run an experiment, now what? Some simple rules of thumb for optimal experimental design. Experimental Economics, 14(4), 439.

Luedicke, J. (2013). Powersim: simulation-based power analysis for linear and generalized linear models. In 2013 Stata Conference (No. 13). Stata Users Group.