Valentinas Sungaila

Methods Paper: Power Analysis for a Continuous Variable in Multiple Regression

Introduction

Given a study with $x_1, \dots, x_n$ predictors, much of the interest is placed on the regression coefficients (β) and their relationship with the response variable (*y*). Because of how expensive studies can be, it is not possible to always collect a big sample size. Small sample sizes can lead to a lack in statistical power, which can prevent researchers from identifying relationships (usually for β ≠ 0) between the predictors and the response variable. Thus, to deal with this problem calculations for the sample needed can be made before a study takes place.

Using power analysis, statistical power for a study or a sample size to achieve that power can be determined. Statistical power is defined as the probability of finding a significant result (usually a relationship different from 0) when there exists a significant relationship. Power can be mathematically defined as *P(reject* $H_0 | H_1$ *is true)* when testing with a hypothesis test of H_0vs H_1. Power can also be calculated as 1 – Type 2 Error Rate. Statistical power depends on three other parameters as well. Significance level (α), effect size, and the sample size. Meaning that if three out of the four parameters are specified, then the fourth one can be calculated.

When calculating power, an assumption needs to be made about the distribution of the test statistic when doing a hypothesis test such as H_0: $\beta = 0$ vs H_1: $\beta \neq 0$. When a distribution is central, the distribution describes the test statistic when the difference tested is null. When the distribution is noncentral, the distribution describes the test statistic when the null is false. This paper will explore the method of power analysis, how the method can be applied, and how

a central and non-central f-distribution can be used to calculate the sample size for a given power.

Case Study

For the case study, a question was posed to determine what effect sizes for age of onset of vitiligo are needed to detect a significant result with the genetic variants at 80% power with three different levels of significance. To determine this a dataset with 46 genetic variants and 2190 observations was provided. The genetic principle components did not show any statistical significance and were not necessary for the regression which is shown below.

$$Age\ of\ Onset\ of\ Vitiligo = \beta_0 + \beta_1 * Genetic\ Variant$$

To calculate statistical power, a distribution needs to be determined by the test being performed. Given the regression above an analysis of variance (ANOVA) can be performed which leads to an F-distribution when testig the following statistic.

$$H_0: \beta_1 = 0$$

$$H_a: \beta_1 \neq 0$$

The way the effect size is dependent on the hypothesis test that is being performed. If the difference of means is desired Cohen's d could be used, which is the difference in means divided by the standard deviation. In the case of this study, the tests being run look at the variance explained by one factor, thus testing within the framework of an F-test, meaning Cohen's f2 can be used to determine the effect size.

Model and Methods

The F-distribution is made up of two independent chi square distributions. The non-centrality parameter for the F-distribution can be calculated as such.

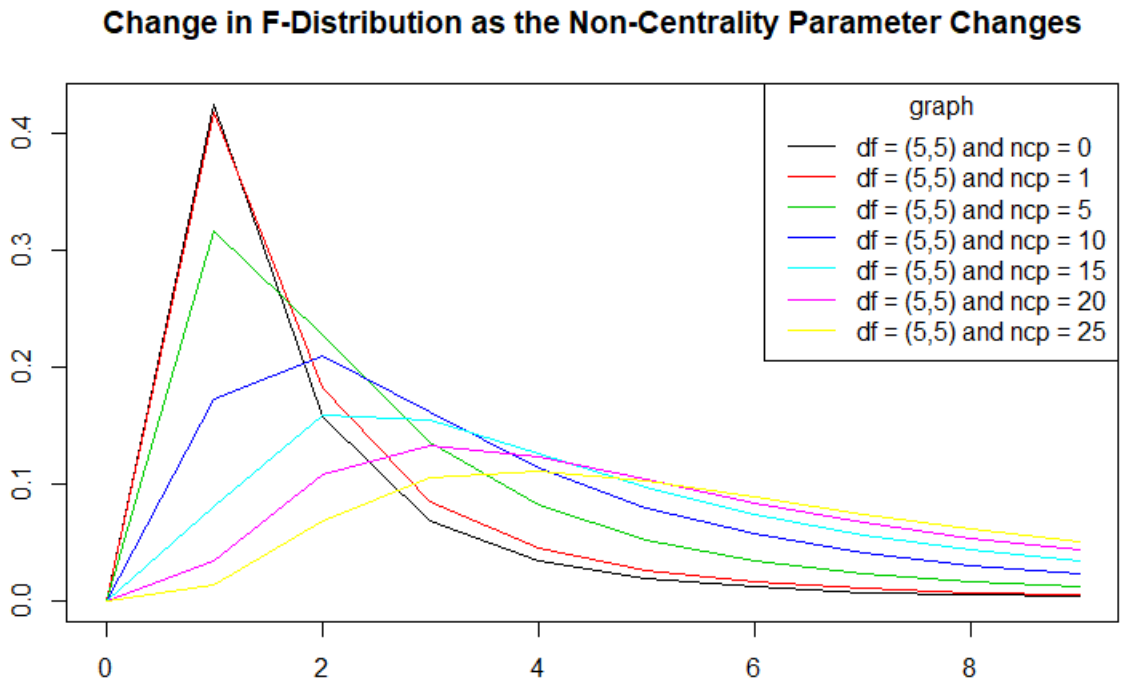
Equation 1

$$\lambda = \frac{n * \sum_{i=1}^{k} \beta_i^2}{\sigma_\varepsilon^2}$$

Where λ represents the non-centrality parameter, n is the number of observations per group, i indexes the groups, σ_ε^2 estimated of population variance, and β_i^2 is the is the difference between a group mean and the grand mean.

The graph below shows what happens to the F-distribution as the non-centrality parameter increase from zero.

Figure 1.



As it can be seen in figure 1 as the non-centrality parameter increases the F-distribution seems to move to the right.

Knowing this figure 2 shows how the non-centrality parameter effects power for different number of observations, keeping $\alpha = 0.05$, and having just one predictor without covariates.

Figure 2

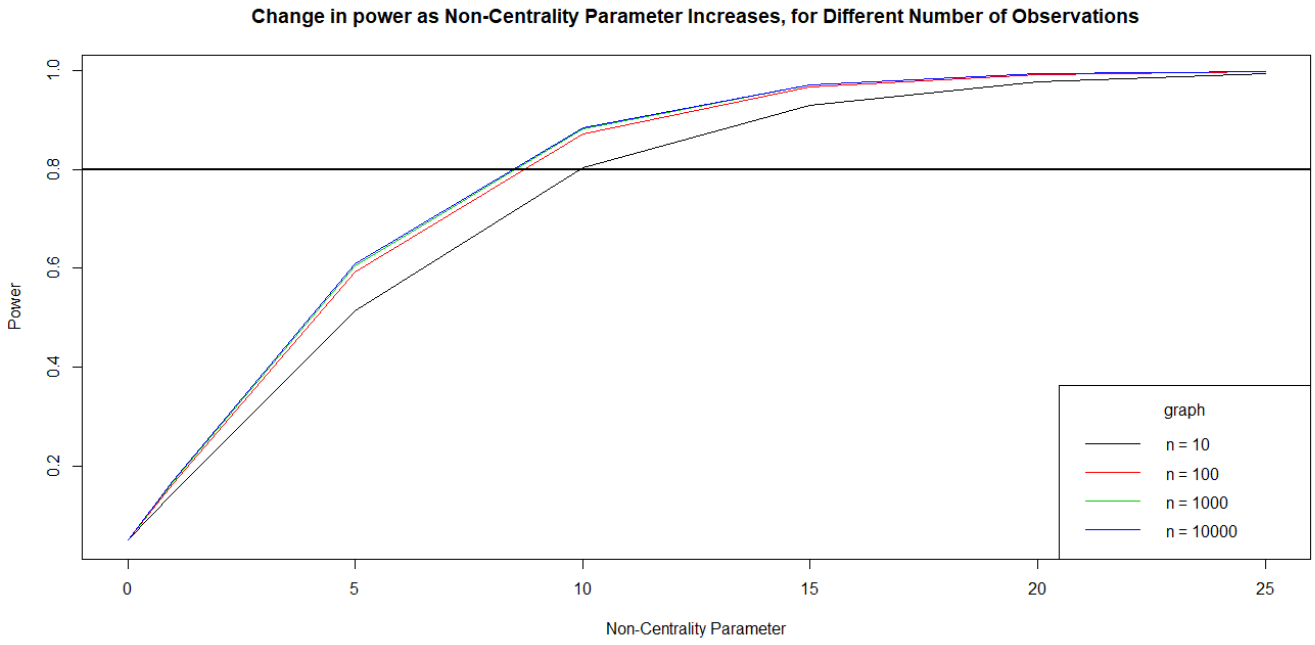


Figure 2 α =0.05 and df = 1

As it can be seen above the higher the non-centrality parameter and number of observations, the greater the power attained for a certain α and number of predictors.

To find the effect size a model needs to be defined first. Having $x_1, \dots, x_n$ predictors *y* response, a model can be defined.

Equation 2

$$Y_i = \beta_0 + \beta_1 X_{1i} + \cdots + \beta_n X_{ni} + \epsilon_i$$

Where β_i are the coefficients and ϵ_i the error term. Now the *X's* needs to be divided into two groups, first group being T(Tested) which are the predictors that we want to test (*k* of

them) and C(Covariates) which are the covariates included in the model (*L* of them). Which would make our model look something like this.

Equation 3

$$Y_i = \beta_0 + \beta_1 T_{1i} + \cdots + \beta_k T_{ki} + \mu_1 C_{1i} + \cdots + \mu_L C_{Li} + \epsilon_i$$

When doing power analysis only variables of interest are the ones needed to be tested which would make the test look like this.

$$H_0: \beta_1 = \beta_2 = \cdots = \beta_k = 0$$

$$H_a: \beta_1 \neq 0, \beta_2 \neq 0 \ldots \beta_k \neq 0$$

Since the *C* variables are not included in the test, but are fit in the regression analysis, any tests performed on the *k* tested predictors assumes that all *C* predictors are constant.

Once the variables to be tested are established, we can look at the test's statistic of the F-distribution. The focus in the significance test is how much *R^2* increases when test variables are added to a regression model that already contains the covariates. First different *R^2* values for the *k* predictors and the *L* predictors need to be defined. $R^2_{T|C} = R^2_{T,c} - R^2_C$ as the amount that *R^2* increases when y is regressed on the variables in the T set when keeping the C variables constant. R^2_C is the *R^2* when Y is regressed only on C variables. $R^2_{T,c}$ is the *R^2* when Y is regressed on the variables in both sets. Which gives the following tests statistic formula for the F-distribution.

Equation 4

$$F_{K,\ n-K-L-1} = \frac{R^2_{T|C}/K}{(1 - R^2_C - R^2_{T|C})/(n-k-L-1)}$$

Now to calculate power all the needs to happen is after calculating the non-centrality parameter λ, determine the critical value (*b*) of the test's statistic using a desired α as $F_{K,\ n-K-L-1,\ \alpha}$. Finally using your favorite statistical software compute the power as the probability of being greater than the calculated critical value in a in a noncentral-F distribution with non-centrality parameter λ. Or more mathematically as *Power = P[F(k, df, λ) ≥ b],* where $df = n - k - L - 1$ and *F(k, df, λ)* denotes a random variable with a non-central F distribution.

To see how *R^2* effects the sample size figure 3 shows the relationship between *R^2* and sample as the number of predictors to be tested increases at power or 0.8 and α = 0.05.

Figure 3

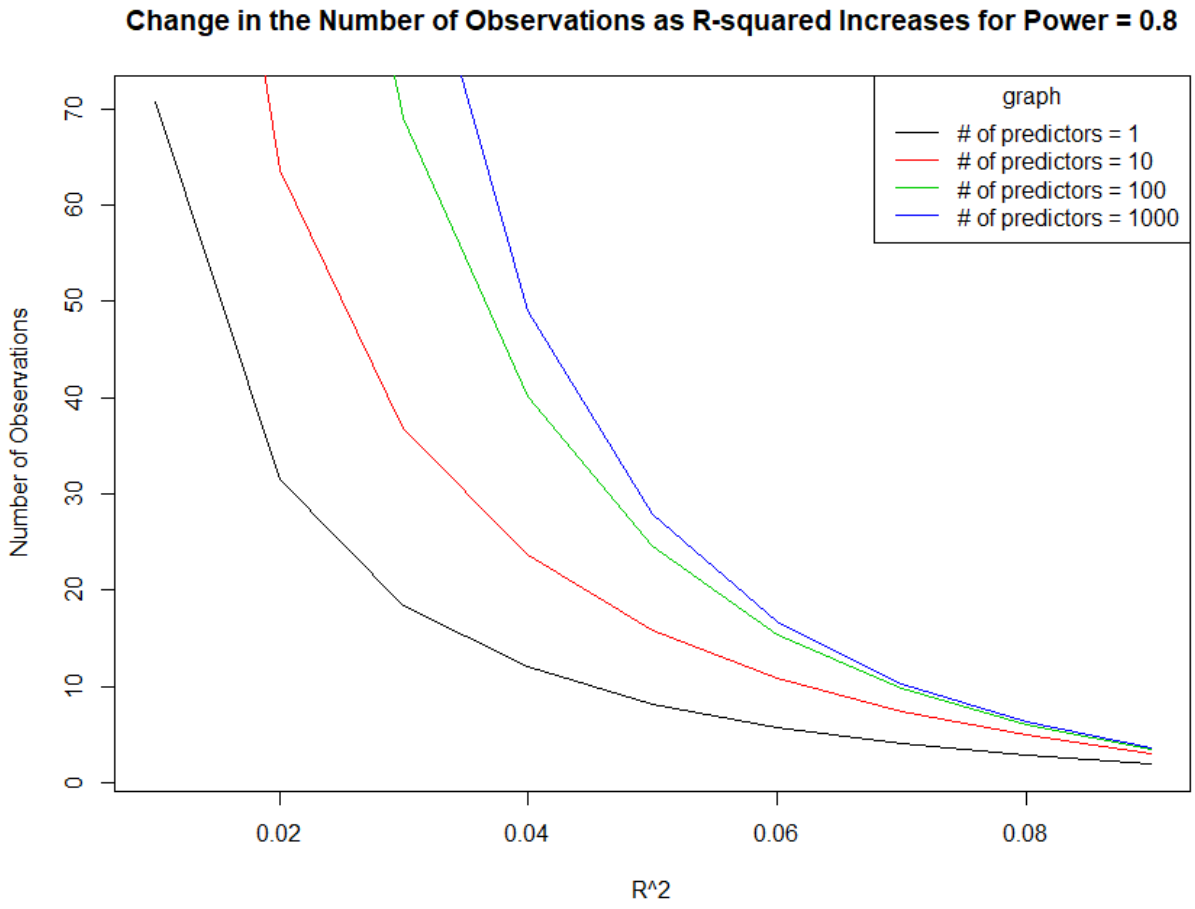


Figure 3 α = 0.5 and df = 1

As it can be observed as *R^2* increases a smaller and smaller sample size is needed to achieve power of 0.8. As the number of predictors needed to be tested increases, keeping *R^2* constant, more observations are needed to achieve the same power.

Given equation 3, *R^2* the effect size *Cohen's f^2* can be obtained.

Equation 5

$$f^2 = \frac{R^2_{T|C} - R^2_C}{\left(1 - R^2_{T|C}\right)}$$

Having the effect size from equation 5, the desired λ level, and the number of observations being tested *T*, and the non-centrality parameter, we can obtain the sample size for the desired power using the equation below.

Equation 6

$$f^2 = \frac{T}{\left(n(T+1)\right)} * F(t, \nu, \lambda)$$

Equation 6 can be manipulated where *n* is the value that is being calculated as such.

Equation 7

$$n = \frac{(T+1) * f^2}{(T) * F(t, \nu, \lambda)}$$

Analysis and Results

Power analysis can easily be done in R statistical software using the pwr.f2.test function in pwr R package (function calculation and methods in the appendix). To answer the posed question Figure 4 shows the value of *Cohen's f^2* for three different α levels. Unadjusted, α = 0.05, Bonferroni adjusted α = 0.05/46, and genome-wide adjusted α = 5e-8. The table also shows the *R^2* level the genetic variant needs to explain.

Figure 4

	Inputs			**Cohen's f^2**			**R-squared Effect Size**		
	Complete Observations	Power Desired	Effective DF	Unadjusted	Bonferroni Adjustment	Genome-wide adjustment	Unadjusted	Bonferroni Adjustment	Genome-wide adjustment
Age of Onset of Vitiligo	2,190	80%	1	0.0035889	0.00774619	0.01821105	0.0035761	0.00768665	0.01788534

Given the calculated effect sizes and the *R^2* level when looking at age of onset as the predictor the genetic variant RS114448410 is the only predictor that possesses a sufficiently high effect size to have at least 80% power when using the unadjusted α and the Bonferroni adjusted α. The effect disappears when genome-wide adjustment is used.

Discussion

Power analysis has shown that overall, 63 genetic variants do not have enough power to reach the 80% threshold at 3 different alpha levels, meaning that the only way to increase power and reach the 80% threshold is to increase the sample size. This can be difficult to do because studies like these can be very expensive and difficult to get data for.

The next step would be to see if more data is available. If that is not the case then the next step would be to see what other genetic markers have an effect on the age of onset of Vitiligo and see if those variables would be better chance at a higher *R^2* which would lead to a higher statistical power.

Power analysis is a great method because the method is very generalizable. No matter the model chosen for data analysis there is some way to calculate power, or effect sizes, or the number of observations needed to find an effect. The one shortcoming is that you are making assumptions about the testing distribution and sometimes the wrong one can be made. But there is enough literature to create confidence in the assumptions made.

In conclusion, power analysis is a useful method and should be preformed on any study where it is difficult to obtain more and more observations for a study because nobody wants to spend millions of dollars on a study and get no results.

Works Cited

Bruin, J. “FAQ HOW IS EFFECT SIZE USED IN POWER ANALYSIS?” IDRE Stats UCLA, 2016, stats.idre.ucla.edu/other/mult-pkg/faq/general/effect-size-power/faqhow-is-effect-size-used-in-power-analysis/.

Carlberg, C. “Informit.” InformIT, 11 Apr. 2013, www.informit.com/articles/article.aspx?p=2036567.

Cohen, J, et al. Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences. Third ed., LAWRENCE ERLBAUM ASSOCIATES, 2003.

“Effect Size.” Wikipedia, Wikimedia Foundation, 9 May 2020, en.wikipedia.org/wiki/Effect_size#Correlation_family:_Effect_sizes_based_on_%22variance_explained%22.

Faramawi, M, F., et al. “The Association between SNPs and a Quantitative Trait: Power Calculation.” European Journal of Environment and Public Health, vol. 2, no. 2, 2018, doi:10.20897/ejeph/3925.

Helwig, N. Effect Sizes and Power Analyses. 4 Jan. 2017, users.stat.umn.edu/~helwig/notes/espa-Notes.pdf.

“Multiple Regression.” NCSS.com, PASS Sample Size Software, ncss-wpengine.netdna-ssl.com/wp-content/themes/ncss/pdf/Procedures/PASS/Multiple_Regression.pdf.

Newsom. Sample Size and Power for Regression. 2020, web.pdx.edu/~newsomj/mvclass/ho_sample%20size.pdf.

“Noncentral F Distribution.” Noncentral F Distribution - MATLAB & Simulink, www.mathworks.com/help/stats/noncentral-f-distribution.html.

“Noncentral F-Distribution.” Wikipedia, Wikimedia Foundation, 8 May 2018, en.wikipedia.org/wiki/Noncentral_F-distribution.

“Noncentrality Parameter.” Wikipedia, Wikimedia Foundation, 13 Feb. 2020, en.wikipedia.org/wiki/Noncentrality_parameter.

“Package ‘Pwr.’” Cran.r-Project.org, 17 Mar. 2020, cran.r-project.org/web/packages/pwr/pwr.pdf.

“Power of a Test.” Wikipedia, Wikimedia Foundation, 29 Apr. 2020, en.wikipedia.org/wiki/Power_of_a_test.

Stephanie. “Non Centrality Parameter (NCP).” Statistics How To, 24 July 2018, www.statisticshowto.com/non-centrality-parameter-ncp/.

Index

pwr.f2.test function.

How the function works:

```
 p.body <- quote({
       lambda <- f2 * (u + v + 1)
       pf(qf(sig.level, u, v, lower = FALSE), u, v, lambda,
          lower = FALSE)
    })
f2 <- uniroot(function(f2) eval(p.body) - power, c(1e-07,
          1e+07))$root
```

Lambda is the non-centrality parameter of an F-test

As it can be seen the function first calculates lambda which is a non-centrality parameter then applies it to the probability of the f-distribution and quantile function. Finally calculating power for the f-distribution.

R code for the method

```
# Valentinas Sungaila
# Methods Paper Code
# Graphing an F distribution with df1 =5 and df2=5
# to show how the distribution changes as the noncentrality parameter changes
# Create the vector x
x <- seq(from = 0, to = 9, lenght = 200)
# Evaluate the densities
y_1 <- df(x, 5, 5, ncp = 0)
y_2 <- df(x, 5, 5, ncp = 1)
y_3 <- df(x, 5, 5, ncp = 5)
y_4 <- df(x, 5, 5, ncp = 10)
y_5 <- df(x, 5, 5, ncp = 15)
y_6 <- df(x, 5, 5, ncp = 20)
y_7 <- df(x, 5, 5, ncp = 25)

# Plot the densities
```

```
plot(x, y_1, col = 1, type = "l", ylab = '', xlab = '',
    main = "Change in F-Distribution as the Non-Centrality Parameter Changes")
lines(x, y_2, col = 2)
lines(x, y_3, col = 3)
lines(x, y_4, col = 4)
lines(x, y_5, col = 5)
lines(x, y_6, col = 6)
lines(x, y_7, col = 7)
# Add the legend
legend("topright", title = "graph",
    c("df = (5,5) and ncp = 0", "df = (5,5) and ncp = 1",
      "df = (5,5) and ncp = 5", "df = (5,5) and ncp = 10",
      "df = (5,5) and ncp = 15", "df = (5,5) and ncp = 20",
      "df = (5,5) and ncp = 25"),
    col = c(1, 2, 3, 4, 5, 6, 7), lty = 1)

###-----------------------------------------------------------------------------

# Computing power for F-distribution
# calculate power with 6 different ncp values and changing the number of observations in model
# n = 10
a = 0.05  # type1 error
n1 = 10 # sample size
p1 = 0 # other covariates
d = 1 # number of predictors we are interested in
v = n1 - p1 - d
c = qf(1-a, d, v)
ncp = c(0, 1, 5, 10, 15, 20, 25)
power_ncp.change1 = pf(c, d, v, ncp, lower.tail = FALSE)

# n = 50
n2 = 50 # other covariates
v2 = n2 - p1 - d
c2 = qf(1-a, d, v2)
power_ncp.change2 = pf(c2, d, v2, ncp, lower.tail = FALSE)

# n = 200
```

```
n3 = 200 # other covariates
v3 = n3 - p1 - d
c3 = qf(1-a, d, v3)
power_ncp.change3 = pf(c3, d, v3, ncp, lower.tail = FALSE)

# n = 10000
n4 = 10000 # other covariates
v4 = n4 - p1 - d
c4 = qf(1-a, d, v4)
power_ncp.change4 = pf(c4, d, v4, ncp, lower.tail = FALSE)

plot(ncp,power_ncp.change1,
   type = "l",
   main = "Change in power as Non-Centrality Parameter Increases, for Different Number of Observations",
   ylab = "Power",
   xlab = "Non-Centrality Parameter")

lines(ncp, power_ncp.change2, col = 2)
lines(ncp, power_ncp.change3, col = 3)
lines(ncp, power_ncp.change4, col = 4)

# Add the legend
legend("bottomright", title = "graph",
    c("n = 10", "n = 100",
      "n = 1000", "n = 10000"),
    col = c(1, 2, 3, 4), lty = 1)

# add the 80% threshold line
abline(h = 0.80, lwd = 2)
###-----------------------------------------------------------------------------------------------------------
# graphing # of parameters as expalined by the proportion of the variation of the predictor,
# for power of .8
library(pwr)
u = 1 # # number of predictors we are interested in
# v is the calculated sample size
power = 0.80 # the desired power level
```

```
number.of.observations_R2.0.1 = pwr.f2.test(u = u, f2 = 0.1/(1-0.1), sig.level = 0.05, power = power)
number.of.observations_R2.0.2 = pwr.f2.test(u = u, f2 = 0.2/(1-0.2), sig.level = 0.05, power = power)
number.of.observations_R2.0.3 = pwr.f2.test(u = u, f2 = 0.3/(1-0.3), sig.level = 0.05, power = power)
number.of.observations_R2.0.4 = pwr.f2.test(u = u, f2 = 0.4/(1-0.4), sig.level = 0.05, power = power)
number.of.observations_R2.0.5 = pwr.f2.test(u = u, f2 = 0.5/(1-0.5), sig.level = 0.05, power = power)
number.of.observations_R2.0.6 = pwr.f2.test(u = u, f2 = 0.6/(1-0.6), sig.level = 0.05, power = power)
number.of.observations_R2.0.7 = pwr.f2.test(u = u, f2 = 0.7/(1-0.7), sig.level = 0.05, power = power)
number.of.observations_R2.0.8 = pwr.f2.test(u = u, f2 = 0.8/(1-0.8), sig.level = 0.05, power = power)
number.of.observations_R2.0.9 = pwr.f2.test(u = u, f2 = 0.9/(1-0.9), sig.level = 0.05, power = power)

number_of_observations_per_R2 = rbind("R2 = 0.01" = number.of.observations_R2.0.1$v,
                      "R2 = 0.02" = number.of.observations_R2.0.2$v,
                      "R2 = 0.03" = number.of.observations_R2.0.3$v,
                      "R2 = 0.04" = number.of.observations_R2.0.4$v,
                      "R2 = 0.05" = number.of.observations_R2.0.5$v,
                      "R2 = 0.06" = number.of.observations_R2.0.6$v,
                      "R2 = 0.07" = number.of.observations_R2.0.7$v,
                      "R2 = 0.08" = number.of.observations_R2.0.8$v,
                      "R2 = 0.09" = number.of.observations_R2.0.9$v)

number_of_observations_per_R2= cbind(number_of_observations_per_R2,
                      c(0.01,
                      0.02,
                      0.03,
                      0.04,
                      0.05,
                      0.06,
                      0.07,
                      0.08,
                      0.09))
number_of_observations_per_R2_1 = as.data.frame(number_of_observations_per_R2)
#####
u = 10 # # number of predictors we are interested in
# v is the calculated sample size
power = 0.80 # the desired power level

number.of.observations_R2.0.1 = pwr.f2.test(u = u, f2 = 0.1/(1-0.1), sig.level = 0.05, power = power)
```

```
number.of.observations_R2.0.2 = pwr.f2.test(u = u, f2 = 0.2/(1-0.2), sig.level = 0.05, power = power)
number.of.observations_R2.0.3 = pwr.f2.test(u = u, f2 = 0.3/(1-0.3), sig.level = 0.05, power = power)
number.of.observations_R2.0.4 = pwr.f2.test(u = u, f2 = 0.4/(1-0.4), sig.level = 0.05, power = power)
number.of.observations_R2.0.5 = pwr.f2.test(u = u, f2 = 0.5/(1-0.5), sig.level = 0.05, power = power)
number.of.observations_R2.0.6 = pwr.f2.test(u = u, f2 = 0.6/(1-0.6), sig.level = 0.05, power = power)
number.of.observations_R2.0.7 = pwr.f2.test(u = u, f2 = 0.7/(1-0.7), sig.level = 0.05, power = power)
number.of.observations_R2.0.8 = pwr.f2.test(u = u, f2 = 0.8/(1-0.8), sig.level = 0.05, power = power)
number.of.observations_R2.0.9 = pwr.f2.test(u = u, f2 = 0.9/(1-0.9), sig.level = 0.05, power = power)
number_of_observations_per_R2 = rbind("R2 = 0.01" = number.of.observations_R2.0.1$v,
                      "R2 = 0.02" = number.of.observations_R2.0.2$v,
                      "R2 = 0.03" = number.of.observations_R2.0.3$v,
                      "R2 = 0.04" = number.of.observations_R2.0.4$v,
                      "R2 = 0.05" = number.of.observations_R2.0.5$v,
                      "R2 = 0.06" = number.of.observations_R2.0.6$v,
                      "R2 = 0.07" = number.of.observations_R2.0.7$v,
                      "R2 = 0.08" = number.of.observations_R2.0.8$v,
                      "R2 = 0.09" = number.of.observations_R2.0.9$v)

number_of_observations_per_R2= cbind(number_of_observations_per_R2,
                      c(0.01,
                      0.02,
                      0.03,
                      0.04,
                      0.05,
                      0.06,
                      0.07,
                      0.08,
                      0.09))
number_of_observations_per_R2_2 = as.data.frame(number_of_observations_per_R2)
#####
u = 100 # # number of predictors we are interested in
# v is the calculated sample size
power = 0.80 # the desired power level
number.of.observations_R2.0.1 = pwr.f2.test(u = u, f2 = 0.1/(1-0.1), sig.level = 0.05, power = power)
number.of.observations_R2.0.2 = pwr.f2.test(u = u, f2 = 0.2/(1-0.2), sig.level = 0.05, power = power)
number.of.observations_R2.0.3 = pwr.f2.test(u = u, f2 = 0.3/(1-0.3), sig.level = 0.05, power = power)
number.of.observations_R2.0.4 = pwr.f2.test(u = u, f2 = 0.4/(1-0.4), sig.level = 0.05, power = power)
```

```
number.of.observations_R2.0.5 = pwr.f2.test(u = u, f2 = 0.5/(1-0.5), sig.level = 0.05, power = power)
number.of.observations_R2.0.6 = pwr.f2.test(u = u, f2 = 0.6/(1-0.6), sig.level = 0.05, power = power)
number.of.observations_R2.0.7 = pwr.f2.test(u = u, f2 = 0.7/(1-0.7), sig.level = 0.05, power = power)
number.of.observations_R2.0.8 = pwr.f2.test(u = u, f2 = 0.8/(1-0.8), sig.level = 0.05, power = power)
number.of.observations_R2.0.9 = pwr.f2.test(u = u, f2 = 0.9/(1-0.9), sig.level = 0.05, power = power)

number_of_observations_per_R2 = rbind("R2 = 0.01" = number.of.observations_R2.0.1$v,
                      "R2 = 0.02" = number.of.observations_R2.0.2$v,
                      "R2 = 0.03" = number.of.observations_R2.0.3$v,
                      "R2 = 0.04" = number.of.observations_R2.0.4$v,
                      "R2 = 0.05" = number.of.observations_R2.0.5$v,
                      "R2 = 0.06" = number.of.observations_R2.0.6$v,
                      "R2 = 0.07" = number.of.observations_R2.0.7$v,
                      "R2 = 0.08" = number.of.observations_R2.0.8$v,
                      "R2 = 0.09" = number.of.observations_R2.0.9$v)

number_of_observations_per_R2= cbind(number_of_observations_per_R2,
                     c(0.01,
                       0.02,
                       0.03,
                       0.04,
                       0.05,
                       0.06,
                       0.07,
                       0.08,
                       0.09))

number_of_observations_per_R2_3 = as.data.frame(number_of_observations_per_R2)
#####
u = 1000 # # number of predictors we are interested in
# v is the calculated sample size
power = 0.80 # the desired power level

number.of.observations_R2.0.1 = pwr.f2.test(u = u, f2 = 0.1/(1-0.1), sig.level = 0.05, power = power)
number.of.observations_R2.0.2 = pwr.f2.test(u = u, f2 = 0.2/(1-0.2), sig.level = 0.05, power = power)
number.of.observations_R2.0.3 = pwr.f2.test(u = u, f2 = 0.3/(1-0.3), sig.level = 0.05, power = power)
number.of.observations_R2.0.4 = pwr.f2.test(u = u, f2 = 0.4/(1-0.4), sig.level = 0.05, power = power)
```

```
number.of.observations_R2.0.5 = pwr.f2.test(u = u, f2 = 0.5/(1-0.5), sig.level = 0.05, power = power)
number.of.observations_R2.0.6 = pwr.f2.test(u = u, f2 = 0.6/(1-0.6), sig.level = 0.05, power = power)
number.of.observations_R2.0.7 = pwr.f2.test(u = u, f2 = 0.7/(1-0.7), sig.level = 0.05, power = power)
number.of.observations_R2.0.8 = pwr.f2.test(u = u, f2 = 0.8/(1-0.8), sig.level = 0.05, power = power)
number.of.observations_R2.0.9 = pwr.f2.test(u = u, f2 = 0.9/(1-0.9), sig.level = 0.05, power = power)

number_of_observations_per_R2 = rbind("R2 = 0.01" = number.of.observations_R2.0.1$v,
                      "R2 = 0.02" = number.of.observations_R2.0.2$v,
                      "R2 = 0.03" = number.of.observations_R2.0.3$v,
                      "R2 = 0.04" = number.of.observations_R2.0.4$v,
                      "R2 = 0.05" = number.of.observations_R2.0.5$v,
                      "R2 = 0.06" = number.of.observations_R2.0.6$v,
                      "R2 = 0.07" = number.of.observations_R2.0.7$v,
                      "R2 = 0.08" = number.of.observations_R2.0.8$v,
                      "R2 = 0.09" = number.of.observations_R2.0.9$v)

number_of_observations_per_R2= cbind(number_of_observations_per_R2,
                     c(0.01,
                       0.02,
                       0.03,
                       0.04,
                       0.05,
                       0.06,
                       0.07,
                       0.08,
                       0.09))

number_of_observations_per_R2_4 = as.data.frame(number_of_observations_per_R2)
#####
plot(number_of_observations_per_R2_1$V2, number_of_observations_per_R2_1$V1,
   type = "l",
   xlab = "R^2",
   ylab = "Number of Observations",
   main = "Change in the Number of Observations as R-squared Increases for Power = 0.8")


lines(number_of_observations_per_R2_1$V2, number_of_observations_per_R2_2$V1, col = 2)
```

```
lines(number_of_observations_per_R2_1$V2, number_of_observations_per_R2_3$V1, col = 3)
lines(number_of_observations_per_R2_1$V2, number_of_observations_per_R2_4$V1, col = 4)

# Add the legend
legend("topright", title = "graph",
   c("# of predictors = 1", "# of predictors = 10",
     "# of predictors = 100", "# of predictors = 1000"),
   col = c(1, 2, 3, 4), lty = 1)
######
######### Libraries #########
library(data.table)
library(nnet)
library(rms)
library(lmtest)
library(pwr)

######### Data #########
dir_path <- "C:/Users/sunga/Desktop/consulting/" #WALDO, change me
dat <- as.data.frame(fread(paste0(dir_path,"subphenAndGenoVarsToUse.csv")))

######### Data Hygiene #########
colnames(dat) <- gsub("bcsnp.vitiligo_assoc","vit_ass",colnames(dat))
colnames(dat) <- gsub("bcsnp.vitiligo","vit_ass",colnames(dat))
colnames(dat) <- gsub("bcsnp.patient","patient",colnames(dat))

dat$vit_ass.rheumatoid_arthritis <- as.integer(ifelse(dat$vit_ass.rheumatoid_arthritis=='yes',1,0))
dat$vit_ass.systemic_lupus_erythematosus <- as.integer(ifelse(dat$vit_ass.systemic_lupus_erythematosus=='yes',1,0))
dat$vit_ass.pernicious_anemia <- as.integer(ifelse(dat$vit_ass.pernicious_anemia=='yes',1,0))
dat$vit_ass_onset.acrofacial <- as.integer(ifelse(dat$vit_ass_onset.acrofacial=='yes',1,0))
dat$vit_ass.halo_nevi <- as.integer(ifelse(dat$vit_ass.halo_nevi=='yes',1,0))
dat$vit_ass_onset.koebner_phenomenon <- as.integer(ifelse(dat$vit_ass_onset.koebner_phenomenon=='yes',1,0))

genotype_cols <- grep('^RS',colnames(dat),value=T)
covariates_cols <- grep('^GWAS',colnames(dat),value=T)
overlay_cols <- grep('patient.|vit_ass',colnames(dat),value=T)
dependent_cols <- grep('SKIN|DURATION',colnames(dat),value=T)
score_cols <- grep("^score",colnames(dat),value = T)
```

```
dat$SKIN <- factor(dat$SKIN,ordered=TRUE,levels=c('up to 25%','26-50%','51-75%','76-100%'))
######### PRE-PROCESSING: Picking top Ancestry Principal Components #########

# All gwas are populated otherwise we'd have to filter down to full observations
fwer_value <- 0.10/length(covariates_cols) # Using family-wise error adjustment (alpha/number of tests)
ageonset_keep <- c()
skin_keep <- c()

# Step 1 - Duration
mean_model_age <- lm(patient.vitiligo_age_of_onset ~ 1, data = dat)
for(count in covariates_cols){
  temp_model <- lm(paste0("patient.vitiligo_age_of_onset ~ ",count),data = dat)
  fwer_test <- anova(mean_model_age,temp_model)$`Pr(>F)`[2] < fwer_value
  if(fwer_test){ageonset_keep <- c(ageonset_keep,count)}
}
######### QUESTION 2 - Duration #########
# Q2: What effect sizes for disease characteristics are detectable with >= 80% power
# without adjustment for multiple testing.
# with adjust for the number of tests in Q1,
# and with adjustment genome-wide?

n_ageonset <- nrow(dat[!is.na(dat$patient.vitiligo_age_of_onset),])
df_num = 2-1 # Predictors minus intercept. We don't have any covariates to account for, otherwise that'd go here
effect_unadj <- pwr.f2.test(u = df_num, v=n_ageonset-1-1, sig.level = 0.05, power = 0.8)
effect_bonferroniadj <- pwr.f2.test(u = df_num, v=n_ageonset-1-1, sig.level = 0.05/46, power = 0.8)
effect_genomewide <- pwr.f2.test(u = df_num, v=n_ageonset-1-1, sig.level = 3.5e-8, power = 0.8)

# Effect (R^2) needed for 80% power with a 2190 sample size:
eff_unadj <- effect_unadj$f2/(1+effect_unadj$f2)  #0.003576067
eff_adj <- effect_bonferroniadj$f2/(1+effect_bonferroniadj$f2) #0.007686651
eff_genadj <- effect_genomewide$f2/(1+effect_genomewide$f2) #0.01823952

age_df$effect_largeenough_unadj <- ifelse(age_df$Rsqd>=eff_unadj,'Sufficient effect','Insufficient effect')
age_df$effect_largeenough_bonfadj <- ifelse(age_df$Rsqd>=eff_adj,'Sufficient effect','Insufficient effect')
age_df$effect_largeenough_genomeadj<- ifelse(age_df$Rsqd>=eff_genadj,'Sufficient effect','Insufficient effect')
# Only RS114448410 has a big enough effect size for 80% power at the unadjusted and bonferroni adjustment level. None at genone-wide
```