Details on power and the noncentral t-distribution why it is needed and how to use it

Alan E. Berger 10 May 2021 v27

This discussion is intended as additional material on statistical power supplementing the section on power in the Statistical Inference Course in the Johns Hopkins University Data Science Specialization on Coursera. This goes into considerable detail and is well beyond what is expected for the class but might be helpful for additional information or for future reference. Note the reader should not infer any endorsement or recommendation or approval for the material in this article from any of the sources or persons cited herein or from any other entities mentioned in this article. This article follows along the lines of the section on power in the Statistical Inference Course (frequent references to corresponding material there are given), however the intent is that this article will be self-contained. This article also contains considerable mathematical details.

Outline - topics covered

Power for one group Z-tests, using the distribution of the Z-score for the null hypothesis and for a specific alternative hypothesis; conceptual plots for one-sided and two-sided power; and example calculations using R's pnorm function

Statistical significance and practical significance, effect size

Power for t-tests; why the noncentral t-distribution is needed and how to use it; comments on one-sided tests - only use when appropriate; example calculations for one group t-tests

Power for two group t-tests; the Welch (unequal variance) t-test is generally preferable; a straightforward function to calculate power for the Welch t-test using the noncentral t-distribution, allowing for unequal sample sizes and unequal standard deviations; the **strict** option; testing this function vs. power values obtained by the power_t_test function from the MESS package; finding the number of samples needed for a given power by constructing a data frame of power values covering relevant sample sizes nx, ny, for both groups; finding the optimal choice for nx, ny, under the constraint that nx + ny = some fixed value N: generally will "want" more samples allocated to the group that has the higher standard deviation

Review what power is and how to calculate it

We first review what power is and how to calculate it, using Z-tests to get a conceptual view, and then power for t-tests will be covered in detail since t-tests are frequently used. Why the noncentral t-distribution is necessary for calculating power for t-tests will be explained. The next article will describe using Monte-Carlo random sampling to calculate power for t-tests, which provides a good example of the usefulness and flexibility of random sampling for calculating a statistical value, for when there is no function available to do so. This

can also be used as an independent check that one is using an R power calculation function correctly.

Power

As well covered in the course material, power (1 - β) is the probability that the statistical test being carried out will reject the null hypothesis (at some given significance level α) when in fact the alternative hypothesis is valid. One is assuming the conditions for the statistical test to be valid are satisfied, for example, for a t-test, the population(s) being examined has a normal distribution, and the samples are random and (unless it is a paired study) independent. R provides the power.t.test function for power calculations for the t-test, and there are many R packages for doing power calculations for a variety of statistical tests - see https://cran.r-project.org/web/packages/available_packages_by_name.html One should also search the web for packages that do power calculations for the statistical test of interest, since power may not be in the short package description as given in this web site. For example the MESS package by Claus Ekstrom has the description "Miscellaneous Esoteric Statistical Scripts" but it includes the **power_t_test** function (note the underscores, rather than periods, in its name) which does power calculations for t-tests also including two group (two.sample) t-tests where the standard deviations in the two groups can be different, and the sample sizes in the two groups can be different. It is important to do a check of calculated values against example(s) where the correct answer is known or can be independently calculated (or at least check the results from several different packages and see if the results agree). Note one can check (with some algebra) that when the sample sizes are equal and the sample standard deviations are equal, then the Welch (unequal variance) t-test and the Student's t-test are the same (the standard error for the difference of the group means, and the value of the degrees of freedom are both the same) (and so then resulting values of power will be the same).

Note the textbook for the Statistical Inference course

[1] Brian Caffo, **Statistical inference for data science**, (Leanpub, last updated on 2016-05-23, https://leanpub.com/LittleInferenceBook

and the lectures and swirl lessons for the Statistical Inference course use the notation $N(\mu, \sigma^2)$ for a normal distribution with mean μ and standard deviation σ, while the R functions dnorm, pnorm, qnorm, rnorm dealing with the normal distribution have the user specify the mean and sd (standard deviation, σ) when calling these functions (the latter notation $N(\mu, \sigma)$ will be used here).

First we will set up the notation we will use (pretty much a review of what is covered in the course). As in the course, we will first use as an example a one-sided Z-test (i.e., when one knows the true standard deviation) for whether the mean μ of a population that has a normal distribution is $>$ some value μ_0. Denote the standard deviation of the population by σ. One takes n independent random samples from the population, whose values we denote by $x_1, ..., x_n$, calculates their mean, which I'll denote by m, and their standard deviation (the sample standard deviation denoted by s). If one knows what σ is, then one can use the

Z-test (and use the pnorm function to calculate the power), otherwise one uses the sample standard deviation s as the value for σ and does a t-test. The p-value from the Z-test is the p-value coming from the Z-score, denoted by ZS:

$$ZS = \frac{(m - \mu_0)}{\sigma/\sqrt{n}} \qquad \text{equation (1)}$$

ZS can be viewed as a random variable since it is a function of the values of the n random samples $x_1, ..., x_n$; it has a particular value given the values in a particular random sample of size n. If the null hypothesis is valid (the mean of the population is μ_0), then the distribution of ZS is a standard normal distribution (mean 0 and standard deviation 1). This is the case since the expected value of the mean of independent random samples from a population is the population mean (here μ_0, given the null hypothesis), and the standard error of the mean, $s/\sqrt{n}$, is the standard deviation of the random variable m (m is the mean of n independent random samples from the population).

If the mean of the population being sampled is in fact equal a value μ_a (and the standard deviation is σ), then the distribution of the Z-score ZS in equation (1) is a normal distribution with standard deviation 1 and mean equal

$$\frac{(\mu_a - \mu_0)}{\sigma/\sqrt{n}}$$

For a given random sample $x_1, ..., x_n$, the resulting one-sided p-value from the one group (one.sample) Z-test with alternative "greater" is then the area under the probability density function (**pdf**) for the normal distribution N(0, 1) having mean 0 and standard deviation 1, to the right of ZS. This p-value can be found using the R function pnorm via `pnorm(ZS, mean = 0, sd = 1, lower.tail = FALSE)`. The p-value for the two-sided Z-test (for the alternative hypothesis: the mean is different from μ_0) is: the area under the pdf for $N(0, 1)$ to the right of $|ZS|$ **PLUS** the area under this pdf to the left of $-|ZS|$. Since $N(0, 1)$ is symmetric about z = 0, each of these two values is the same.

If we set a significance level α (often set to 0.05), we say the result is **statistically significant at level $\boldsymbol{\alpha}$** if the p-value from whatever statistical test we are using is $\leq \alpha$. Since the probability for the statistics we are using to have any one value is 0, we could just as well use the condition that the p-value $< \alpha$. (Note for discrete distributions (probability mass functions, where single values can have positive probability) the distinction between $<$ and $\leq$ matters, and one should use $\leq$). I'm going to use z_α (**not** $z_{1-\alpha}$) to denote the value at which the cumulative distribution function (**cdf**) for $N(0, 1)$ is $(1 - \alpha)$ (so then the probability **to the right** of z_α, i.e., the area under the pdf to the right of z_α, is α). Note the R code for z_α is

```
z_alpha <- qnorm(alpha, mean = 0, sd = 1, lower.tail = FALSE)
```

or equivalently,

```
z_alpha <- qnorm(1 - alpha, mean = 0, sd = 1, lower.tail = TRUE)
```

Then the condition for the one group (one.sample) one-sided z-test (alternative "greater") to be statistically significant is $ZS \geq z_\alpha$, and the condition for the two-sided z-test to be statistically significant is $|ZS| \geq z_{\alpha/2}$

Significance

An aside: before getting to power calculations, it is worth repeating, as done in the course, that **statistical significance** does not necessarily imply **importance** (significance as the word is commonly used). For example, if a new medicine for reducing blood pressure in fact only reduced it on average by 0.1 mm Hg and with a small variance (so precluding some subset of subjects for which the effect was distinctly greater), I rather doubt anyone would be interested since that amount would likely not be considered enough to be of much benefit. However with a large enough study (a large enough number of samples) one could obtain statistical significance. Indeed, if the true mean for the **change** in blood pressure from taking the medicine, denote it by μ_a, is not exactly equal $\mu_0 = 0$ (the null hypothesis value for this example), then if one takes n large enough (would need to be very large if μ_a is very close to μ_0), one would have enough power to be likely to get a statistically significant test result. Whether the difference between μ_a and μ_0 is "important" (of practical significance) is a judgement to be made by an expert in the subject area of the study. The **effect size** refers to a measure of the difference, which can, for example, be defined as **delta**, given by $\mu_a - \mu_0$; or **Cohen's** $d = (\mu_a - \mu_0)/\sigma$ (or for a two group test, the difference between the group means divided by the pooled standard deviation). For gene expression studies, measuring (relative) amounts of mRNA, a commonly used measure of the effect size for each gene is the *signed fold change*, given by the average amount of mRNA for the gene in group 1 divided by the average amount of mRNA for the gene in group 2 (but if this ratio is < 1, then take its reciprocal and put a minus sign in front; this makes the magnitude always at least 1 and indicates which group has the higher average). When, for example, there is a "treatment" (or "disease") group and a "control" group (not given the treatment (or healthy)); often the "treatment" (or "disease") group is taken as group 1, and the control as group 2, so genes with a positive signed fold change have a higher average expression level in the "treatment" (or "disease") group. In this setting it is also important to consider whether the largest of these two averages is large enough to have a "noticeable" biological effect, otherwise the difference may not matter.

A plot illustrating Power for a one-sided test

Let's do some plots illustrating power, first using a one-sided (alternative "greater") one group Z-test. The null hypothesis is that the population has a normal distribution with mean μ_0 and standard deviation σ. **To calulate power, we assume that a specific alternative hypothesis is true**: the population is actually a normal distribution with mean μ_a (and standard deviation σ). Note, since the normal distribution (and Student's t-distribution) is symmetric about its mean, a power calculation for the case of a one-sided z-test (or t-test) with alternative "less" is equivalent (symmetric to) the power calculation for the one-sided z-test (or t-test) with alternative "greater", so for the rest of this article, for one-sided tests, we will only discuss this case (alternative "greater"). (If we were looking at a one-sided alternative "less" test, with $\mu_a < \mu_0$, the power will be the same as the power for the alternative "greater" test with the alternative distribution mean $= \mu_0 + (\mu_0 - \mu_a)$)

As noted in [1] Brian Caffo "Statistical inference for data science" Leanpub, May 2016, if

mua = mu0 the power will be equal alpha, and if mua were < mu0 then the power for the one-sided ("greater") test will be < alpha (now I'm using variable names that will be used in the R code below). Assume the mean for the alternative hyothesis is (at least) some value denoted by mua, so then the power will be (at least) the power for when the mean equals mua. As noted in [1], what really matters is the value of delta = mua - mu0, so for calculating power we can when convenient take mu0 to be 0 and mua = delta. Below is a plot illustrating power for a one-sided one group Z-test (alternative "greater"). Note for the plot, I am using the Z-score statistic for the null and for the alternative hypotheses, not the "original" normal distributions of the null and alternative hypotheses themselves (so the distributions have been shifted by subtracting mu0 and then scaled by the standard error of the mean, sigma / sqrt(n). While perhaps not quite as intuitive as using the "original" distributions, this is the form convenient for a detailed discussion of power for t-tests.

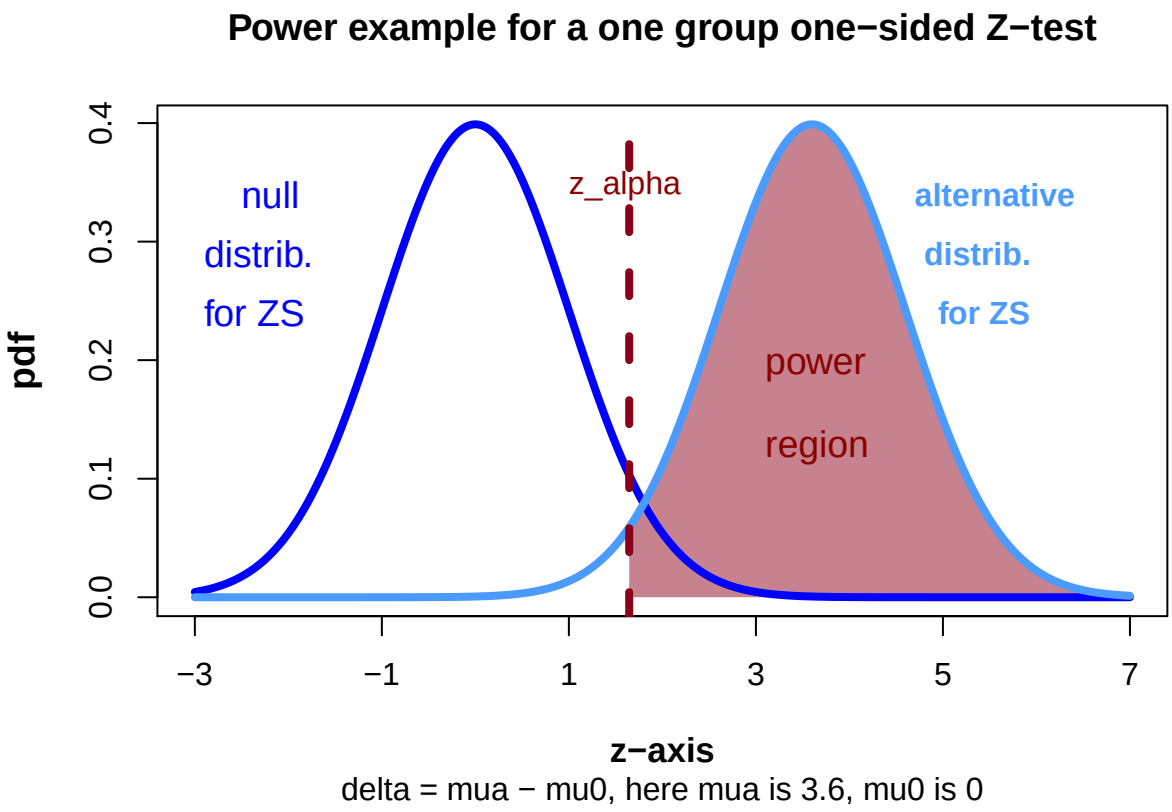
```
# set plot size
# from
# https://stackoverflow.com/questions/1279003/specify-width-and-height-of-plot
# see answer by Cybernetic May 20, 2018
# sets the options for the plot size in inches
set_plot_dimensions <- function(width_choice, height_choice) {
       options(repr.plot.width=width_choice, repr.plot.height=height_choice)
       }
set_plot_dimensions(5, 4)


# Do a plot to illustrate power for a one-sided one group Z-test
# corresponding to an individual plot produced by the R code
# in Caffo [1] "Statistical inference for data science" page 102

# Here we will shift the null and alternative distributions
# by subtracting mu0 (so in effect taking mu0 = 0, and mua = delta),
# and then we scale the distributions
# by dividing by the standard error of the mean, sigma / sqrt(n)
# This is OK since the power only depends
# on the difference delta = mua - mu0,
# not on mua and mu0 individually (the power also depends on the
# sigificance level alpha, the sample size n, and the standard
# deviation sigma of the normal distribution being sampled).
# We are then looking directly at the Z-statistic (Z-score) for the Z-test,
# for the null and for the alternative distributions


# For how to do shading under a curve see for example:
# Paul Murrell, "R Graphics", Chapman & Hall / CRC,
# Boca Raton FL 2006 page 104 - 106;
# and R-bloggers Example 9.22: shading plots and inequalities,
```

```r
# posted on March 1, 2012 by Nick Horton
# https://www.r-bloggers.com/2012/03/example-9-22-shading-plots-and-inequalities/


# set parameters, mostly the same as in Caffo [1] page 102
n <- 16
sigma <- 4
mu0 <- 0.
mua <- 3.6  # the effect size delta = mua - mu0
alpha <- 0.05
alphad2 <- alpha / 2
# will later use alphad2 to get the quantile for a two-sided test

xmin <- -3  # specify x-axis limits in plot
xmax <- 7
ymin <- 0

x <- seq(from = xmin, to = xmax, by = 0.01)

# values at which to evaluate the normal
# probability distributions we will use
# in this example

# the null distribution

# Letting m be the sample mean of n independent random samples taken from
# a normal distribution with mean mu0 and standard deviation sigma,
# the Z-score (Z statistic) ZS = (m - mu0) / (sigma / sqrt(n))
# has a standard normal distribution (mean = 0, standard deviation = 1)
# since the standard error of the mean (sigma / sqrt(n)) is the standard
# deviation of the random variable m

# get plotting values for the Z-score for the null distribution

y <- dnorm(x, mean = 0, sd = 1)  # values of ZS to plot
# y contains values of the probability density function (pdf) of a normal
# distribution with the given mean mean and standard deviation,
# evaluated at each of the x points

# plot the probability density function (pdf) as a line/curve

plot(x, y, lty = "blank", lwd = 4, , type = "l", col = "blue", xaxt = "n",
     xlab = ' ',  ylab = ' ')
# lty = 1 would draw a solid line/curve, but will draw the pdf curves below
```

```r
# after shading in the "power region"
# xaxt = "n" suppresses the x axis labeling so can specify it as we want

axis(1, at = seq(-3, 7, by = 2)) # requests tic marks and labels at the
# integers -3, -1, ... , 7

# plot a vertical line at z_alpha equal to the
# (1-alpha) quantile of this null distribution,
# i.e., where the probability to the right of z_alpha for the null
# distribution, which here is N(0, 1), is alpha.
# This is the relevant quantile for getting the power for a
# one-sided one group Z-test, alternative "greater" where the
# null hypothesis is: the mean mu is equal mu0

# The power for this example is the fraction of time that the Z-score ZS for
# means m from samples from the alternative distribution (mean = mua) satisfies
# ZS >= z_alpha (so then the p-value is <= alpha).
# Here m is the mean of n independent random samples from the
# alternative distribution (normal with mean mua and standard deviation sigma).

z_alpha <- qnorm(alpha, mean = 0, sd = 1, lower.tail = FALSE)
# plot a vertical line at z = z_alpha
# abline(v = z_alpha, lwd = 4, lty = 2,  col = "red4")  # vertical dashed line
# below will plot the line after shading in the "power region"

# Let's plot a case where mua > mu0 and graphically display the "power region"

# the alternative distribution

# Now let m be the sample mean of n independent random samples taken from
# a normal distribution with mean mua and standard deviation sigma; we then
# have ZS = (m - mu0) / (sigma / sqrt(n))  is the statistic for the one group
# Z-test for the null hypothesis that the mean is mu0 (which we are taking as 0).
# So then ZS has a normal distribution with mean = (mua - mu0) / (sigma / sqrt(n))
# and standard deviation = 1
# (since the standard error of the mean, sigma / sqrt(n), is the standard
# deviation of the random variable m (the mean of n independent random samples
# from the alternative distribution))
# If this ZS is >= z_alpha then the p-value for the one group
# one-sided (alternative "greater") Z-test will be <= alpha. Hence the power is
# the area under the pdf of ZS for means m from the alternative distribution to
# the right of z_alpha (which is the shaded region in the plot below).

# get points to plot for the alternative distribution
```

```r
ya <- dnorm(x, mean = mua / (sigma / sqrt(n)), sd = 1)  # here mu0 is 0
# ya has values from the probability density function (pdf) of a normal
# distribution with the given mean and standard deviation,
# evaluated at each of the x points

# plot ya
# lines(x, ya, lty = "blank", lwd = 4, col = "cyan", type = "l")
# will plot the pdf after do shading to indicate the "power region"

# define some custom colors

# Add a bit of blue to red4 and get partially transparent version,
# and some red to cyan
# see https://www.dataanalytics.org.uk/make-transparent-colors-in-r/
# Trying to pick colors still visible under some forms of color blindness,
# see for example
# https://www.color-blindness.com/coblis-color-blindness-simulator/

# get rgb values for a color name
# col2rgb("red4")  # 139, 0, 0
# fully transparent is alpha = 0, solid is alpha = 255
# define some custom colors
red4.with.some.blue <- rgb(139, 0, 25, max = 255, alpha = 255, names = "red4b")
red4.with.some.blue.transparent <- rgb(139, 0, 25, max = 255,
                                       alpha = 125, names = "red4bt")
cyan.with.some.red <- rgb(75, 155, 255, max = 255, alpha = 255, names = "cyanwr")

# shade in the "power region", i.e., under the part of the pdf
# for ZS for the alternative hypothesis for z >= z_alpha

# use the polygon function
xpower <- seq(from = z_alpha, to = xmax, by = 0.01)
ypower <- dnorm(xpower, mean = mua / (sigma / sqrt(n)), sd = 1)
xpoly <- c(z_alpha, xpower, xmax)
ypoly <- c(ymin, ypower, ymin)
polygon(xpoly, ypoly, col = red4.with.some.blue.transparent, border = NA)


# plot the pdf's to have those curves visible, and also the vertical line
lines(x, y, lty = 1, lwd = 4, col = "blue", type = "l")
lines(x, ya, lty = 1, lwd = 4, col = cyan.with.some.red, type = "l")

# vertical line
abline(v = z_alpha, lwd = 4, lty = 2,  col = red4.with.some.blue)
```

```r
# add a plot title
title(main = "Power example for a one group one-sided Z-test", cex.lab = 1.1)

# text in plot
text(-2.5, 0.34, "null", cex = 1.2, col = "blue", adj = c(0., 0.5))
text(-2.9, 0.29, "distrib.", cex = 1.2, col = "blue", adj = c(0., 0.5))
text(-2.9, 0.24, "for ZS", cex = 1.2, col = "blue", adj = c(0., 0.5))

text(4.7, 0.34, "alternative", cex = 1.0, col = cyan.with.some.red,
     adj = c(0, 0.5), font = 2)
text(4.8, 0.29, "distrib.", cex = 1.0, col = cyan.with.some.red,
     adj = c(0, 0.5), font = 2)
text(4.95, 0.24, "for ZS", cex = 1.0, col = cyan.with.some.red,
     adj = c(0, 0.5), font = 2)

text(1, 0.35, "z_alpha", cex = 1.0, col = "red4", adj = c(0., 0.5))

text(3.10, 0.20, "power", cex = 1.2, col = "red4", adj = c(0., 0.5))
text(3.10, 0.13, "region", cex = 1.2, col = "red4", adj = c(0., 0.5))

# axis labels
title(main = NULL, sub = "delta = mua - mu0, here mua is 3.6, mu0 is 0",
      xlab = "z-axis", ylab = NULL, line = NA, outer = FALSE, font.lab = 2,
      cex.lab = 1.1, col.lab = "black")

title(main = NULL, sub = NULL, xlab = NULL, ylab = "pdf",
      line = NA, outer = FALSE, font.lab = 2,
      cex.lab = 1.2, col.lab = "black")
```



Power for a two-sided test

Now let's do a corresponding plot for the two-sided Z-test. For calculating the power in this case, what matters is the absolute value of delta, so we can just as well take mua $\geq$ mu0. For the two-sided test, the relevant value of z is z_alphad2, such that the area under the pdf for $N(0,1)$ to the right of z_alphad2 is $\alpha/2$. Then the "**strict**" power (when the alternative hypothesis mean is mua) is the fraction of time the Z-score ZS in equation (1) (for n independent random samples from a normal distribution with mean mua and standard deviation sigma) is greater than z_alphad2 **PLUS** $\mathcal{B}$ = the fraction of time ZS is **below** -z_alphad2. For cases where there is any appreciable power (i.e., mua is "appreciably" > mu0), the latter term (the fraction of time ZS is below -z_alphad2) will be relatively small (since this will be "way out on the left tail" of the pdf for the alternative distribution of ZS). Since we are assuming mua is at least mu0, such values also have the wrong sign (wrong direction) so arguably are not what one would want to count toward power for detecting a significant change (p-value below alpha) and in the correct direction. This is noted in page 103 of [1]. Many R functions that calculate power have an argument called *strict* whose default is usually FALSE (don't include the contribution corresponding to $\mathcal{B}$ in the power for a two-sided test); if one wants to include it then one sets the argument strict to be TRUE. Note if one does the power calculation for a two-sided test and includes $\mathcal{B}$ (the strict = TRUE case) then when delta = (mua - mu0) is 0, one gets alpha (as expected) but if

one takes strict = FALSE (only including the contribution to power from the "correct side", then when delta is 0, one will get alpha/2. In cases of practical interest, as noted above, the contribution to the power for Z-tests and t-tests from $\mathcal{B}$ will be inconsequential.

Below is a plot for a two-sided one group Z-test, here I have increased alpha to 0.1 and decreased mua in order to have $\mathcal{B}$ be "visible" (but this is a case where the power is much too small to be useful other than for doing a conceptual graphic). Even with this small value for mua, the power region on the left (to the left of -z_alphad2 is rather small in the plot.

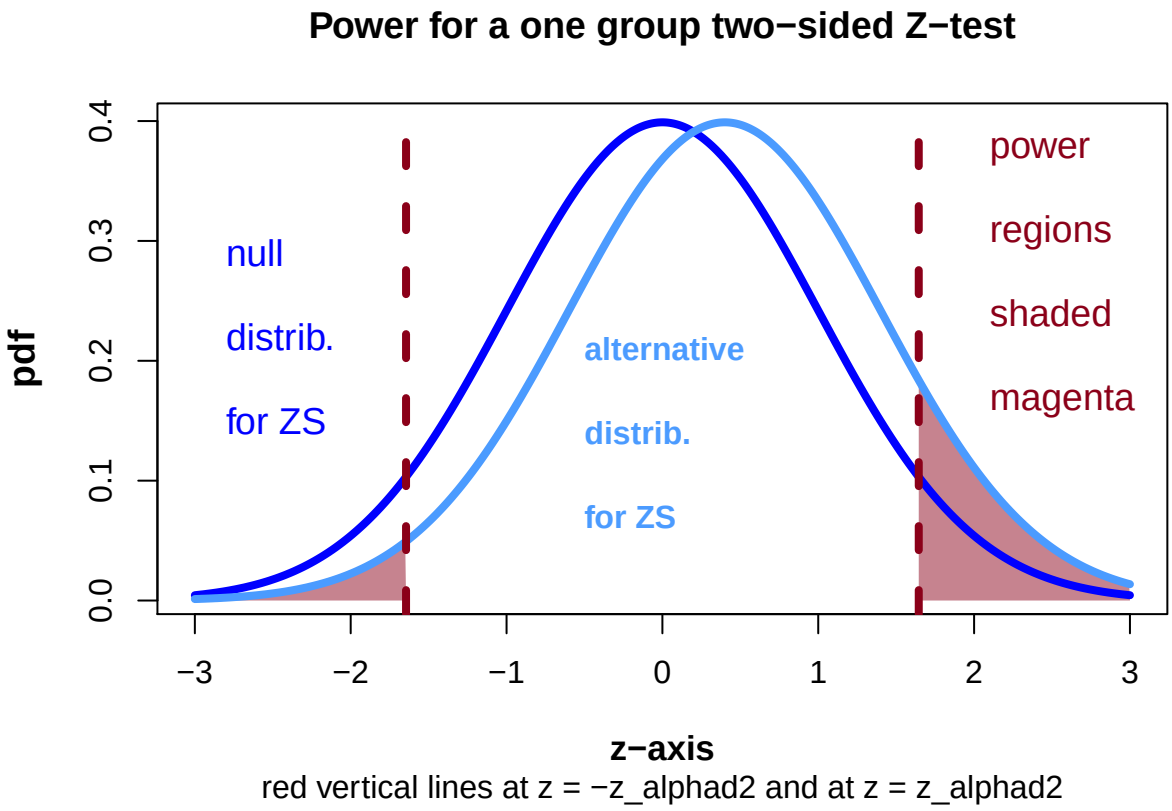
```
# Do a plot to illustrate power for a two-sided one group Z-test

# Here we will scale the distributions by dividing by
# the standard error of the mean and we are taking mu0 equal 0;

# set parameters, mostly the same as in Caffo [1] page 102
# but pick values so the "left tail contribution" to the power
# is visible in the plot
n <- 16
sigma <- 4
mu0 <- 0.
mua <- 0.4  # the effect size delta = mua - mu0 which here
# gives a power much too small to be of interest
# except for doing this conceptual plot
alpha <- 0.1
alphad2 <- alpha / 2

xmin <- -3  # specify x-axis limits in plot
xmax <- 3
ymin <- 0

x <- seq(from = xmin, to = xmax, by = 0.01)

# values at which to evaluate the normal
# probability distributions we will use
# in this example

# the null distribution

# Letting m be the sample mean of n independent random samples taken from
# a normal distribution with mean mu0 and standard deviation sigma,
# the Z-score (Z statistic) ZS = (m - mu0) / (sigma / sqrt(n))
# has a standard normal distribution (mean = 0, standard deviation = 1)

# get plotting values for the Z-score for the null distribution
```

```r
y <- dnorm(x, mean = 0, sd = 1)  # values of ZS to plot
# y contains values of the probability density function (pdf) of a
# standard normal distribution

# plot the probability density function (pdf) as a line/curve

plot(x, y, lty = "blank", lwd = 4, , type = "l", col = "blue", xaxt = "n",
     xlab = ' ',  ylab = ' ')
# lty = 1 would draw a solid line/curve, but will draw the pdf curves below
# after shading in the "power region"
# xaxt = "n" suppresses the x axis labeling so can specify it as we want

axis(1, at = seq(-3, 3, by = 1)) # requests tic marks and labels at the
# integers -3, -2, ... , 3

# plot a vertical line at z_alphad2 equal to the
# (1 - alpha/2) quantile of this null distribution,
# i.e., where the probability to the right of z_alphad2 for the null
# distribution, which here is N(0, 1), is alpha/2.
# This is the relevant quantile for getting the power for a
# two-sided one group Z-test, where the
# null hypothesis is: the mean mu is equal mu0 (here 0)

# The strict power here is the fraction of time that ZS is >= z_alphad2
# PLUS the fraction of time that ZS is <= -z_alphad2
# (so then the two-sided p-value is <= alpha) when
# m is the mean of n independent random samples from the
# alternative distribution (normal with mean mua and standard deviation sigma).

z_alphad2 <- qnorm(alphad2, mean = 0, sd = 1, lower.tail = FALSE)
# plot a vertical line at z = z_alphad2 and at z = -z_alphad2
# abline(v = z_alphad2, lwd = 4, lty = 2,  col = "red4")  # vertical dashed line
# abline(v = -z_alphad2, lwd = 4, lty = 2,  col = "red4")  # vertical dashed line
# below will plot the lines after shading in the "power region"

# graphically display the strict "power region" (including the "left side")

# the alternative distribution

# Now let m be the sample mean of n independent random samples taken from
# a normal distribution with mean mua and standard deviation sigma; we then
# have ZS = (m - mu0) / (sigma / sqrt(n))  is the statistic for the one group
# Z-test for the null hypothesis that the mean is mu0 (which we are taking as 0).
# So then ZS has a normal distribution with mean = (mua - mu0) / (sigma / sqrt(n))
```

```
# and standard deviation = 1

# If this ZS is >= z_alphad2 or <= -z_alphad2
# then the p-value for the one group
# two-sided Z-test will be <= alpha. Hence the "strict" power is
# the area under the pdf of ZS for the alternative distribution
# to the right of z_alphad2 PLUS the area under the alternative distribution pdf
# to the left of -z_alphad2
# (which is the shaded region in the plot below).

# get points to plot for the alternative distribution

ya <- dnorm(x, mean = mua / (sigma / sqrt(n)), sd = 1)  # mu0 is 0
# ya has values from the probability density function (pdf) of a normal
# distribution with the given mean and standard deviation,
# evaluated at each of the x points

# plot ya
# lines(x, ya, lty = "blank", lwd = 4, col = "cyan", type = "l")
# will plot the pdf after do shading to indicate the "power region"

# define some custom colors

# Add a bit of blue to red4 and get partially transparent version,
# and some red to cyan
# see https://www.dataanalytics.org.uk/make-transparent-colors-in-r/
# Trying to pick colors still visible under some forms of color blindness

# get rgb values for a color name
# col2rgb("red4")  # 139, 0, 0
# fully transparent is 0, solid is 255
# define some custom colors
red4.with.some.blue <- rgb(139, 0, 25, max = 255, alpha = 255, names = "red4b")
red4.with.some.blue.transparent <- rgb(139, 0, 25, max = 255,
                                       alpha = 125, names = "red4bt")
cyan.with.some.red <- rgb(75, 155, 255, max = 255, alpha = 255, names = "cyanwr")

# shade in the "power region", i.e., under the part of the pdf
# for ZS for the alternative hypothesis for z >= z_alphad2,

# use the polygon function
xpower <- seq(from = z_alphad2, to = xmax, by = 0.01)
ypower <- dnorm(xpower, mean = mua / (sigma / sqrt(n)), sd = 1)
xpoly <- c(z_alphad2, xpower, xmax)
```

```
ypoly <- c(ymin, ypower, ymin)
polygon(xpoly, ypoly, col = red4.with.some.blue.transparent, border = NA)

# AND shade in the "left power region", i.e., under the part of the pdf
# for ZS for the alternative hypothesis for z <= -z_alphad2

# use the polygon function
xpower <- seq(from = xmin, to = -z_alphad2, by = 0.01)
ypower <- dnorm(xpower, mean = mua / (sigma / sqrt(n)), sd = 1)
xpoly <- c(xmin, xpower, -z_alphad2)
ypoly <- c(ymin, ypower, ymin)
polygon(xpoly, ypoly, col = red4.with.some.blue.transparent, border = NA)

# plot the pdf's to have those curves visible, and also the vertical lines
lines(x, y, lty = 1, lwd = 4, col = "blue", type = "l")
lines(x, ya, lty = 1, lwd = 4, col = cyan.with.some.red, type = "l")

# vertical lines
abline(v = z_alphad2, lwd = 4, lty = 2,  col = red4.with.some.blue)
abline(v = -z_alphad2, lwd = 4, lty = 2,  col = red4.with.some.blue)

# add a plot title
title(main = "Power for a one group two-sided Z-test", cex.lab = 1.1)

# text in plot
text(-2.8, 0.29, "null", cex = 1.2, col = "blue", adj = c(0., 0.5))
text(-2.8, 0.22, "distrib.", cex = 1.2, col = "blue", adj = c(0., 0.5))
text(-2.8, 0.15, "for ZS", cex = 1.2, col = "blue", adj = c(0., 0.5))

text(-0.5, 0.21, "alternative", cex = 1.0, col = cyan.with.some.red,
     adj = c(0, 0.5), font = 2)
text(-0.5, 0.14, "distrib.", cex = 1.0, col = cyan.with.some.red,
     adj = c(0, 0.5), font = 2)
text(-0.5, 0.07, "for ZS", cex = 1.0, col = cyan.with.some.red,
     adj = c(0, 0.5), font = 2)


text(2.10, 0.38, "power", cex = 1.2, col = red4.with.some.blue, adj = c(0., 0.5))
text(2.10, 0.31, "regions", cex = 1.2, col = red4.with.some.blue, adj = c(0., 0.5))
text(2.10, 0.24, "shaded", cex = 1.2, col = red4.with.some.blue, adj = c(0., 0.5))
text(2.10, 0.17, "magenta", cex = 1.2, col = red4.with.some.blue, adj = c(0., 0.5))

# axis labels
title(main = NULL, sub = "red vertical lines at z = -z_alphad2 and at z = z_alphad2",
```

```
      xlab = "z-axis", ylab = NULL, line = NA, outer = FALSE, font.lab = 2,
      cex.lab = 1.1, col.lab = "black")

title(main = NULL, sub = NULL, xlab = NULL, ylab = "pdf",
      line = NA, outer = FALSE, font.lab = 2,
      cex.lab = 1.2, col.lab = "black")
```



red vertical lines at z = −z_alphad2 and at z = z_alphad2

A couple power calculations for Z-tests

Let's do a power calculation for the one group one-sided Z-test given in page 101 of [1], using the framework we have set up above. In [1], mua is 32 and mu0 is 30; so delta is mua - mu0 = 2, and so we will do the following:

```
n <- 16
sigma <- 4
mua <- 2
mu0 <- 0
alpha <- 0.05
z_alpha = qnorm(alpha, mean = 0, sd = 1, lower.tail = FALSE)
```

```
# get the power for a one-sided one group (one.sample) Z-test
# for the values above

# get the area under the pdf for the Z-score for the alternative distribution
# to the right of z_alpha
pnorm(z_alpha, mean = (mua - mu0) / (sigma / sqrt(n)), sd = 1, lower.tail = FALSE)
# [1] 0.63876
# which agrees with [1] to the number of significant digits printed in [1]
```

And now a power calculation for a one group two-sided Z-test, where I'm increasing alpha to be 0.10, so the power (for strict = FALSE) will equal the value just calculated above.

```
n <- 16
sigma <- 4
mua <- 2
mu0 <- 0

alpha <- 0.10  # so the contribution to the power
# from the "correct side" will be the same as above
alphad2 <- alpha / 2  # here 0.05
z_alphad2 = qnorm(alphad2, mean = 0, sd = 1, lower.tail = FALSE)

correct.side.power <- pnorm(z_alphad2, mean = (mua - mu0) / (sigma / sqrt(n)),
                            sd = 1, lower.tail = FALSE)
left.side.power <- pnorm(-z_alphad2, mean = (mua - mu0) / (sigma / sqrt(n)),
                         sd = 1, lower.tail = TRUE)
correct.side.power
# [1] 0.63876
# the same as above since I took alpha = 0.10 for this calculation

left.side.power
# [1] 0.000133772  # negligible

strict.power <- correct.side.power + left.side.power
strict.power
# [1] 0.6388938
```

Except in unusual circumstances (or for precisely checking a calculation), one wouldn't report power to more than 2 or 3 significant digits, so in general, the "left side" contribution to the power in a two-sided Z-test or t-test test won't really matter when the power is large enough to be of interest (in which case the contribution to the power from the "left tail" will be quite small). **Henceforth for the calculation of power for two-sided Z-tests and two-sided t-tests, I'll only consider the strict = FALSE case** (only consider contributions to power from the "correct" side).

power for t-tests and why the noncentral t-distribution is needed

The calculation of power for one-sided and two-sided one group t-tests follows the same conceptual form as above for Z-tests. There is one significant change which is a direct consequence of using the sample standard deviation s as an approximation to the population standard deviation σ.

The T-score TS for the null distribution for a one-group t-test (mean = μ_0) is given by

$$TS = \frac{(m - \mu_0)}{s/\sqrt{n}} \qquad \text{equation (2)}$$

where m is the mean of n random samples $x_1, ..., x_n$ from the null distribution $N(\mu_0, \sigma)$. This TS then has the Student's t-distribution with (n-1) degrees of freedom. For one group Z-tests, the distribution of the Z-score for the alternative distribution also had a normal distribution with just a shifted mean. However for t-tests, dealing with the shift requires a **noncentral** t-distribution. Here is why: The T-score TS for samples from the alternative distribution (mean = μ_a, and as above, for calculating power, we can take $\mu_a \geq \mu_0$), is given by

$$TS = \frac{(m - \mu_0)}{s/\sqrt{n}} = \frac{[(m - \mu_a) + (\mu_a - \mu_0)]}{s/\sqrt{n}} \qquad \text{equation (3)}$$

If we tried to proceed as for Z-tests, we would look at the probability for TS for samples from the alternative distribution being as large or larger than some critical value t_α, (or $t_{\alpha/2}$ for a two-sided test) ([1] uses the notation $t_{1-\alpha,\ n-1}$). The dependence of t_α and $t_{\alpha/2}$ on the degrees of freedom is to be understood here. The probability for $TS \geq t_\alpha$ is, from equation (3), the same as for

$$\frac{(m - \mu_a)}{s/\sqrt{n}} \geq t_\alpha - \frac{(\mu_a - \mu_0)}{s/\sqrt{n}} \qquad \text{equation (4)}$$

The left side of equation (4) is nicely governed by the Student's t-distribution, but, alas, the right side contains s, the sample standard deviation, which is itself a random variable. In the case of Z-tests we had a known **constant** value for the population standard deviation σ so this issue did not arise in that case.

Looking back at equation (3), where the samples are from the alternative distribution with mean μ_a (**not** μ_0) the quantity

$$TS = \frac{(m - \mu_0)}{s/\sqrt{n}}$$

has the distribution of a **noncentral** t-distribution with the **non-centrality parameter ncp** given by (see for example the R help for the **dt** function):

$$\text{ncp} = \frac{(\mu_a - \mu_0)}{\sigma/\sqrt{n}} \qquad \text{equation (5)}$$

Calculating power for a t-test using the noncentral t-distribution

Since R kindly provides functions (dt, pt, qt, rt) to do calculations with the t-distribution, including the case for noncentral t-distributions, it is then easy to calculate the power for a one group t-test (we'll deal with two group t-tests shortly). The power for a one group one-sided t-test is the area under pdf for the noncentral t-distribution (with non-centrality parameter ncp given in equation (5)) that is to the right of t_α. Recall, t_α is given by `qt(alpha, df = n-1, lower.tail = FALSE)` If ncp is not specified in a call to dt, pt, qt, or rt, one gets results for the "common" Student's t-distribution (ncp = 0). For the two-sided one group t-test, the power (with strict = FALSE, i.e., not including the contribution to the power from the left ("incorrect") tail) is the area under the pdf for the noncentral t-distribution with non-centrality parameter ncp given by equation (5) that is to the right of $t_{\alpha/2}$. Note the dependence of the power on delta = (mua - mu0), sigma, and n is accounted for in the value of ncp, and the dependence on α is accounted for in the critical value t_α or $t_{\alpha/2}$ (depending on whether the test is one or two sided, and the critical value also depends on the number of degrees of freedom (n-1)). If we use a direct calculation using R's **pt** function, or use the **power.t.test** we should (we better) get the same result; some sample calculations are given below.

An aside on one-sided tests

The following is my opinion on using one-sided tests. A classic example of when a one-sided test would be appropriate is the situation where an engineer has come up with a potentially improved new way to manufacture light bulbs, and suppose the cost of the new process is the same, and the appearance of the light bulbs and the characteristics of the light they give out is unchanged, but the time the new light bulbs keep working might well be larger. So then one might do an experiment measuring the lifetime of some number of light bulbs manufactured using the new process and ask whether their average lifetime mua is greater than the value mu0 equal the average lifetime of light bulbs manufactured using the current process. And let's assume the standard deviation of the new lifetimes is similar to the previous process, so one doesn't need to be concerned with the situation that the average lifetime from the new process is larger, but enough of the light bulbs from the new process fail so soon that this would be unacceptable (so the main question is whether or not mua is *significantly* greater than mu0, in both the statistical sense and the "greater than by enough to matter" sense). In this circumstance, if the average lifetime from the new process was smaller than mu0, we would (or the engineer would) "pack up and go home", i.e., there is really only interest in the case where mua > mu0 (and by enough so changing the manufacturing process is worthwhile). Then one would be justified in using a one-sided test. In situations where one doesn't have interest in only one particular direction (whether mua > mu0, or mua < mu0), i.e., one would pursue or "publish" a statistically significant result in either direction (that has a "sufficient" effect size), then one really must use the two-sided test. Otherwise one risks someone thinking one just used a one-sided test to artificially get a smaller (here, for Z-tests and t-tests, by half) p-value.

Some example one group t-test power calculations, see page 104 of [1]

We will calculate directly using the noncentral t-distribution and then check with the power.t.test function.

```
# one-sided one group power, and two-sided one group power with strict = FALSE
n <- 16
sigma <- 4
mua <- 0
mu0 <- 0  # delta = mua - mu0 = 0
############ so should get the one-sided power = alpha
############ and the two-sided power = alpha/2 (since using strict = FALSE)

alpha <- 0.05
t_alpha = qt(alpha, df = n-1, lower.tail = FALSE)

# use the noncentral t-disribution to get the one-sided ("greater") power
ncp.value <- (mua - mu0) /  (sigma / sqrt(n))
pt(t_alpha, df = n-1, ncp = ncp.value, lower.tail = FALSE)  # the one-sided power
### [1] 0.05   # as expected for delta = 0


# the two-sided power (for strict = FALSE, i.e., only include the "correct tail")
alphad2 <- alpha / 2
t_alphad2 = qt(alpha/2, df = n-1, lower.tail = FALSE)
# use the noncentral t-disribution to get the two-sided power, strict = FALSE
pt(t_alphad2, df = n-1, ncp = ncp.value, lower.tail = FALSE)  # the two-sided power
### [1] 0.025  # as expected for delta = 0


# check using the power.t.test function
# one-sided ("greater") power
power.t.test(n = n, delta = mua - mu0, sd = sigma, sig.level = alpha, power = NULL,
             type = "one.sample", alternative = "one.sided", strict = FALSE)$power
### [1] 0.05   # as expected for delta = 0

# two-sided power
power.t.test(n = n, delta = mua - mu0, sd = sigma, sig.level = alpha, power = NULL,
             type = "one.sample", alternative = "two.sided", strict = FALSE)$power
### [1] 0.025  # as expected for delta = 0


############################################################
```

```
# repeat this calculation but now with delta = 2 (an example in page 104 of [1])

mua = 2  # the rest of the parameters alpha, n, and sigma stay the same
# delta is now 2

# use the noncentral t-disribution to get the one-sided ("greater") power
ncp.value <- (mua - mu0) /  (sigma / sqrt(n))
pt(t_alpha, df = n-1, ncp = ncp.value, lower.tail = FALSE)  # the one-sided power
### [1] 0.6040329


# the two-sided power (for strict = FALSE, i.e., only include the "correct tail")
# use the noncentral t-disribution to get the two-sided power, strict = FALSE
pt(t_alphad2, df = n-1, ncp = ncp.value, lower.tail = FALSE)  # the two-sided power
### [1] 0.4648089

# check using the power.t.test function
# one-sided ("greater") power
power.t.test(n = n, delta = mua - mu0, sd = sigma, sig.level = alpha, power = NULL,
             type = "one.sample", alternative = "one.sided", strict = FALSE)$power
### [1] 0.6040329

# two-sided power
power.t.test(n = n, delta = mua - mu0, sd = sigma, sig.level = alpha, power = NULL,
             type = "one.sample", alternative = "two.sided", strict = FALSE)$power
### [1] 0.4648089
```

Solving for the number of samples

As noted in [1], the power for a one group Z-test and for a one group t-test equals a function, call it $\mathcal{F}$ of $\alpha, n,$ delta $= \mu_a - \mu_0,$ and σ. Hence, if for example, we are given a value of the power and want to solve for the number of samples n needed to obtain that power, we need to solve $\mathrm{F}(n) = 0$ where F is defined by

$$\mathrm{F}(n) = \text{power } - \mathcal{F}(\alpha, n, \text{delta}, \sigma) \qquad \text{equation (6)}$$

(and values for the other variables in equation (6) have been given). Note the function $\mathrm{F}(n)$ is actually well defined for non-integer values of n. $\mathrm{F}(n) = 0$ is then one (quite non-linear) equation in the one unknown n, and there are efficient numerical methods for finding solution(s) when they exist (e.g., the R function **uniroot**). One can think of this as plotting a graph of $\mathrm{F}(n)$ as a function of n (given values for the other variables) and find where the graph of this function crosses 0. In general the value of n that solves $\mathrm{F}(n) = 0$ is not an integer, so for a power calculation for the number of samples n, one would "round up" to the smallest integer greater than or equal the solution.

Now let's consider power for a two group t-test

For a two group t-test, one has the choice of using the equal variance form (the Student's t-test), or the form that explicitly deals with unequal variances in the two populations the samples are from (called the Welch t-test, also called the Smith-Satterthwaite t-test). Several references (I'm not saying this is a complete list of studies on this) suggest always using the unequal variance form (Welch) of the t-test (unless one has definitive information that the variances for the 2 populations are the same), because, in general, when the population variances are in fact equal, the Welch form of the t-test gives, on average, results pretty close to the equal variance form, and the Welch form is the proper choice when the variances for the 2 groups (the 2 populations that the samples are from) are not equal. (Also, it's not a coincidence that the default for R's t.test function is to use the unequal variance form.) The references also indicate using Welch is in general better than doing a test for equal variances and using that result to decide whether to use the equal variance (Student's t-test) form or the unequal variance form (Welch).

References for this:

J.S. Milton and J.C. Arnold, Introduction to Probability and Statistics, 3rd Edition. McGraw-Hill, Boston, 1995, see page 353

M. Delacre, D. Lakens and C. Leys, Why Psychologists Should by Default Use Welch's t-test Instead of Student's t-test, International Review of Social Psychology, (30)1, 92-101

A. F. Hayes and L. Cai, Further evaluating the conditional decision rule for comparing two independent means, British Journal of Mathematical and Statistical Psychology (2007), 60, 217–244

I am thus only going to discuss power for the Welch form of the t-test for comparing means of two groups. As can be found in most any statistics book (see page 83 of [1]), the formulas for the Welch t-test are the following, given in the form of R code (note in particular dfW (the Welch estimate for the degrees of freedom; and standard.errorW (the Welch estimate for the standard error of sampled values for the difference of the means of the two groups); to be used below in calculating power for the Welch t-test):

```
# Given independent random samples x and y from two
# normal distributions Nx and Ny whose means are mux and muy,
# carry out the Welch (unequal variance) form of the t-test.

# The null hypothesis here is that the difference of the
# means, mux - muy, is 0

nx  <- length(x)
ny  <- length(y)

nxm1  <- nx - 1
nym1  <- ny - 1
```

```
# sample means
meanx  <- mean(x)
meany  <- mean(y)

 sample standard deviations and variances
sx  <- sd(x)
sy  <- sd(y)
vx  <- sx^2
vy  <- sy^2

# Welch (Smith-Satterthwaite) procedure for number of degrees of freedom dfW

dfW  <- (vx/nx + vy/ny)^2 / (  (vx/nx)^2 / nxm1  +  (vy/ny)^2 / nym1 )
# Welch degrees of freedom for a two group t-test
cat("        ", "\n")
cat("Welch df  ", dfW, "\n")

# get the Welch estimate for the standard error for the
# difference of the means
standard.errorW <- sqrt(vx/nx + vy/ny)  # Welch standard error
cat("        ", "\n")
cat("Welch standard error   ", standard.errorW , "\n")

# the null hypothesis is that the difference of the means is 0
tvalue  <- (meanx - meany) / sqrt(vx/nx + vy/ny)
#### When the null hypothesis is true, this t-value
# is APPROXIMATELY distributed as a t-distribution with dfW degrees of freedom
cat("        ", "\n")
cat("tvalue  ", tvalue, "\n")

# two-sided Welch t-test
pvalue  <- 2 * pt(abs(tvalue), dfW, lower.tail = FALSE)
cat("        ", "\n")
cat("two-sided p-value   ", pvalue, "\n")

# check using t.test
cat("        ", "\n")
cat("results for two-sided Welch t-test from t.test")
welch.t.test.result <- t.test(x, y, alternative = "two.sided")
welch.t.test.result
```

For the Welch t-test, the value of the Welch form of the T-score is

TS = ((meanx - meany) - d0) / sqrt(vx/nx + vy/ny) equation (7)

where meanx and meany are from independent random samples from normal distributions

Nx and Ny of size nx and ny, respectively, and vx and vy are the squares of the respective sample standard deviations. The Welch estimate for the degrees of freedom is given by dfW above. Let the null hypothesis be that the difference of the population means for Nx and Ny is d0. When the null hypothesis is valid, TS defined in equation (7) has **approximately** a t-distribution with dfW degrees of freedom. If the alternative hypothesis is taken to be that the difference of the means is actually some value da (which without loss of generality we can take as >= d0), then as in the cases above, what matters is delta = (da - d0), and we can use the noncentral t-distribution to calculate the power similarly to the way it was just done above, but now the number of degrees of freedom is taken to be (estimated by) dfW, and the non-centrality parameter ncp is given by

ncp = (da - d0) / sqrt(Vx/nx + Vy/ny) equation (8)

where for equation (8) sigmax and sigmay are the standard deviations of the populations Nx and Ny respectively, and Vx = sigmax^2 and Vy = sigmay^2 are the variances. Values for power for the Welch t-test, given values for delta = (da - d0), nx, ny, sigmax, sigmay, and the significance level alpha, can be obtained using the noncentral t-distribution with dfW degrees of freedom and non-centrality parameter ncp as given in equation (8) via the pt function or a function from an appropriate R package.

Here are some sample calculations of power for the Welch t-test.

```
# R version 3.6.0 (2019-04-26) -- "Planting of a Tree"

# set values for a Welch t-test power calculation

nx <- 35
ny <- 25
delta <- 2.6  # da - d0
alpha <- 0.05
sigmax <- 4
sigmay <- 2

##########  code a function that calculates power for the Welch t-test
##########  using the noncentral t-distribution

power_for_Welch_via_noncentral_t_distr <-
          function(nx, ny, delta, sigmax, sigmay, alpha) {

# Call by:
# power_for_Welch_via_noncentral_t_distr(nx, ny, delta, sigmax, sigmay, alpha)

# Calculates both the one-sided and two-sided power for the Welch t-test
# This calculates the strict = FALSE power for the two-sided test, i.e., does not
#      include the contribution from the "wrong side"
# Returns a data frame containing the  one-sided and two-sided power and the
```

```
#          input values,
#          and also the power from the wrong side that would be included in
#          the two-sided power if strict was TRUE,
#          and the Welch degrees of freedom dfW, and the
#          non-centrality parameter ncp

# The two groups are called x and y (whose populations are assumed to have
#     normal distributions Nx and Ny)
# The number of samples from each group are nx and ny
# The population standard deviations for each group are sigmax and sigmay
# alpha is the significance level

# delta is (da - d0) where the null hypothesis is that the difference of
# the means (mean of Nx - mean of Ny) equals d0
# and the specific alternative hypothesis (for which the power is being computed)
# is that the difference of the means is da
# (and without loss of generality
# we are taking delta to be non-negative)

delta <- abs(delta)

vx <- sigmax^2
vy <- sigmay^2
ncp_value <- delta / sqrt(vx/nx + vy/ny)
# the non-centrality parameter, see equation (8)

nxm1 <- nx - 1
nym1 <- ny - 1

dfW = (vx/nx + vy/ny)^2 / (  (vx/nx)^2 / nxm1  +  (vy/ny)^2 / nym1 )
# the Welch number of degrees of freedom
# see the R code for the Welch t-test above

t_alpha <- qt(alpha, df = dfW, lower.tail = FALSE)
alphad2 <- alpha / 2
t_alphad2 <- qt(alphad2, df = dfW, lower.tail = FALSE)

power_two_sided_strict_FALSE <-
      pt(t_alphad2, df = dfW, ncp = ncp_value, lower.tail = FALSE)
# the strict = FALSE power

wrong_side_power <- pt(-t_alphad2, df = dfW, ncp = ncp_value, lower.tail = TRUE)

power_one_sided <- pt(t_alpha, df = dfW, ncp = ncp_value, lower.tail = FALSE)
```

```
# define the data frame to be returned
datafr <- data.frame(power_two_sided_strict_FALSE = power_two_sided_strict_FALSE,
             power_one_sided = power_one_sided, wrong_side_power = wrong_side_power,
             nx = nx, ny = ny, delta = delta, sigmax = sigmax, sigmay = sigmay,
             alpha = alpha, dfW = dfW, ncp = ncp_value)
return(datafr)
}

######################
######################


# call this function with the parameter values above
power_for_Welch_via_noncentral_t_distr(nx, ny, delta, sigmax, sigmay, alpha)
```

```
##   power_two_sided_strict_FALSE power_one_sided wrong_side_power nx ny delta
## 1                    0.9012841       0.9475901     9.378596e-08 35 25   2.6
##   sigmax sigmay alpha     dfW      ncp
## 1      4      2  0.05 52.8017 3.309638
```

test our power_for_Welch_via_noncentral_t_distr function using the power_t_test function from the MESS package

It is always a good idea to test a function by comparing its results against known values or independent calculations. Here we have the advantage that there are functions in packages available from CRAN (The Comprehensive R Archive Network) that calculate power for many statistics tests including the Welch t-test. Here we will use the power_t_test function from the MESS package.

```
# R version 3.6.0 (2019-04-26) -- "Planting of a Tree"

nx <- 35
ny <- 25
delta <- 2.6  # da - d0
alpha <- 0.05
sigmax <- 4
sigmay <- 2

## source("power_for_Welch_via_noncentral_t_distr.R")  # it has been "compiled" above
# see the comments in this function above for what is in
# the 1 row data frame it returns
```

```
# rerun the previous power calculation
power_for_Welch_via_noncentral_t_distr(nx, ny, delta, sigmax, sigmay, alpha)

###   power_two_sided_strict_FALSE power_one_sided wrong_side_power
### 1                    0.9012841       0.9475901     9.378596e-08
###   nx ny delta sigmax sigmay alpha     dfW      ncp
### 1 35 25   2.6      4      2  0.05 52.8017 3.309638

# dfW is the Welch degrees of freedom, ncp is the non-centrality parameter

# compare with the results from using power_t_test

library(MESS)
Warning message:
package 'MESS' was built under R version 3.6.3
packageVersion("MESS")
[1] '0.5.7'
?power_t_test

# for calculating power using power_t_test for a two group t-test:

# n is the smaller sample size if there are 2 groups (as will be the case here)
# delta is the effect size; (da - d0) in the notation we are using above
# sd is the standard deviation in the group with the smaller sample size
# sig.level is the significance level (alpha)
# power # set to NULL to calculate it given values for the other arguments
# ratio is the ratio of the larger sample size to the smaller sample size
#       (1 if equal)
# sd.ratio is the ratio of the standard deviations:
#          sd for the group with more samples / sd for the
#                           group with fewer samples
#          Need to be careful with sd.ratio if using
#                           power_t_test to solve for n when
#          have set ratio = 1 (so forcing equal sample sizes): check the two
#          output standard deviations to see if they are what was intended
# type is one of: c("two.sample", "one.sample", "paired")
# alternative is one of: c("two.sided", "one.sided")
# df.method = c("welch", "classical")  # method for calculating
#         the degrees of freedom for a two group (two.sample) t-test;
#         "classical" gives the equal variance (Student's) t-test degrees
#         of freedom (nx + ny - 2 in our notation),
#         "welch" gives dfW in our notation
# strict # FALSE to not include the "wrong side" contribution to the power
#          for a two-sided test.  NOTE the default in power_t_test is TRUE
```

```
# for the data above, there are more samples for the "x group", whose
# standard deviation is 4,
# and the standard deviation for the group with the smaller number
# of samples (here y) has standard deviation equal 2

# calculate the power for  alternative = "two.sided"

power_t_test(n = min(c(nx, ny)), delta = delta, sd = 2,
  sig.level = alpha, power = NULL, ratio = max(c(nx, ny)) / min(c(nx, ny)),
  sd.ratio = 2,   type = "two.sample", alternative = "two.sided",
  df.method = "welch", strict = FALSE)

     Two-sample t test power calculation with unequal sample sizes and unequal variances

              n = 25, 35
          delta = 2.6
             sd = 2, 4
      sig.level = 0.05
          power = 0.9012841
    alternative = two.sided

NOTE: n is vector of number in each group


# calculate the power for  alternative = "one.sided"

power_t_test(n = min(c(nx, ny)), delta = delta, sd = 2,
  sig.level = alpha, power = NULL, ratio = max(c(nx, ny)) / min(c(nx, ny)),
  sd.ratio = 2,   type = "two.sample", alternative = "one.sided",
  df.method = "welch", strict = FALSE)

     Two-sample t test power calculation with unequal sample sizes and unequal variances

              n = 25, 35
          delta = 2.6
             sd = 2, 4
      sig.level = 0.05
          power = 0.9475901
    alternative = one.sided

NOTE: n is vector of number in each group


### these agree with the two.sided and one.sided power results from
### power_for_Welch_via_noncentral_t_distr  for this example
```

Finding the number of samples needed to obtain a given power, with set values for delta, sigmax, sigmay, alpha

One can use power_t_test for this. One also could simply build a table (that is, a data frame, and then write it out to a file) containing the power for a relevant range of nx and ny values. In general, if the standard deviation in one population is larger than the standard deviation in the other, then one will "want to have" more samples from the population with the larger standard deviation.

Here is a small example of building a table (a data frame) of power values.

```
## source("power_for_Welch_via_noncentral_t_distr.R")  # it has been "compiled" above
# calculate power for a two-sided Welch t-test, with strict = FALSE

nxvec <- 37:43
nyvec <- 15:25
delta <- 2.6  # da - d0
alpha <- 0.05
sigmax <- 4
sigmay <- 2

# make data frame of Welch two-sided power (strict = FALSE) for
# a range of sample values in each group, for the parameter values above

# matrix to hold values
power.values <- matrix(0., nrow = 11, ncol = 7)

# irow and jcol are indices to place values into the power.values matrix

jcol <- 0L  # column index

for (nx in nxvec) {
   jcol <- jcol + 1L
   irow <- 0  # row index - reset to 0 at start of each column

   for (ny in nyvec) {
      irow <- irow + 1L
      pv <- power_for_Welch_via_noncentral_t_distr(nx, ny, delta,
                                                   sigmax, sigmay, alpha)[[1]]
      pv <- round(pv, digits = 5)
      power.values[irow, jcol] <- pv
   }
}

# construct the data frame
```

```
colnames(power.values) <- as.character(nxvec)
power.rows.are.ny.columns.are.nx <- data.frame("ny / nx -->" = nyvec, power.values,
                     check.names = FALSE)
# display it
power.rows.are.ny.columns.are.nx
```

```
##    ny / nx -->      37      38      39      40      41      42      43
## 1           15 0.86132 0.86689 0.87214 0.87710 0.88177 0.88618 0.89035
## 2           16 0.86993 0.87550 0.88073 0.88566 0.89030 0.89468 0.89882
## 3           17 0.87738 0.88293 0.88813 0.89303 0.89764 0.90198 0.90607
## 4           18 0.88388 0.88940 0.89457 0.89943 0.90400 0.90829 0.91233
## 5           19 0.88959 0.89508 0.90021 0.90503 0.90955 0.91380 0.91778
## 6           20 0.89464 0.90009 0.90519 0.90996 0.91444 0.91863 0.92256
## 7           21 0.89914 0.90455 0.90961 0.91434 0.91876 0.92290 0.92678
## 8           22 0.90316 0.90853 0.91355 0.91823 0.92261 0.92670 0.93053
## 9           23 0.90678 0.91211 0.91708 0.92172 0.92605 0.93009 0.93387
## 10          24 0.91005 0.91534 0.92027 0.92486 0.92915 0.93314 0.93687
## 11          25 0.91302 0.91827 0.92316 0.92771 0.93194 0.93589 0.93957
```

```
# write the data frame out to a tab delimited text file in the R working directory

outpfilename <- "power_calculation_example_May_2021.txt"
# write.table(power.rows.are.ny.columns.are.nx, file = outpfilename,
#             append = FALSE, quote = FALSE, sep = "\t",
#             row.names = FALSE, col.names = TRUE)
```

Another example: nx + ny = a fixed value N

Consider the situation where obtaining samples (or measurements for the samples) is quite expensive or difficult, so one has a "budget" of N total samples; so then one has the condition nx + ny = N, so ny = N - nx, and given values for delta, sigmax, sigmay, and alpha one wants to choose nx (and hence ny = N - nx) to obtain the maximum power subject to this constraint (nx + ny = N) and the given values for delta, sigmax, sigmay, and alpha. One could do this with an optimization function, but it is instructive to simply calculate the power over a "reasonable" range of nx values between, say, 3 and N - 3 (having only 2 samples in a group would be *really* "sparse"). This will clearly demonstrate that if, for example, sigmay > sigmax, one will likely be better off with ny > nx. In fact, if one "ignores" the effect of the number of degrees of freedom, which for the Welch t-test is a rather nonlinear function of nx, ny, vx = sigmax^2 and vy = sigmay^2, then (for the Welch t-test) one wants to choose nx to maximize the T-score given by **delta/sqrt**$(vx/nx + vy/ny)$ Hence (since here delta has a given value) one wants to minimize the quantity Q that occurs within the expression for the denominator of the T-score:

$$Q = vx/nx + vy/ny \qquad \text{equation (9)}$$

If sigmay equals some constant $K \geq 1$ times sigmax, and ny = N - nx, then we want to minimize

$$Q(nx) = vx/nx + K^2 vx/(N - nx)$$

Using calculus, the minimum of Q(nx) (for nx between 3 and N-3)) is where the derivative of $Q(nx)$ is 0, which leads to the result that if sigmay = K sigmax with K $\geq$ 1, then the minimum of Q(nx) occurs when

nx = N / (1 + K) and so then ny = N K / (1 + K) and ny = K nx equation (10a)

If sigmax > sigmay so sigmax = J sigmay with J $\geq$ 1, then, symmetric with equation (10a) (just exchange x and y, or set K = 1/J) we have

ny = N / (1 + J) and so then nx = N J / (1 + J) and nx = J ny equation (10b)

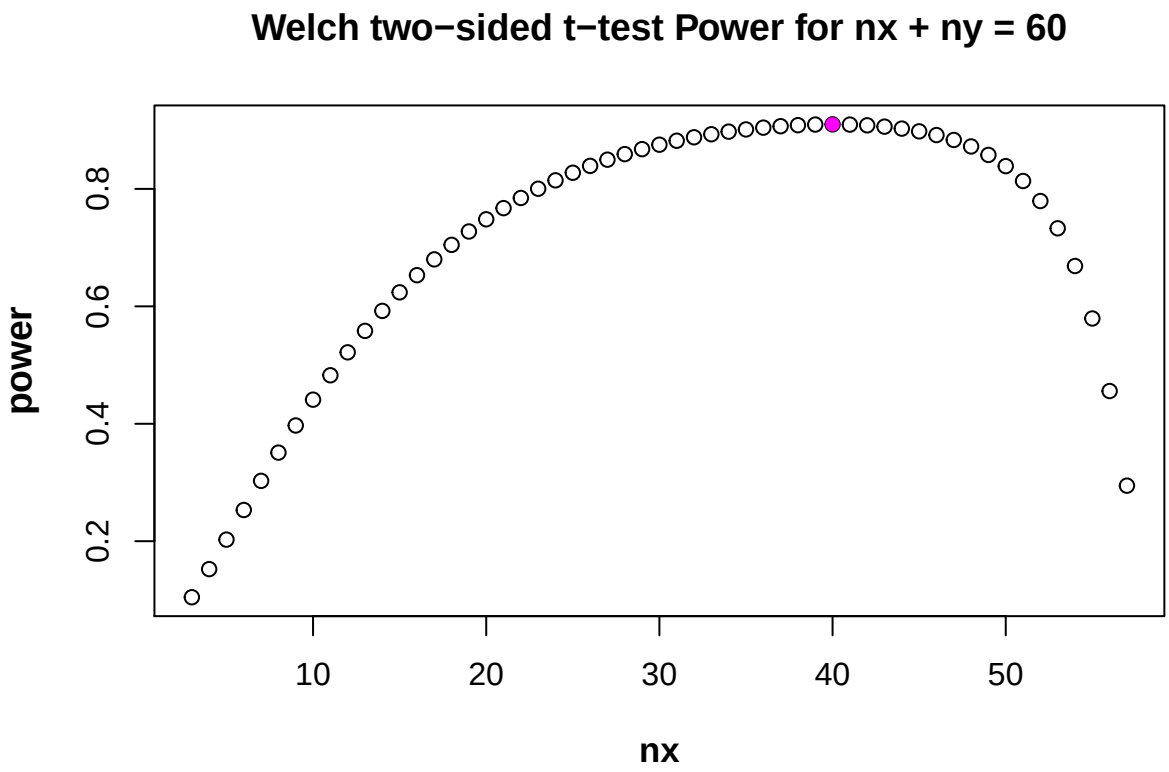
Note the formula for the power from the Welch t-test can be evaluated at non-integer values of nx and ny (though obviously we will want integer values that give (at least) a specified power, or that give the optimal power subject to a constraint such as nx + ny = N). This means for the case of nx + ny = N, so ny = N - nx, we can use an optimization function to search for the nx value that gives the maximum power, and then adjust to the nearby integers nx, ny, satisfying nx + ny = N, that give the best power.

For our example above, we have sigmax = 2 sigmay (i.e., the case J = 2 in equation (10b)), so if N is 60, the minimum of Q occurs for nx = 40 and ny = 20. With this many samples, we'll see that the dependence of the Welch t-test degrees of freedom given by dfW above on nx for nx near 40 (with N = 60 and J = 2) does not have a large effect, and the optimal choice to get the largest power given the constraint nx + ny = 60 and the values of the rest of the parameters; delta = 2.6, sigmax = 4, sigmay = 2, is very close to nx = 40, ny = 20 (and so nx = 40, ny = 20 is the optimal solution for integer values of nx and ny). The magenta colored point in the plot below is at nx = 40.

Here is the calculation verifying this

```
N <- 60L  # constrain nx + ny to be N
# find nx and ny = N - nx that give the best power
delta <- 2.6  # da - d0
alpha <- 0.05
sigmax <- 4
sigmay <- 2

# calculate the power for these values of delta, sigmax, sigmay
# for nx between 3 and N-3 = 57 (with ny = N - nx)

power.vec <- numeric(N)
nx.vec <- numeric(N)
for (nx in 3L:(N - 3L)) {
   ny = N - nx
   power.value <-
      power_for_Welch_via_noncentral_t_distr(nx, ny, delta,
```

```
                                        sigmax, sigmay, alpha)[[1]]
   nx.vec[nx] <- nx
   power.vec[nx]  <- power.value
}

xvals <- nx.vec[3L:(N - 3L)]
yvals <- power.vec[3L:(N - 3L)]

plot(xvals, yvals, xaxt = "n", xlab = ' ',  ylab = ' ')

# xaxt = "n" suppresses the x axis labeling so can specify it as we want

# display the point for nx = 40 in magenta
i40 <- which(xvals == 40)
x40 <- xvals[i40]  # should be 40
y40 <- yvals[i40]
lines(x40, y40, pch = 16, col = "magenta", type = "p")

axis(1, at = seq(10, 50, by = 10)) # requests tic marks and labels at the
# integers 10, 20, ... , 50

# add a plot title
title(main = "Welch two-sided t-test Power for nx + ny = 60", cex.lab = 1.1)

# axis labels

sub1 <- "ny = 60 - nx, strict = FALSE, "
sub2 <- "other parameters as given above"
subcharstring <- paste(sub1, sub2, sep = "")
title(main = NULL,
      sub = subcharstring,
      xlab = "nx", ylab = NULL, line = NA, outer = FALSE, font.lab = 2,
      cex.lab = 1.1, col.lab = "black")

title(main = NULL, sub = NULL, xlab = NULL, ylab = "power",
      line = NA, outer = FALSE, font.lab = 2,
      cex.lab = 1.2, col.lab = "black")
```




```
# the point in the plot in magenta is at nx = 40

power.vec[35:45]
```

```
##  [1] 0.9012841 0.9043629 0.9067808 0.9085345 0.9096069 0.9099642 0.9095523
##  [8] 0.9082910 0.9060667 0.9027216 0.8980382
```

```
# [1] 0.9012841 0.9043629 0.9067808 0.9085345 0.9096069 0.9099642 0.9095523
# [8] 0.9082910 0.9060667 0.9027216 0.8980382
```

So we see that indeed the optimal result (for integer nx, ny = N - nx) in this case is at nx = 40 (and ny = 20). The degrees of freedom does have a small effect in that if we don't require nx to be an integer (again, these formulas admit non-integer values for nx and ny), then the minimum is not exactly at nx = 40 as shown by the following bit of code below (but with N = 60 the effect of the degrees of freedom from the Welch formula is tiny for this example :

```
## source("power_for_Welch_via_noncentral_t_distr.R")  # it has been "compiled" above

# define the function to use in the optimization function
```

```
power_for_Welch_via_noncentral_t_distr_optimize_nx <-
    function(nx, N, delta, sigmax, sigmay, alpha) {
        ny <- N - nx
        result <- power_for_Welch_via_noncentral_t_distr(nx,
                        ny, delta, sigmax, sigmay, alpha)
        pwr <- result$power_two_sided_strict_FALSE
        return(pwr)
}


N <- 60L  # constrain nx + ny to be N
# find nx and ny = N - nx that give the best power
delta <- 2.6  # da - d0
alpha <- 0.05
sigmax <- 4
sigmay <- 2

# ?optimize
optimize(power_for_Welch_via_noncentral_t_distr_optimize_nx,
   interval = c(3, N-3),
   N = N, delta = delta, sigmax = sigmax, sigmay = sigmay, alpha = alpha,
   maximum = TRUE)

# $maximum
# [1] 39.97874

# $objective
# [1] 0.9099644

#############  Note the last digit printed is a 4 (rather than 2)

# the power for nx = 40, ny = 20 is
# very slightly smaller than 0.9099644 so the Welch degrees of freedom
# does matter, though not much for this many samples, given the other parameter values,
# so equation (10) leads to the optimal integer values for this example:

nx <- 40
ny <- 20

# display again the power for nx = 40, ny = 20

power_for_Welch_via_noncentral_t_distr(nx, ny, delta, sigmax, sigmay, alpha)
#   power_two_sided_strict_FALSE power_one_sided wrong_side_power nx ny delta
# 1                    0.9099642       0.9527557      7.10435e-08 40 20   2.6
#   sigmax sigmay alpha     dfW      ncp
```

```
# 1      4      2  0.05 57.9913 3.356586

##############

# and another example - with the same parameters except now N is 15
# again the choice nx = J ny from equation (10b) leads to the
# optimal integers nx, ny

N <- 15L

optimize(power_for_Welch_via_noncentral_t_distr_optimize_nx,
   interval = c(3, N-3),
   N = N, delta = delta, sigmax = sigmax, sigmay = sigmay, alpha = alpha,
   maximum = TRUE)

# $maximum
# [1] 9.922649

# $objective
# [1] 0.3426671

# calculate the power at nx = 10, ny = 5

nx <- 10
ny <- 5

> power_for_Welch_via_noncentral_t_distr(nx, ny, delta, sigmax, sigmay, alpha)
#   power_two_sided_strict_FALSE power_one_sided wrong_side_power nx ny delta
# 1                    0.3426068       0.4781344     0.0002123278 10  5   2.6
#   sigmax sigmay alpha   dfW      ncp
# 1      4      2  0.05 12.96 1.678293

# so we see the power at the the maximum (allowing non-integer values of nx) is again
# only slightly larger than the power at the integer values nx = 10, ny = 15 - nx = 5
# given using equation (10b) (here J = sigmax / sigmay = 2); and one can check
# by examining nearby integer pairs ((11, 4), (12, 3), (9, 6), (8, 7)) that indeed
# nx = 10, ny = 5 is the optimal integer pair for maximizing the power
# subject to nx + ny = 15 (and the values of the other parameters given above)
```

Hope this discussion of power is helpful.