Power and sample size

Patrick Breheny

October 27, 2016

Today's lab will focus on the two most common settings in which a researcher will carry out a power calculation: two-sample studies in which the outcome is continuous (i.e., power calculations for t-tests) and two-sample studies in which the outcome is binary (i.e., power caluclations for χ^2 tests).

1 Equal allocation

1.1 Continuous data

The base function in R for calculating power for t-tests is power.t.test. You supply n, the sample size in each group, $\Delta = \mu_1 - \mu_2$, the difference in means, and σ, the standard deviation, and the function returns the power:

```
> power.t.test(n=3, delta=3, sd=1)


     Two-sample t test power calculation

              n = 3
          delta = 3
             sd = 1
      sig.level = 0.05
          power = 0.7825542
    alternative = two.sided

NOTE: n is number in *each* group
```

You can also supply α, the type I error rate; this is set to $\alpha = 0.05$ by default. The function also allows you to pass the power and leave n blank, in which case the function will calculate the appropriate sample size for you:

```
> power.t.test(power=0.9, delta=3, sd=1)


     Two-sample t test power calculation

              n = 3.624818
          delta = 3
             sd = 1
      sig.level = 0.05
          power = 0.9
    alternative = two.sided
```

```
NOTE: n is number in *each* group
```

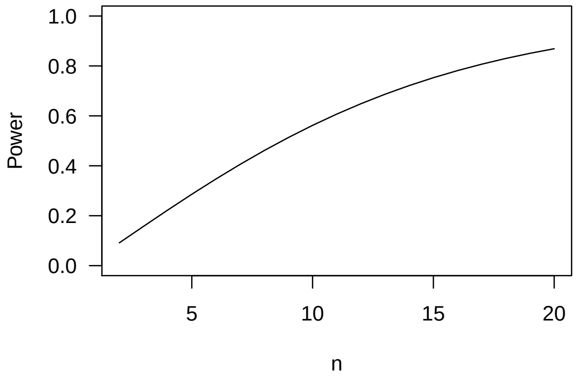
You can actually leave any of the above blank, and the function will solve for the remaining one. For example,

```
> power.t.test(power=0.9, n=2, sd=1)


     Two-sample t test power calculation

              n = 2
          delta = 6.795615
             sd = 1
      sig.level = 0.05
          power = 0.9
    alternative = two.sided

NOTE: n is number in *each* group
```

This returns $\Delta = 6.8$, meaning that the true difference in means would have to be at least 6.8 in order for us to detect that difference with 90% power if the sample size were 2 per group; this is sometimes called the *minimum detectable difference*. You cannot leave more than one option blank, or supply all arguments.

Conveniently, the function also accepts vectors as arguments, which makes it easy to construct power curves:

```
> n <- 2:20
> Power <- power.t.test(n=n, delta=2, sd=2)$power
> plot(n, Power, type="l", las=1, ylim=0:1)
```



As we discussed in class, there is no closed-form solution for sample size when we're dealing with the t distribution, but we can set up the calculation as a root-finding problem:

```
> power.t.test(power=0.9, delta=5, sd=2)$n

[1] 4.574784
```

```
> f <- function(n) {qt(0.975, 2*n-2) - qt(0.1, 2*n-2, 5/(2*sqrt(2/n)))}
> uniroot(f, c(2, 100))$root

[1] 4.574784
```

Note that we obtain a different answer for the t-test than we get from the formula we derived in class for the z-test:

```
> 2*((qnorm(0.975)+qnorm(0.9))/(5/2))^2

[1] 3.362375
```

The z-test calculation would indicate that $n = 4$ in each group easily suffices to obtain 90% power; this would be true if we knew the standard deviation, but the t-test calculations show that we need $n = 5$ in each group to overcome this limitation and achieve 90% power.

Finally, it is worth knowing that `power.t.test` can also be used for paired t-tests:

```
> power.t.test(n=6, delta=3, sd=2, type="paired")


     Paired t test power calculation

              n = 6
          delta = 3
             sd = 2
      sig.level = 0.05
          power = 0.8325291
    alternative = two.sided

NOTE: n is number of *pairs*, sd is std.dev. of *differences* within pairs

> power.t.test(n=6, delta=3, sd=2)


     Two-sample t test power calculation

              n = 6
          delta = 3
             sd = 2
      sig.level = 0.05
          power = 0.6495744
    alternative = two.sided

NOTE: n is number in *each* group
```

Note that the paired test is considerably more powerful, even though it only has $n = 6$ subjects (assuming a crossover design) while the unpaired study has 12 subjects. It should be said, though, that this calculation is somewhat misleading, as it assumes that σ is the same for each study, even though the two standard deviations represent rather different quantities (SD of individuals vs. SD of differences within an individual); without knowing more about the problem, it is impossible to say which one will be larger.

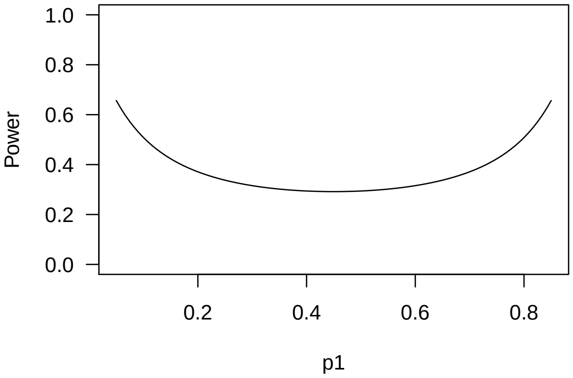
1.2 Binary data

`R` has a very similar function for binary data, `power.prop.test`. For binary data, we do not need to specify σ (since it is determined by π), but we do need to specify both π_1 and π_2, the proportions in the two groups (i.e., it is not sufficient to merely specify the difference, $\pi_1 - \pi_2$, as we did earlier). So, for example:

```
> power.prop.test(n=50, p1=.4, p2=.2)


     Two-sample comparison of proportions power calculation

              n = 50
             p1 = 0.4
             p2 = 0.2
      sig.level = 0.05
          power = 0.5900612
    alternative = two.sided

NOTE: n is number in *each* group

> power.prop.test(power=0.8, p1=.4, p2=.2)


     Two-sample comparison of proportions power calculation

              n = 81.22424
             p1 = 0.4
             p2 = 0.2
      sig.level = 0.05
          power = 0.8
    alternative = two.sided

NOTE: n is number in *each* group
```

We need a little over 80 subjects per group for 80% power in this setting; with $n = 50$ in each group, we only have $\approx 60\%$ power. As remarked earlier, power depends on both π_1 and π_2, not merely their difference:

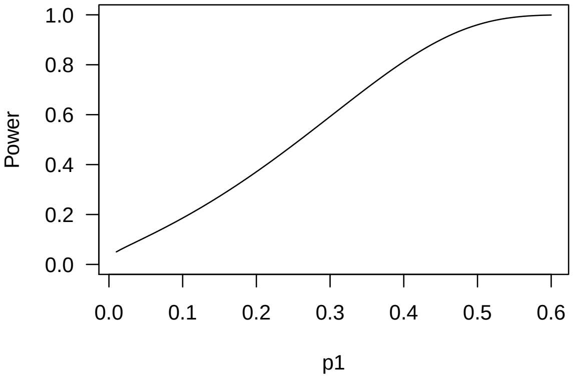
```
> p1 <- seq(0.05, 0.85, len=99)
> Power <- power.prop.test(n=100, p1=p1, p2=p1+0.1)$power
> plot(p1, Power, type="l", ylim=0:1, las=1)
```



In the plot, $\pi_2 - \pi_1 = 0.1$ at all points, but the power is higher for values of π_1 close to 0 and 1. Discussion: Why does this happen?

The power is also not constant if we keep the ratio of π_1 and π_2 the same:

```
> p1 <- seq(0.01, 0.6, len=99)
> Power <- power.prop.test(n=100, p1=p1, p2=p1*1.5)$power
> plot(p1, Power, type="l", ylim=0:1, las=1)
```



Here, π_2 is 50% larger than π_1 at all points, but there is an enormous difference in power as we change π_1. Discussion: Again, what is the explanation for this?

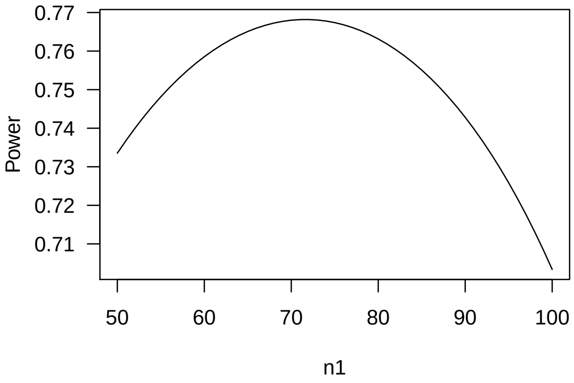
2 Unequal allocation

One limitation of `power.t.test` and `power.prop.test` is that they require $n_1 = n_2$; i.e., equal allotment of subjects to the two groups. This is often the case, but not always. Not infrequently, I am asked to calculate power in settings where the design uses a 1:2 or 1:3 ratio of sample sizes among groups, so it's useful to know about your options in such cases, although we have to go outside of base `R` for this.

2.1 Binary data

Frank Harrell's `Hmisc` package has a rather nice function `bpower` that offers many of the same features as `power.prop.test` but also allows you to specify different sample sizes in the two groups:

```
> require(Hmisc)
> bpower(n1=50, n2=100, p1=.4, p2=.2)

    Power
0.7335206

> ## A power curve where we vary the allocation
> n1 <- 50:100
> Power <- bpower(n1=n1, n2=150-n1, p1=.4, p2=.2)
> plot(n1, Power, type="l", las=1)
```



```
> n1[which.max(Power)]

[1] 72
```

As the power curve indicates, we actually obtain optimal power here when $n_1 = 72$ and $n_2 = 78$, although the gain in power over a balanced approach is minimal. The package also contains a useful function, `bsamsize` that solves for the necessary total sample size while keeping the ratio n_1/n_2 fixed:

```
> ## fraction: Fraction in group 1
> bsamsize(p1=.4, p2=.2, power=0.8, fraction=1/3)

       n1        n2
 59.10645 118.21290
```

So, if we intend a 1:2 ratio, we should aim for $n_1 = 60$ and $n_2 = 120$ to attain 80% power.

2.2 Continuous data

I really like the `bpower` and `bsamsize` functions and often wished that there was an equivalent function for *t*-tests. I never found one I liked, so I wrote my own:

```
> source("http://myweb.uiowa.edu/pbreheny/5710/f16/labs/tpower.R")
> tpower(n1=6, n2=6, delta=3, sd=2) ## Same as before

[1] 0.6495744

> tpower(n1=6, n2=12, delta=3, sd=2)

[1] 0.8040217

> ## w: Sampling weight for each group
> tsamsize(delta=3, sd=4, power=0.8, w=c(3,1))

      n1       n2
43.34945 14.44982
```

So, to achieve 80% power with a 3:1 ratio between groups, we need $n_1 = 45$ and $n_2 = 15$. A nice improvement to `tpower` and `tsamsize` would be to allow $\sigma_1 \neq \sigma_2$, but I've never been asked to perform such a power calculation, so I haven't (yet!) seen the need to extend it in this manner.

3 Conclusion

These two settings (two-sample studies with continuous and binary data) are the most commonly encountered, and any statistician should know how to calculate power and sample size in these settings. There are dozens of specialized packages out there for `R` that calculate power for specific situations (mixed models, genetics, pharmacology, etc.), but if you understand the general principles, you should be able to familiarize yourself with how those packages work (assuming, of course, you understand the underlying models!).