Of course, we're all thinking about one thing these days, so it seems particularly inconsequential to be writing about anything that doesn't contribute to solving or addressing in some meaningful way this pandemic crisis. But, I find that working provides a balm from reading and hearing all day about the events swirling around us, both here and afar. (I am in NYC, where things are definitely swirling.) And for me, working means blogging, at least for a few hours every couple of weeks.

I have tried in some small way to get involved with researchers who are trying to improve outcomes for patients who are showing the symptoms or test positive for COVID-19. One group that reached out to me is concerned with how patients with heart conditions will be adversely affected by the disease, and is evaluating a number of drug treatments that could improve their outcomes.

Given that we know that outcomes under usual care are not that great for heart patients, there is a desire to try to get possible treatments to as many people as possible, even in a randomized control trial. One question that came up in the design of this study was whether there would be efficiency gains by using a 1:2 randomization scheme? That is, should we randomize two patients to the experimental drug treatment for every one patient we randomize to the usual care group? In the case of a binary outcome, it appears that we will only *lose* efficiency if we use anything other than a 1:1 randomization.

Brief public service announcement: simstudy update

When it became clear that I needed to explore the implications of unbalanced randomization for this project, I realized that the `simstudy` package, which supports much of the simulations on this blog, could not readily handle anything other than 1:1 randomization. I had to quickly rectify that shortcoming. There is a new argument `ratio` in the `trtAssign` function where you can now specify any scheme for any number of treatment arms. This is available in version 1.16, which for the moment can be found only on github (kgoldfeld/simstudy).

Here is an example based on a 1:2:3 allocation. I'm not sure if that would ever be appropriate, but it shows the flexibility of the new argument. One counter-intuitive aspect of this implementation is that the `balance` argument is set to `TRUE`, indicating that the allocation to the groups will be perfect, or as close as possible to the specified ratios. If `balance` is `FALSE`, the ratios are used as relative probabilities instead.

```
library(simstudy)
library(parallel)

RNGkind("L'Ecuyer-CMRG")
set.seed(16)

dx <- genData(600)
dx <- trtAssign(dx, nTrt = 3, balanced = TRUE,
                ratio = c(1,2,3), grpName = "rx")

dx[, table(rx)]
```

```
## rx
##   1   2   3
## 100 200 300
```

Unbalanced designs with a binary outcome

The outcome in the COVID-19 study is a composite binary outcome (at least one of a series of bad events has to occur within 30 days to be considered a failure). Here, I am considering the effect of different randomization schemes on the power of the study. We assumed in the usual care arm 40% of the patients would have a bad outcome and that the drug treatment would reduce the bad outcomes by 30% (so that 28% of the drug treatment arm would have a bad outcome).

If we generate a single data set under these assumptions, we can fit a logistic regression model to recover these parameters.

```
estCoef <- function(n, formula, ratio) {

  def <- defDataAdd(varname = "y", formula = formula, dist = "binary")

  dx <- genData(n)
  dx <- trtAssign(dx, grpName = "rx", ratio = ratio)
  dx <- addColumns(def, dx)

  coef(summary(glm(y~rx, family = binomial, data = dx)))
}
```

```
estCoef(n = 244*2, formula = "0.4 - 0.3 * 0.4 * rx", ratio = c(1, 1))
```

```
##               Estimate Std. Error   z value     Pr(>|z|)
## (Intercept) -0.4328641  0.1310474 -3.303111 0.0009561867
## rx          -0.4577476  0.1924533 -2.378487 0.0173838304
```

The probabilities of a bad outcome for the usual care group and drug treatment group are

```
c(usual = 1/(1 + exp(0.433)), drug = 1/(1+exp(0.433 + 0.458)))
```

```
##     usual      drug
## 0.3934102 0.2909035
```

Assessing power

In order to assess power, we need to generate many data sets and keep track of the p-values. The power is calculated by estimating the proportion of p-values that fall below 0.05.

Here is the analytic solution for a 1:1 ratio.

```
power.prop.test(p1 = .4, p2 = .7*.4, power = .80)
```

```
##
##      Two-sample comparison of proportions power calculation
##
##               n = 243.4411
##              p1 = 0.4
##              p2 = 0.28
##       sig.level = 0.05
##           power = 0.8
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

The sample size estimate based on 80% suggests we would need 244 patients per arm, or 488 total patients. If we use this estimated \(n\) in a simulation for power (using 1000 datasets), we should be close to 80%:

```
est.power <- function(n, ratio, p1, reduction) {

  formula = paste(p1, "* (1 -", reduction, "* rx)")
  p.val <- estCoef(n, formula, ratio)["rx", 4]
  return(p.val)

}

pvals <- unlist(mclapply(1:1000,
    function(x) est.power(488, c(1, 1), 0.4, 0.3)))

mean(pvals < 0.05)
```

```
## [1] 0.814
```

The power experiment

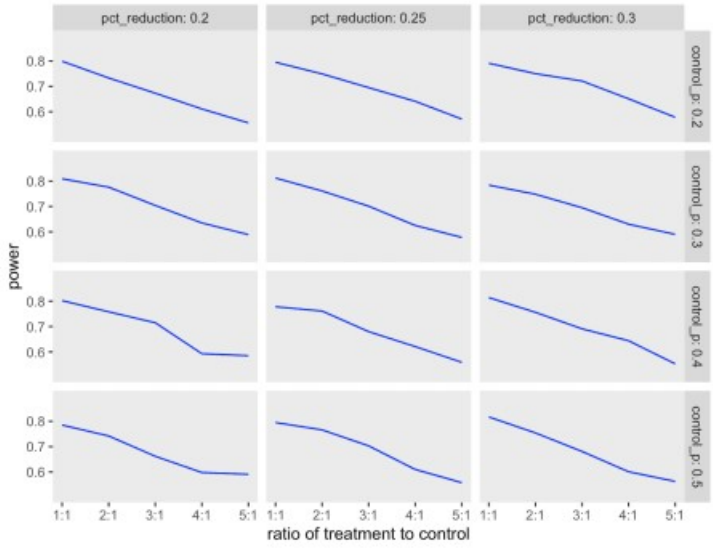
Now we are ready to evaluate the question that motivated all of this. If we start to change the ratio from 1:1 to 1:2, to 1:3, etc., what happens to the power? And does this pattern change based on the assumptions about failure rates in the usual care arm and the expected reductions in the treatment arm? Here is the code that will allow us to explore these questions:

```
res <- list()

for (p1 in c(.2, .3, .4, .5)) {
  for (r in c(.2, .25, .3)) {

    p2 <- (1- r) * p1
    n <- ceiling(power.prop.test(p1 = p1, p2 = p2, power = .80)$n)*2

    for (i in c(1:5)) {
```

```
      pvals <- mclapply(1:1000, function(x) est.power(n, c(1, i), p1, r))
      pvals <- unlist(pvals)

      dres <- data.table(n, control_p = p1, pct_reduction = r,
                         control = 1, rx = i, power = mean( pvals < .05))

      res <- append(res, list(dres))

    }
  }
}

res <- rbindlist(res)
```

Repeating the power simulation for a variety of assumptions indicates that, at least in the case of a binary outcome, using an unbalanced design does not improve the quality of the research even though it might get more patients the drug treatment:

```
ggplot(data = res, aes(x = rx, y = power)) +
  geom_line(color = "blue") +
  facet_grid(control_p ~ pct_reduction, labeller = label_both) +
  theme(panel.grid = element_blank()) +
  scale_x_continuous(name = "ratio of treatment to control",
                     breaks = c(1:5), labels = paste0(c(1:5),":1")) +
  scale_y_continuous(limits = c(.5,.9), breaks = c(.6, .7, .8))
```



Continuous outcomes

In the case of binary outcomes, reducing sample size in the control group reduces our ability to efficiently estimate the proportion of events, even though we may improve estimation for the treatment group by adding patients. In the case of a continuous outcome, we may be able to benefit from a shift of patients from one group to another if the variability of responses differs across groups. In particular, arms with more variability could benefit from a larger sample. Next time, I'll show some simulations that indicate this might be the case.