Computation of Power Curves

Zeki Kazan

2023-03-10

The following document provides code to produce the power curves for the Test of Tests as seen in Figures 2-4.

Note that when creating the final plots for the paper (saved in the `figures` folder), the lengths of `m_grid` and `effect_grid` are increased. The grids used in this document should produce values for power that differ from those in the final plots by less than 0.01.

Figure 2

We begin with Figure 2, which demonstrates a one-sample t-test. The vector of sample sizes and function for the public power are provided below.

```
n_vec <- c(seq(4,10,2), seq(20, 100, 20), seq(125, 200, 25), seq(300, 500, 100))

pub_power <- function(effect, n, alpha){
  power.t.test(n = n, delta = effect, sig.level = alpha, type = "one.sample",
               alternative = "two.sided", strict = TRUE)$power
}
```

We now create a vector of powers for the first panel.

```
epsilon <- 1; mu <- 0.4

power <- rep(NA, length(n_vec))

for(i in 1:length(n_vec)){
  pars <- practical.m.alpha0(rho = 0.9, pub_power = pub_power, epsilon = epsilon,
                             n = n_vec[i], effect_grid = seq(0.1, 1.5, 0.1))
  power[i] <- power.test.of.tests(effect = mu, pub_power = pub_power,
                                  epsilon = epsilon, n = n_vec[i], m = pars$m,
                                  alpha0 = pars$alpha0)
}
```

We now create a vector of powers for the second panel.

```
epsilon <- 0.1; mu <- 3

power <- rep(NA, length(n_vec))

for(i in 1:length(n_vec)){
```

```
  pars <- practical.m.alpha0(rho = 0.9, pub_power = pub_power, epsilon = epsilon,
                             n = n_vec[i], effect_grid = seq(0.1, 1.5, 0.1))
  power[i] <- power.test.of.tests(effect = mu, pub_power = pub_power,
                                  epsilon = epsilon, n = n_vec[i], m = pars$m,
                                  alpha0 = pars$alpha0)
}
```

Figure 3

We now do the same for Figure 3, which demonstrates a test for the mean of a multivariate normal distribution.

```
# The true mean is a vector of 0 n_zeros times and effect (d-n_zeros) times
pub_power <- function(effect, n, alpha, d, n_zeros){
  lambda = n*(d-n_zeros)*effect^2

  return(1-pchisq(qchisq(1-alpha, df = d), df = d, ncp = lambda))
}
```

The following creates the vector of powers for the first panel

```
n_vec <- c(seq(3,5,1), seq(10, 100, 5), seq(125, 300, 25), seq(500, 900, 200),
           seq(2000, 10000, 1000))

epsilon <- 1; effect <- 0.1; d <- 100; n_zeros = 0; alpha <- 0.5

power <- rep(NA, length(n_vec))

for(i in 1:length(n_vec)){
  rho <- ifelse(n_vec[i] <= 500, 0.9, 0.99)
  pars <- practical.m.alpha0(rho = rho, pub_power = pub_power, epsilon = epsilon,
                             n = n_vec[i], d = d, n_zeros = n_zeros, alpha = alpha,
                             effect_grid = seq(0.1, 1.5, 0.1))
  power[i] <- power.test.of.tests(effect = effect, pub_power = pub_power,
                                  epsilon = epsilon, n = n_vec[i], d = d,
                                  n_zeros = n_zeros, m = pars$m,
                                  alpha0 = pars$alpha0, alpha = alpha)
}
```

The following creates the vector of powers for the second panel

```
n_vec <- c(seq(3,5,1), seq(5, 100, 10), seq(150, 200, 20), 220, 230, 250, 260,
           300, seq(400, 500, 50), seq(600, 1000, 50), seq(1000, 10000, 1000))

epsilon <- 1; effect <- 0.5; d <- 100; n_zeros = 99; alpha <- 0.5

power <- rep(NA, length(n_vec))

for(i in 1:length(n_vec)){
  rho <- ifelse(n_vec[i] <= 500, 0.9, 0.99)
  pars <- practical.m.alpha0(rho = rho, pub_power = pub_power, epsilon = epsilon,
```

```
                              n = n_vec[i], d = d, n_zeros = n_zeros, alpha = alpha,
                              effect_grid = seq(0.1, 1.5, 0.1))
  power[i] <- power.test.of.tests(effect = effect, pub_power = pub_power,
                                  epsilon = epsilon, n = n_vec[i], d = d,
                                  n_zeros = n_zeros, m = pars$m,
                                  alpha0 = pars$alpha0, alpha = alpha)
}
```

Figure 4

We now perform a similar process for Figure 4, which demonstrates a one-way ANOVA. We are not aware of a closed form expression for the power of this test with uneven group sizes (which, unless the group assignments are public, is required to obtain a closed form for the power of our test, even if the data of interest has balanced groups). Thus instead, we will use the exact power calculation with public groups to determine the practical m and α_0. We then use these parameters to estimate the exact power via simulation.

```
# Sample sizes from Couch et al.'s saved data on GitHub
n_vec <- c(9, 12, 18, 27, 42, 60, 87, 123, 174, 240, 330, 450, 606, 807, 1071)

# Public power with balanced groups
pub_power <- function(effect, n, alpha, g){
  power.anova.test(groups = g, n = floor(n/3), between.var = effect,
                   within.var = 1, sig.level = alpha)$power
}

# Run an ANOVA and output p-value for data with response y and grouping group
pub_test <- function(data){
  s <- summary(aov(y ~ group, data = data))
  s[[1]]$`Pr(>F)`[1]
}
```

```
epsilon <- 1; effect <- 1; g <- 3; nsims <- 1e4; alpha <- 0.05

effect_grid <- c(seq(0.02, 0.2, 0.02), seq(0.4, 3, 0.2))

power <- rep(NA, length(n_vec))

for(i in 1:length(n_vec)){
  n <- n_vec[i]

  m_grid <- c(1:sqrt(n), floor(n/rev(1:(sqrt(n)+1))))
  m_grid <- m_grid[!duplicated(m_grid)]
  m_grid <- m_grid[m_grid <= n/6]

  pars <- practical.m.alpha0(rho = 0.9, pub_power = pub_power, epsilon = epsilon,
                             n = n, g = g, m_grid = m_grid, effect_grid = effect_grid)

  sims <- rep(NA, nsims)
  for(j in 1:nsims){
    data <- data.frame(y = rnorm(n, c(-effect, 0, effect)),
                       group = rep(as.character(1:g), n/g))
    sims[j] <- test.of.tests(data = data, pub_test = pub_test, epsilon = epsilon,
```

```
                                m = pars$m, alpha0 = pars$alpha0)$p.value
  }
  power[i] <- mean(sims <= alpha)
  print(n_vec[i])
}
```