Variuos results for hypothesis testing

Iyar Lin

01 June, 2020

Contents

Note: All entries in the table of contents are hyperlinks

1 Intro

This document contains proofs and formulas for several use cases encountered by the author in the context of experimental design. Some background in Statistics is preferable for understanding some of the results (You're welcome to contact the author in case you'd like to get more clarifications).

The first few sections contain proofs. Section Formulas for combining min required lift, number of sides and number of hypothesis tests contains unified formulae along with simulation validation.

2 Notations

Let a control population samples be denoted by: $X_1, \ldots X_{n_0} \sim Ber(p_0)$ where $Ber(p_0)$ is the Bernoulli distribution (meaning $P(X_i = 1) = p_0$ and conversely $P(X_i = 0) = 1 - p_0$).

We similarly denote the treatment population by $Y_1, \ldots Y_{n_1} \sim Ber(p_1)$.

We'd like to test weather applying the treatment results in any kind of lift.

Formally we'd like to test the null hypothesis

$$H_0 : p_1 - p_0 \leq 0$$

vs the alternative:

$$H_1 : p_1 - p_0 > 0$$

We estimate the population distribution parameter p using the population mean:

$$\hat{p_0} = \frac{\sum X_j}{n_0}, \hat{p_1} = \frac{\sum Y_j}{n_1}$$

We note that the "hat" in $\hat{p_i}$ means this is a random variable which is aimed at estimating the unknown parameter p_i.

Given that $\hat{p_1}$ is greater "enough" then $\hat{p_0}$ we can conclude that we should reject the null H_0 and accept the alternative H_1.

When rejecting the null we'd like to ensure that the probability we wrongly do so does not exceed some probability α (often 0.05).

3 Constructing the test

3.1 One sided test

To that end we'll construct a test such that we reject the null if the difference $\hat{p_1} - \hat{p_0}$ exceeds some critical value C with probability α when the null is in fact true:

$$P_{H_0}(\hat{p}_1 - \hat{p}_0 > C) = \alpha$$

The above probability is also called type 1 error.

We now turn to finding C.

We can subtract the mean (which is 0 under the null) from both sides of the inequality and divide by the standard deviation of $\hat{p}_1 - \hat{p}_0$:

$$P_{H_0}\left(\frac{\hat{p}_1 - \hat{p}_0}{SD(\hat{p}_1 - \hat{p}_0)} > \frac{C}{SD(\hat{p}_1 - \hat{p}_0)}\right) = \alpha$$

where $SD(\hat{p}_1 - \hat{p}_0)$ denotes the standard deviation of $\hat{p}_1 - \hat{p}_0$.

We note that according to the central limit therom:

$$\frac{\hat{p}_1 - \hat{p}_0}{SD(\hat{p}_1 - \hat{p}_0)} \underset{n\to\infty}{\rightsquigarrow} \mathcal{N}(0,1)$$

This means that $\frac{C}{SD(\hat{p}_1 - \hat{p}_0)}$ should equal the α quantile of the standard normal distribution (in the case of $\alpha = 0.05$ we have $\Phi^{-1}(0.95) = 1.65$):

$$\frac{C}{SD(\hat{p}_1 - \hat{p}_0)} = \Phi^{-1}(1-\alpha) \Rightarrow C = \Phi^{-1}(1-\alpha) \cdot SD(\hat{p}_1 - \hat{p}_0)$$

We use the unpooled variance estimator:

$$SD(\hat{p}_1 - \hat{p}_0) = \sqrt{p_1(1-p_1)/n_1 + p_0(1-p_0)/n_0}$$

and **finally**

$$\boxed{C = \Phi^{-1}(1-\alpha) \cdot \sqrt{\hat{p}_1(1-\hat{p}_1)/n_1 + \hat{p}_0(1-\hat{p}_0)/n_0}}$$

3.2 2 sided hypothesis test

If we we'd like to test if either populations has higher p (not necessarily treatment higher than control like in the previous section) we'd be doing what's called a 2 sided hypothesis test:

$$H_0 : p_1 - p_0 = 0$$

vs

$$H_1 : p_1 - p_0 \neq 0$$

In this case we'd be constructing a test of the form:

$$P_{H_0}(|\hat{p}_1 - \hat{p}_0| > C^{two}) = \alpha$$

Which translates to

$$P_{H_0}(\hat{p}_1 - \hat{p}_0 > C^{two} \cup \hat{p}_0 - \hat{p}_1 < -C^{two}) = \alpha$$

The events $\hat{p}_1 - \hat{p}_0 > C^{two}$ and $\hat{p}_0 - \hat{p}_1 < -C^{two}$ are disjoint thus we have

$$P_{H_0}(\hat{p}_1 - \hat{p}_0 > C^{two} \cup \hat{p}_0 - \hat{p}_1 < -C^{two}) = P_{H_0}(\hat{p}_1 - \hat{p}_0 > C^{two}) + P_{H_0}(\hat{p}_0 - \hat{p}_1 < -C^{two}) = \alpha$$

From symmetry of the Gaussian distribution we get that

$$P_{H_0}(\hat{p}_1 - \hat{p}_0 > C^{two}) = P_{H_0}(\hat{p}_0 - \hat{p}_1 < -C^{two}) = \frac{\alpha}{2}$$

Using the same calculations from the last section we get:

$$\boxed{C^{two} = \Phi^{-1}\left(1 - \frac{\alpha}{2}\right) \cdot \sqrt{\hat{p}_1(1-\hat{p}_1)/n_1 + \hat{p}_0(1-\hat{p}_0)/n_0}}$$

3.3 Requiring minimum lift

Sometimes instead of testing $p_1 > p_0$, we'd like to test a more demanding criteria $p_1 > p_0 + \gamma$ where γ is some minimum required lift (e.g. if applying the treatment costs us money such as in the case of direct mail pieces we'd like the treatment lift to be "at least as high as γ).

We'll thus test:

$$H_0 : p_1 - p_0 \leq \gamma$$

vs the alternative:

$$H_1 : p_1 - p_0 > \gamma$$

Now in order to standardize our statistic we'd subtract *gamma* instead of 0:

$$P_{H_0}\left(\frac{\hat{p}_1 - \hat{p}_0}{SD(\hat{p}_1 - \hat{p}_0)} > \frac{C - \gamma}{SD(\hat{p}_1 - \hat{p}_0)}\right) = \alpha$$

We next have

$$\frac{C - \gamma}{SD(\hat{p}_1 - \hat{p}_0)} = \Phi^{-1}(1 - \alpha) \Rightarrow C - \gamma = \Phi^{-1}(1 - \alpha) \cdot SD(\hat{p}_1 - \hat{p}_0)$$

And finally:

$$\boxed{C = \Phi^{-1}(1 - \alpha) \cdot \sqrt{\hat{p}_1(1-\hat{p}_1)/n_1 + \hat{p}_0(1-\hat{p}_0)/n_0} + \gamma}$$

4 Power

Let's assume that in reality the alternative is true (so $p_1 - p_0 > 0$).

We denote by β the probability of **not** rejecting the null (also known as type 2 error):

$$\beta = P_{H_1}\left(\hat{p}_1 - \hat{p}_0 < C\right)$$

We can then again subtract the mean and divide by the standard deviation (this time under H_1) and write:

$$\beta = P_{H_1}\left(\frac{\hat{p}_1 - \hat{p}_0 - (p_1 - p_0)}{SD(\hat{p}_1 - \hat{p}_0)} < \frac{C - (p_1 - p_0)}{SD(\hat{p}_1 - \hat{p}_0)}\right)$$

We note that according to the central limit therom:

$$\frac{\hat{p}_1 - \hat{p}_0 - (p_1 - p_0)}{SD(\hat{p_1} - \hat{p_0})} \underset{n\to\infty}{\rightsquigarrow} \mathcal{N}(0,1)$$

We can thus write the below equation:

$$\Phi^{-1}(\beta) = \frac{C - (p_1 - p_0)}{SD(\hat{p_1} - \hat{p_0})} \Rightarrow \beta = \Phi\left(\frac{C - (p_1 - p_0)}{SD(\hat{p_1} - \hat{p_0})}\right)$$

We can think about the test **power** as the probability of detecting the treatment effect (or conversely, not making the type 2 error β). We thus have that power is equal to $1 - \beta$ and:

$$1 - \beta = 1 - \Phi\left(\frac{C - (p_1 - p_0)}{SD(\hat{p_1} - \hat{p_0})}\right)$$

And finally the test power is:

$$\boxed{1 - \beta = 1 - \Phi\left(\frac{\Phi^{-1}(1-\alpha)\cdot\sqrt{p_1(1-p_1)/n_1 + p_0(1-p_0)/n_0} - (p_1 - p_0)}{\sqrt{p_1(1-p_1)/n_1 + p_0(1-p_0)/n_0}}\right)}$$

We can usually have a good "guess" regarding p_0 based on past data. p_1 is usually derived from the required minimum detectable effect such that $p_1 = p_0 + MDE$ and this is how it's implemented as a function.

5 MDE (minimum detectable effect)

Often times it's useful to choose a sample size and power and see what MDE they yield. Let's find the formula.

Starting from the equation arrived at in the power section:

$$\Phi^{-1}(\beta) = \frac{C - (p_1 - p_0)}{SD(\hat{p_1} - \hat{p_0})}$$

We can further develop:

$$\Phi^{-1}(\beta)\cdot\sqrt{p_1(1-p_1)/n_1 + p_0(1-p_0)/n_0} = \Phi^{-1}(1-\alpha)\cdot\sqrt{p_1(1-p_1)/n_1 + p_0(1-p_0)/n_0} - (p_1 - p_0) \Rightarrow$$

$$p_1 - p_0 = (\Phi^{-1}(1-\alpha) - \Phi^{-1}(\beta))\sqrt{p_1(1-p_1)/n_1 + p_0(1-p_0)/n_0}$$

Finally, using the fact that under H_0 we have $p_1 = p_0$ we get:

$$\boxed{MDE = p_1 - p_0 = \left(\Phi^{-1}(1-\alpha) - \Phi^{-1}(\beta)\right)\sqrt{p_0(1-p_0)/n_1 + p_0(1-p_0)/n_0}}$$

6 Required minimum sample size

Most often, an important part of an experiment design is calculating required sample sizes.

In this section we'll find the formula for calculating required minimum sample size assuming the treatment and control group samples are equal (equal samples sizes yield the highest power as demonstrated in section 2). Let's assume we'd like to obtain a test with power of $1-\beta$. Using the result from section 2 we can rearrange (also denote $n_0 = n_1 = n$):

$$\beta = \Phi\left(\frac{\Phi^{-1}(1-\alpha)\cdot\sqrt{\left(p_1(1-p_1)+p_0(1-p_0)\right)/n}-(p_1-p_0)}{\sqrt{(p_1(1-p_1)+p_0(1-p_0))/n}}\right) \Rightarrow$$

$$\Phi^{-1}(\beta) = \frac{\Phi^{-1}(1-\alpha)\cdot\sqrt{\left(p_1(1-p_1)+p_0(1-p_0)\right)/n}-(p_1-p_0)}{\sqrt{(p_1(1-p_1)+p_0(1-p_0))/n}} \Rightarrow$$

$$\Phi^{-1}(\beta)\cdot\sqrt{(p_1(1-p_1)+p_0(1-p_0))/n}+(p_1-p_0) = \Phi^{-1}(1-\alpha)\cdot\sqrt{\left(p_1(1-p_1)+p_0(1-p_0)\right)/n} \Rightarrow$$

$$(p_1-p_0) = \Phi^{-1}(1-\alpha)\cdot\sqrt{\left(p_1(1-p_1)+p_0(1-p_0)\right)/n}-\Phi^{-1}(\beta)\cdot\sqrt{(p_1(1-p_1)+p_0(1-p_0))/n} \Rightarrow$$

$$(p_1-p_0) = \left(\Phi^{-1}(1-\alpha)-\Phi^{-1}(\beta)\right)\sqrt{(p_1(1-p_1)+p_0(1-p_0))/n}$$

$$\sqrt{n} = \frac{\left(\Phi^{-1}(1-\alpha)-\Phi^{-1}(\beta)\right)\sqrt{p_1(1-p_1)+p_0(1-p_0)}}{(p_1-p_0)}$$

And **finally**:

$$\boxed{n = \left(\frac{\left(\Phi^{-1}(1-\alpha)-\Phi^{-1}(\beta)\right)\sqrt{p_1(1-p_1)+p_0(1-p_0)}}{(p_1-p_0)}\right)^2}$$

We can usually have a good "guess" regarding p_0 based on past data. p_1 is usually derived from the required minimum detectable effect such that $p_1 = p_0 + MDE$ and this is how it's implemented as a function.

7 Formulas for combining min required lift, number of sides and number of hypothesis tests

Below we can see the formulas presented above combining min required lift γ, number of sides $s \in \{1,2\}$ and number of hypothesis tests $h \in \mathbb{N}$ (Using Bonferroni correction for multiple hypothesis testing).

7.1 Critical value

The below is true for all cases **except** $s = 2 \cap \gamma \neq 0$.

$$\boxed{C = \Phi^{-1}(1-\frac{\alpha}{(s\cdot h)})\cdot\sqrt{\hat{p}_1(1-\hat{p}_1)/n_1+\hat{p}_0(1-\hat{p}_0)/n_0}+\gamma}$$

7.1.1 Simulation validation

```
rm(list = ls())
critical_value <- function(p_1_hat, n_1, p_0_hat, n_0, alpha, s, h, gamma) {
  if (gamma != 0 & s == 2) {
    gamma <- abs(gamma)
    s <- 1
  }
  qnorm(1 - alpha / (s * h)) *
    sqrt(p_1_hat * (1 - p_1_hat) / n_1 +
      p_0_hat * (1 - p_0_hat) / n_0) + gamma
}

p_0 <- 0.2
p_1 <- 0.2 # Null is true
n_0 <- 8000
n_1 <- 12000
alpha <- 0.05

s <- 1:2
h <- 1:5
gamma <- seq(-0.03, 0.02, 0.01)
M <- 10000
rejected_null <- array(
  dim = c(length(h), length(gamma), length(s)),
  dimnames = list(paste0("#tests = ", h), paste0("gamma = ", gamma), paste0("sides = ", s))
)

for (i in 1:dim(rejected_null)[1]) {
  for (j in 1:dim(rejected_null)[2]) {
    for (k in 1:dim(rejected_null)[3]) {
      p_1_hat <- as.matrix(replicate(
        n = h[i],
        rbinom(M, size = n_1, prob = p_1 + gamma[j]) / n_1
      ), ncol = h[i])

      p_0_hat <- as.matrix(replicate(
        n = h[i],
        rbinom(M, size = n_0, prob = p_0) / n_0
      ), ncol = h[i])
      const <- mapply(
        function(p_1_hat, p_0_hat) {
          critical_value(
            p_1_hat = p_1_hat, n_1 = rep(n_1, M),
            p_0_hat = p_0_hat, n_0 = rep(n_0, M),
            alpha = alpha, s = s[k], h = h[i],
            gamma = gamma[j]
          )
        },
        split(p_1_hat, rep(1:ncol(p_1_hat),
          each = nrow(p_1_hat)
        )),
        split(p_0_hat, rep(1:ncol(p_0_hat),
```

```
          each = nrow(p_0_hat)
        ))
      )

      if (s[k] == 2) {
        diff <- abs(p_1_hat - p_0_hat)
      } else {
        diff <- p_1_hat - p_0_hat
      }

      rejected_null[i, j, k] <- mean(apply(diff - const, 1, function(row) any(row > 0)))
    }
  }
}
```

Below we can see the results:

Table 1: sides = 1

	gamma = -0.03	gamma = -0.02	gamma = -0.01	gamma = 0	gamma = 0.01	gamma = 0.02
#tests = 1	0.0522	0.0491	0.0523	0.0511	0.0482	0.0483
#tests = 2	0.0525	0.0492	0.0497	0.0476	0.047	0.0455
#tests = 3	0.0512	0.0478	0.0528	0.0515	0.0512	0.0505
#tests = 4	0.0485	0.0537	0.051	0.05	0.0468	0.0503
#tests = 5	0.0498	0.0468	0.0541	0.0523	0.0513	0.0494

Table 2: sides = 2

	gamma = 0
#tests = 1	0.0482
#tests = 2	0.0493
#tests = 3	0.0485
#tests = 4	0.0475
#tests = 5	0.0493

7.2 Power

$$\boxed{1-\beta = 1-\Phi\left(\frac{\Phi^{-1}(1-\frac{\alpha}{s\cdot h})\cdot\sqrt{p_1(1-p_1)/n_1+p_0(1-p_0)/n_0}-(p_1-(p_0+\gamma))}{\sqrt{p_1(1-p_1)/n_1+p_0(1-p_0)/n_0}}\right)}$$

7.2.1 Simulation validation

```
power <- function(mde, n_1, p_0, n_0, alpha, s, h, gamma) {
  if (!s %in% c(1, 2)) stop("s has to be either 1 or 2")
  if (gamma < 0 & s == 1) stop("1 sided test (s=1) with negative minimum required lift (gamma < 0) doesn
  if (s == 2 & gamma < 0) {
    p_1 <- p_0
    p_0 <- p_1 + mde
```

```r
  } else {
    p_1 <- p_0 + mde
  }
  1 - pnorm((critical_value(p_1, n_1, p_0, n_0, alpha, s, h, gamma) -
    (p_1 - p_0)) / sqrt(p_1 * (1 - p_1) / n_1 + p_0 * (1 - p_0) / n_0))
}

p_0 <- 0.2
n_0 <- 8000
n_1 <- 12000
alpha <- 0.05
coef <- 1.05
s <- 1:2
h <- 1:5
gamma <- seq(-0.03, 0.02, 0.01)
M <- 10000

rejected_null_calc_power <- array(
  dim = c(length(h), length(gamma), length(s)),
  dimnames = list(paste0("#tests = ", h), paste0("gamma = ", gamma), paste0("sides = ", s))
)

for (i in 1:dim(rejected_null)[1]) {
  for (j in 1:dim(rejected_null)[2]) {
    for (k in 1:dim(rejected_null)[3]) {
      if (gamma[j] == 0) {
        mde <- 0.01 * coef # alternative is true
      } else if (gamma[j] < 0 & s[k] == 1) {
        next
      } else {
        mde <- gamma[j] * coef # alternative is true
      }
      p_1 <- p_0 + mde
      p_1_hat <- as.matrix(replicate(
        n = h[i],
        rbinom(M, size = n_1, prob = p_1) / n_1
      ), ncol = h[i])

      p_0_hat <- as.matrix(replicate(
        n = h[i],
        rbinom(M, size = n_0, prob = p_0) / n_0
      ), ncol = h[i])
      const <- mapply(
        function(p_1_hat, p_0_hat) {
          critical_value(
            p_1_hat = p_1_hat, n_1 = rep(n_1, M),
            p_0_hat = p_0_hat, n_0 = rep(n_0, M),
            alpha = alpha, s = s[k], h = h[i],
            gamma = gamma[j]
          )
        },
        split(p_1_hat, rep(1:ncol(p_1_hat),
          each = nrow(p_1_hat)
```

```
        )),
        split(p_0_hat, rep(1:ncol(p_0_hat),
          each = nrow(p_0_hat)
        ))
      )

      if (s[k] == 2) {
        diff <- abs(p_1_hat - p_0_hat)
      } else {
        diff <- p_1_hat - p_0_hat
      }

      rejected_null_calc_power[i, j, k] <- paste0(
        "calc = ",
        round(power(
          p_0 = p_0,
          mde = mde,
          n_1 = n_1,
          n_0 = n_0,
          alpha = alpha,
          s = s[k],
          h = h[i],
          gamma = gamma[j]
        ), 3),
        ", actual = ",
        round(mean(apply(
          diff - const, 1,
          function(row) {
            row[1] > 0
          }
        )), 3)
      )
    }
  }
}
```

Below we can see the results:

Table 3: sides = 1

	gamma = -0.03	gamma = -0.02	gamma = -0.01	gamma = 0	gamma = 0.01	gamma = 0.02
#tests = 1	NA	NA	NA	calc = 0.564, actual = 0.555	calc = 0.06, actual = 0.063	calc = 0.07, actual = 0.071
#tests = 2	NA	NA	NA	calc = 0.438, actual = 0.441	calc = 0.03, actual = 0.028	calc = 0.037, actual = 0.035
#tests = 3	NA	NA	NA	calc = 0.373, actual = 0.376	calc = 0.021, actual = 0.019	calc = 0.025, actual = 0.025
#tests = 4	NA	NA	NA	calc = 0.331, actual = 0.335	calc = 0.016, actual = 0.016	calc = 0.019, actual = 0.021
#tests = 5	NA	NA	NA	calc = 0.301, actual = 0.304	calc = 0.013, actual = 0.013	calc = 0.016, actual = 0.015

Table 4: sides = 2

	gamma = 0
#tests = 1	calc = 0.438, actual = 0.44
#tests = 2	calc = 0.331, actual = 0.332
#tests = 3	calc = 0.278, actual = 0.277
#tests = 4	calc = 0.244, actual = 0.239
#tests = 5	calc = 0.22, actual = 0.225

7.3 MDE

$$\boxed{MDE = p_1 - p_0 = \left(\Phi^{-1}(1 - \frac{\alpha}{s \cdot h}) - \Phi^{-1}(\beta)\right) \sqrt{p_0(1-p_0)/n_1 + p_0(1-p_0)/n_0} + \gamma}$$

7.3.1 Simulation validation

```
MDE <- function(power, n_1, p_0, n_0, alpha, s, h, gamma) {
  if (s == 2) gamma <- abs(gamma)
  (qnorm(1 - alpha / (s * h)) - qnorm(1 - power)) *
    sqrt(p_0 * (1 - p_0) / n_1 + p_0 * (1 - p_0) / n_0) + gamma
}

p_0 <- 0.2
n_0 <- 8000
n_1 <- 12000
alpha <- 0.05
s <- 1:2
h <- 1:5
gamma <- seq(-0.03, 0.02, 0.01)
M <- 10000
pow <- 0.8
rejected_null_calc_MDE <- array(
  dim = c(length(h), length(gamma), length(s)),
  dimnames = list(paste0("#tests = ", h), paste0("gamma = ", gamma), paste0("sides = ", s))
)

for (i in 1:dim(rejected_null)[1]) {
  for (j in 1:dim(rejected_null)[2]) {
    for (k in 1:dim(rejected_null)[3]) {
      mde <- MDE(
        power = pow, n_1 = n_1, n_0 = n_0, p_0 = p_0, alpha = alpha,
        s = s[k], h = h[i], gamma = gamma[j]
      )
      p_1 <- p_0 + mde # MDE is true

      p_1_hat <- as.matrix(replicate(
        n = h[i],
        rbinom(M, size = n_1, prob = p_1) / n_1
      ), ncol = h[i])

      p_0_hat <- as.matrix(replicate(
```

```
        n = h[i],
        rbinom(M, size = n_0, prob = p_0) / n_0
      ), ncol = h[i])
      const <- mapply(
        function(p_1_hat, p_0_hat) {
          critical_value(
            p_1_hat = p_1_hat, n_1 = rep(n_1, M),
            p_0_hat = p_0_hat, n_0 = rep(n_0, M),
            alpha = alpha, s = s[k], h = h[i],
            gamma = gamma[j]
          )
        },
        split(p_1_hat, rep(1:ncol(p_1_hat),
          each = nrow(p_1_hat)
        )),
        split(p_0_hat, rep(1:ncol(p_0_hat),
          each = nrow(p_0_hat)
        ))
      )

      if (s[k] == 2) {
        diff <- abs(p_1_hat - p_0_hat)
      } else {
        diff <- p_1_hat - p_0_hat
      }

      rejected_null_calc_MDE[i, j, k] <- paste0(
        "MDE = ",
        round(mde, 4),
        ", rejected = ",
        round(mean(apply(
          diff - const, 1,
          function(row) {
            row[1] > 0
          }
        )), 3)
      )
    }
  }
}
```

Below we can see the results:

Table 5: sides = 1

	gamma = -0.03	gamma = -0.02	gamma = -0.01	gamma = 0	gamma = 0.01	gamma = 0.02
#tests = 1	MDE = -0.0156, rejected = 0.808	MDE = -0.0056, rejected = 0.799	MDE = 0.0044, rejected = 0.792	MDE = 0.0144, rejected = 0.78	MDE = 0.0244, rejected = 0.789	MDE = 0.0344, rejected = 0.785

	gamma = -0.03	gamma = -0.02	gamma = -0.01	gamma = 0	gamma = 0.01	gamma = 0.02
#tests = 2	MDE = -0.0138, rejected = 0.809	MDE = -0.0038, rejected = 0.806	MDE = 0.0062, rejected = 0.795	MDE = 0.0162, rejected = 0.786	MDE = 0.0262, rejected = 0.785	MDE = 0.0362, rejected = 0.785
#tests = 3	MDE = -0.0129, rejected = 0.81	MDE = -0.0029, rejected = 0.803	MDE = 0.0071, rejected = 0.798	MDE = 0.0171, rejected = 0.796	MDE = 0.0271, rejected = 0.781	MDE = 0.0371, rejected = 0.78
#tests = 4	MDE = -0.0122, rejected = 0.813	MDE = -0.0022, rejected = 0.801	MDE = 0.0078, rejected = 0.793	MDE = 0.0178, rejected = 0.781	MDE = 0.0278, rejected = 0.784	MDE = 0.0378, rejected = 0.777
#tests = 5	MDE = -0.0117, rejected = 0.804	MDE = -0.0017, rejected = 0.796	MDE = 0.0083, rejected = 0.793	MDE = 0.0183, rejected = 0.787	MDE = 0.0283, rejected = 0.778	MDE = 0.0383, rejected = 0.775

Table 6: sides = 2

	gamma = 0
#tests = 1	MDE = 0.0162, rejected = 0.787
#tests = 2	MDE = 0.0178, rejected = 0.797
#tests = 3	MDE = 0.0187, rejected = 0.788
#tests = 4	MDE = 0.0193, rejected = 0.789
#tests = 5	MDE = 0.0197, rejected = 0.782

7.4 Required minimum sample size

$$\boxed{n = \left(\frac{\left(\Phi^{-1}(1-\frac{\alpha}{s\cdot h}) - \Phi^{-1}(\beta)\right)\sqrt{p_1(1-p_1)+p_0(1-p_0)}}{(p_1-(p_0+\gamma))}\right)^2}$$

7.4.1 Simulation validation

```
min_sample_size <- function(mde, p_0, alpha, s, h, gamma, power) {
  if (gamma < 0 & s == 1) stop("1 sided test (s=1) with negative minimum required lift (gamma < 0) does
  p_1 <- p_0 + mde
  round((((qnorm(1 - alpha / (s * h)) - qnorm(1 - power)) *
    sqrt(p_1 * (1 - p_1) + p_0 *
      (1 - p_0))) / (p_1 - (p_0 + gamma)))^2)
}

p_0 <- 0.2
alpha <- 0.05
coef <- 1.3
s <- 1:2
h <- 1:5
```

```r
gamma <- seq(-0.03, 0.02, 0.01)
M <- 10000
pow <- 0.8

rejected_null_calc_min_n <- array(
  dim = c(length(h), length(gamma), length(s)),
  dimnames = list(paste0("#tests = ", h), paste0("gamma = ", gamma), paste0("sides = ", s))
)

for (i in 1:dim(rejected_null)[1]) {
  for (j in 1:dim(rejected_null)[2]) {
    for (k in 1:dim(rejected_null)[3]) {
      if (gamma[j] == 0) {
        mde <- 0.01 * coef # alternative is true
      } else if (gamma[j] < 0 & s[k] == 1) {
        next
      } else {
        mde <- gamma[j] * coef # alternative is true
      }
      p_1 <- p_0 + mde

      n_0 <- n_1 <- min_sample_size(
        mde = mde, p_0 = p_0, alpha = alpha,
        s = s[k], h = h[i], gamma = gamma[j], power = pow
      )
      p_1_hat <- as.matrix(replicate(
        n = h[i],
        rbinom(M, size = n_1, prob = p_1) / n_1
      ), ncol = h[i])

      p_0_hat <- as.matrix(replicate(
        n = h[i],
        rbinom(M, size = n_0, prob = p_0) / n_0
      ), ncol = h[i])
      const <- mapply(
        function(p_1_hat, p_0_hat) {
          critical_value(
            p_1_hat = p_1_hat, n_1 = rep(n_1, M),
            p_0_hat = p_0_hat, n_0 = rep(n_0, M),
            alpha = alpha, s = s[k], h = h[i],
            gamma = gamma[j]
          )
        },
        split(p_1_hat, rep(1:ncol(p_1_hat),
          each = nrow(p_1_hat)
        )),
        split(p_0_hat, rep(1:ncol(p_0_hat),
          each = nrow(p_0_hat)
        ))
      )

      if (s[k] == 2) {
        diff <- abs(p_1_hat - p_0_hat)
```

```r
      } else {
        diff <- p_1_hat - p_0_hat
      }

      rejected_null_calc_min_n[i, j, k] <- paste0(
        "n = ",
        n_0,
        ", rejected = ",
        round(mean(apply(
          diff - const, 1,
          function(row) {
            row[1] > 0
          }
        )), 3)
      )
    }
  }
}
```

Below we can see the results:

Table 7: sides = 1

	gamma = -0.03	gamma = -0.02	gamma = -0.01	gamma = 0	gamma = 0.01	gamma = 0.02
#tests = 1	NA	NA	NA	n = 11986, rejected = 0.798	n = 225066, rejected = 0.796	n = 57519, rejected = 0.802
#tests = 2	NA	NA	NA	n = 15216, rejected = 0.805	n = 285726, rejected = 0.797	n = 73022, rejected = 0.802
#tests = 3	NA	NA	NA	n = 17097, rejected = 0.799	n = 321039, rejected = 0.798	n = 82046, rejected = 0.794
#tests = 4	NA	NA	NA	n = 18427, rejected = 0.8	n = 346016, rejected = 0.8	n = 88430, rejected = 0.794
#tests = 5	NA	NA	NA	n = 19456, rejected = 0.804	n = 365346, rejected = 0.796	n = 93370, rejected = 0.798

Table 8: sides = 2

	gamma = -0.03	gamma = -0.02	gamma = -0.01	gamma = 0	gamma = 0.01	gamma = 0.02
#tests = 1	n = 28593, rejected = 0.878	n = 66219, rejected = 0.878	n = 272122, rejected = 0.876	n = 15216, rejected = 0.802	n = 285726, rejected = 0.879	n = 73022, rejected = 0.878
#tests = 2	n = 34626, rejected = 0.863	n = 80192, rejected = 0.868	n = 329541, rejected = 0.869	n = 18427, rejected = 0.803	n = 346016, rejected = 0.864	n = 88430, rejected = 0.874
#tests = 3	n = 38138, rejected = 0.87	n = 88326, rejected = 0.862	n = 362965, rejected = 0.866	n = 20296, rejected = 0.802	n = 381112, rejected = 0.864	n = 97399, rejected = 0.871
#tests = 4	n = 40623, rejected = 0.859	n = 94079, rejected = 0.868	n = 386610, rejected = 0.861	n = 21618, rejected = 0.798	n = 405939, rejected = 0.861	n = 103744, rejected = 0.862

	gamma = -0.03	gamma = -0.02	gamma = -0.01	gamma = 0	gamma = 0.01	gamma = 0.02
#tests = 5	n = 42546, rejected = 0.861	n = 98533, rejected = 0.865	n = 404911, rejected = 0.861	n = 22641, rejected = 0.802	n = 425155, rejected = 0.866	n = 108655, rejected = 0.865

8 Confidence intervals

Below we have the formula for the confidence interval.

We note it does not included MDE, power or γ. As such, it's a tool to convey estimate uncertainty, rather then for planning an experiment.

$$\boxed{CI = \hat{p}_1 - \hat{p}_0 \pm \Phi^{-1}(1 - \frac{\alpha}{2h})\sqrt{\hat{p}_1(1-\hat{p}_1)/n_1 + \hat{p}_0(1-\hat{p}_0)/n_0}}$$

8.1 Simulation validation

```
CI <- function(p_1_hat, n_1, p_0_hat, n_0, alpha, h) {
  point_estimate <- p_1_hat - p_0_hat
  rhs <- qnorm(1 - alpha / (2 * h)) * sqrt(p_1_hat * (1 - p_1_hat) / n_1 +
    p_0_hat * (1 - p_0_hat) / n_0)
  ans <- list(
    lower = point_estimate - rhs,
    upper = point_estimate + rhs
  )
  return(ans)
}

p_0 <- 0.2
p_1 <- 0.2 # Null is true
n_0 <- 16000
n_1 <- 24000
alpha <- c(0.01, 0.05, 0.1, 0.25)
diff <- c(-0.02, 0, 0.03)

h <- 1:5
M <- 10000
CI_coverage <- array(
  dim = c(length(h), length(alpha), length(diff)),
  dimnames = list(
    paste0("#tests = ", h), paste0("alpha = ", alpha),
    paste0("diff = ", diff)
  )
)

for (i in 1:dim(CI_coverage)[1]) {
  for (j in 1:dim(CI_coverage)[2]) {
    for (k in 1:dim(CI_coverage)[3]) {
      p_1_hat <- as.matrix(replicate(
```

```r
        n = h[i],
        rbinom(M, size = n_1, prob = p_0 + diff[k]) / n_1
      ), ncol = h[i])

      p_0_hat <- as.matrix(replicate(
        n = h[i],
        rbinom(M, size = n_0, prob = p_0) / n_0
      ), ncol = h[i])

      CI_mat <- mapply(
        function(p_1_hat, p_0_hat) {
          ci <- CI(
            p_1_hat = p_1_hat, n_1 = rep(n_1, M),
            p_0_hat = p_0_hat, n_0 = rep(n_0, M),
            alpha = alpha[j], h = h[i]
          )
          in_ci <- ci$lower < diff[k] & diff[k] < ci$upper
          return(in_ci)
        },
        split(p_1_hat, rep(1:ncol(p_1_hat),
          each = nrow(p_1_hat)
        )),
        split(p_0_hat, rep(1:ncol(p_0_hat),
          each = nrow(p_0_hat)
        ))
      )

      CI_coverage[i, j, k] <- mean(apply(CI_mat, 1, function(row) any(row == 0)))
    }
  }
}
```

Below we can see the results:

Table 9: diff = -0.02

	alpha = 0.01	alpha = 0.05	alpha = 0.1	alpha = 0.25
#tests = 1	0.0094	0.0525	0.098	0.2501
#tests = 2	0.01	0.0497	0.0964	0.2304
#tests = 3	0.01	0.0477	0.0991	0.2221
#tests = 4	0.0092	0.0501	0.0976	0.2333
#tests = 5	0.0097	0.0459	0.0992	0.23

Table 10: diff = 0

	alpha = 0.01	alpha = 0.05	alpha = 0.1	alpha = 0.25
#tests = 1	0.0094	0.0487	0.0977	0.2511
#tests = 2	0.0111	0.0516	0.0938	0.2343
#tests = 3	0.0103	0.0484	0.0967	0.2253
#tests = 4	0.0096	0.0492	0.0955	0.225
#tests = 5	0.0099	0.0473	0.0987	0.2271

Table 11: diff = 0.03

	alpha = 0.01	alpha = 0.05	alpha = 0.1	alpha = 0.25
#tests = 1	0.0095	0.051	0.0983	0.2444
#tests = 2	0.0092	0.0504	0.0975	0.2411
#tests = 3	0.0092	0.0481	0.0954	0.2323
#tests = 4	0.0098	0.051	0.0945	0.2336
#tests = 5	0.0085	0.0478	0.0928	0.2272

9 Testing with several samples

Do note this section isn't implemented yet in the *hype* package.

9.1 Notations

Sometimes we'd like to collect several samples and do hypothesis testing over them.

Formally speaking let's assume we collect samples in 2 periods (this will be further generalized to T periods later). We have the populations:

First period control:

$$X_1^1, \dots X_{n_0^1}^1 \sim Ber(p_0^1)$$

First period treatment:

$$Y_1^1, \dots Y_{n_1^1}^1 \sim Ber(p_1^1)$$

Second period control:

$$X_1^2, \dots X_{n_0^2}^2 \sim Ber(p_0^2)$$

First period treatment:

$$Y_1^2, \dots Y_{n_0^2}^2 \sim Ber(p_1^2)$$

While we don't assume $p_0^1 = p_0^2$ or $p_1^1 = p_1^2$ we do assume that the lift between treatment and control is the same such that $p_1^1 - p_0^1 = p_1^2 - p_0^2 = \delta$.

So far we've really tested

$$H_0 : \delta \leq 0$$

vs

$$H_1 : \delta > 0$$

We can estimate the lift using the average of both periods estimates:

$$\hat{\delta} = \frac{\hat{\delta_1} + \hat{\delta_2}}{2}$$

where $\hat{\delta_1} = \hat{p}_1^1 - \hat{p}_0^1$ and $\hat{\delta_2} = \hat{p}_1^2 - \hat{p}_0^2$.

Let's assume that $var(\delta_1) < var(\delta_2)$. The question arises: when should we use $\hat{\delta}$ instead of $\hat{\delta}_1$?

We'd do so when $var(\hat{\delta}_1) > var(\hat{\delta})$. This happens when:

$$var(\hat{\delta}_1) > var(\hat{\delta}) = var\left(\frac{\hat{\delta}_1 + \hat{\delta}_2}{2}\right) = \frac{1}{4}\left(var(\hat{\delta}_1) + var(\hat{\delta}_2)\right)$$

re-arranging we get

$$3 \cdot var(\hat{\delta}_1) > var(\hat{\delta}_2)$$

Given T periods we can use this logic to choose periods in an inductive way.

9.2 Constructing the test

Again, we'd like to construct a test of the form

$$P_{H_0}(\hat{\delta} > C) = \alpha$$

We note that the standard deviation of our estimator for this case is:

$$SD(\hat{\delta}) = \frac{1}{2}\sqrt{var(\hat{\delta}_1) + var(\hat{\delta}_2)} = \frac{1}{2}\sqrt{p_1^1(1-p_1^1)/n_1^1 + p_0^1(1-p_0^1)/n_0^1 + p_1^2(1-p_1^2)/n_1^2 + p_0^2(1-p_0^2)/n_0^2}$$

We thus have that our constant is:

$$\boxed{C = \Phi^{-1}(1-\alpha) \cdot \frac{1}{2}\sqrt{p_1^1(1-p_1^1)/n_1^1 + p_0^1(1-p_0^1)/n_0^1 + p_1^2(1-p_1^2)/n_1^2 + p_0^2(1-p_0^2)/n_0^2}}$$

Given T periods used in the test we have:

$$\boxed{C = \Phi^{-1}(1-\alpha) \cdot \frac{1}{T}\left(var(\delta^1) + var(\delta^2) + \cdots + var(\delta^T)\right)}$$

The rest of the results in this document can be found in a similar manner by swapping the SD term.

9.2.1 Simulation validation

Let's validate using 3 periods, different sample sized and base conversion rates.

```
rm(list = ls())
p_01 <- p_11 <- 0.2 # Null is true
p_02 <- p_12 <- 0.24
p_03 <- p_13 <- 0.23

n_01 <- 5000
n_11 <- 7000
n_02 <- 4000
n_12 <- 10000
n_03 <- 3000
n_13 <- 5000

alpha <- 0.05
```

```r
M <- 100000 # number of simulations
p_01_hat <- rbinom(M, size = n_01, prob = p_01) / n_01
p_11_hat <- rbinom(M, size = n_11, prob = p_11) / n_11
p_02_hat <- rbinom(M, size = n_02, prob = p_02) / n_02
p_12_hat <- rbinom(M, size = n_12, prob = p_12) / n_12
p_03_hat <- rbinom(M, size = n_03, prob = p_03) / n_03
p_13_hat <- rbinom(M, size = n_13, prob = p_13) / n_13

var_1_hat <- p_11_hat * (1 - p_11_hat) / n_11 + p_01_hat * (1 - p_01_hat) / n_01
var_2_hat <- p_12_hat * (1 - p_12_hat) / n_12 + p_02_hat * (1 - p_02_hat) / n_02
var_3_hat <- p_13_hat * (1 - p_13_hat) / n_13 + p_03_hat * (1 - p_03_hat) / n_03

C <- qnorm(1 - alpha) * (1 / 3) * sqrt(var_1_hat + var_2_hat + var_3_hat)

diff <- ((p_11_hat - p_01_hat) + (p_12_hat - p_02_hat) + (p_13_hat - p_03_hat)) / 3
rejected_null <- mean(diff > C)
```

We rejected 0.05084 of simulations. Cool.

9.3 Power

General formula for T periods is

$$\boxed{1-\beta = 1-\Phi\left(\frac{\Phi^{-1}(1-\alpha)\cdot\frac{1}{T}\sqrt{var(\hat{\delta^1})+var(\hat{\delta^2})+\cdots+var(\hat{\delta^T})}-\delta)}{\frac{1}{T}\sqrt{var(\hat{\delta^1})+var(\hat{\delta^2})+\cdots+var(\hat{\delta^T})}}\right)}$$

Let's validate that!

9.3.1 Simulation validation

```r
rm(list = ls())
delta <- 0.003
p_01 <- 0.2
p_11 <- p_01 + delta
p_02 <- 0.24
p_12 <- p_02 + delta
p_03 <- 0.23
p_13 <- p_03 + delta

n_01 <- 5000
n_11 <- 7000
n_02 <- 4000
n_12 <- 10000
n_03 <- 3000
n_13 <- 5000

var_1 <- p_11 * (1 - p_11) / n_11 + p_01 * (1 - p_01) / n_01
var_2 <- p_12 * (1 - p_12) / n_12 + p_02 * (1 - p_02) / n_02
var_3 <- p_13 * (1 - p_13) / n_13 + p_03 * (1 - p_03) / n_03
```

```r
alpha <- 0.05

power_calc <- 1 -
  pnorm((qnorm(1 - alpha) * (1 / 3) * sqrt(var_1 + var_2 + var_3) - delta) /
    ((1 / 3) * sqrt(var_1 + var_2 + var_3)))

M <- 100000 # number of simulations
p_01_hat <- rbinom(M, size = n_01, prob = p_01) / n_01
p_11_hat <- rbinom(M, size = n_11, prob = p_11) / n_11
p_02_hat <- rbinom(M, size = n_02, prob = p_02) / n_02
p_12_hat <- rbinom(M, size = n_12, prob = p_12) / n_12
p_03_hat <- rbinom(M, size = n_03, prob = p_03) / n_03
p_13_hat <- rbinom(M, size = n_13, prob = p_13) / n_13

var_1_hat <- p_01_hat * (1 - p_01_hat) / n_11 + p_01_hat * (1 - p_01_hat) / n_01
var_2_hat <- p_02_hat * (1 - p_02_hat) / n_12 + p_02_hat * (1 - p_02_hat) / n_02
var_3_hat <- p_03_hat * (1 - p_03_hat) / n_13 + p_03_hat * (1 - p_03_hat) / n_03

C <- qnorm(1 - alpha) * (1 / 3) * sqrt(var_1_hat + var_2_hat + var_3_hat)

diff <- ((p_11_hat - p_01_hat) + (p_12_hat - p_02_hat) + (p_13_hat - p_03_hat)) / 3
rejected_null <- mean(diff > C)
```

The calculated power is 0.151632. The fraction of rejected simulations is 0.15153. Pretty close.

10 Converting all results for the continuous case

All results in this document are derived for the binary case.

In case our distributions of interest are continuous with means μ_0, μ_1 and variances σ_0^2, σ_1^2 (note that they don't have to be necessarily Gaussian) then all results in this paper can be used, converting in all formulas:

$$p \to \mu$$

$$p \to \mu$$

$$SD(\hat{\mu}_1 - \hat{\mu}_0) \to \sqrt{\sigma_0^2/n_1 + \sigma_0^2/n_0}$$

$$SD(\hat{\mu}_1 - \hat{\mu}_0) \to \sqrt{\sigma_1^2/n_1 + \sigma_0^2/n_0}$$

So for example in the continuous case the test constant C would be

$$\boxed{C = \Phi^{-1}(1-\alpha) \cdot \sqrt{\hat{\sigma}_0^2/n_1 + \hat{\sigma}_0^2/n_0}}$$

```r
rm(list = ls())
mu_0 <- mu_1 <- 10 # Null is true
sigma_0 <- sigma_1 <- 4
n_0 <- n_1 <- 1000
alpha <- 0.05
```

```r
M <- 100000 # number of simulations
mu_0_samples <- replicate(n = M, rnorm(n = n_0, mean = mu_0, sd = sqrt(sigma_0)))
mu_1_samples <- replicate(n = M, rnorm(n = n_1, mean = mu_1, sd = sqrt(sigma_1)))

mu_0_hat <- apply(mu_0_samples, 2, mean)
sigma_0_hat <- apply(mu_0_samples, 2, var)
mu_1_hat <- apply(mu_1_samples, 2, mean)
C <- qnorm(1 - alpha) * sqrt(2 * sigma_0_hat / n_0)

diff <- mu_1_hat - mu_0_hat
rejected_null <- mean(diff > C)
```

10.0.0.1 Simulation validation The fraction of simulations we rejected the null was 0.05086, pretty close to our chosen $\alpha = 0.05$.