Power analysis and bacterial population sizes

Josselin Noirel, Antoine Bridier-Nahmias, Nicolas Godron

2024 – 2025-02-27

```r
library('foreach')
library('doParallel')
library('tidyverse')
library('scales')

library('lemon')
knit_print.data.frame <- lemon_print

theme_set(theme_bw())
set.seed(123)

registerDoParallel(cores=16)
```

Background

We establish a simple, additive genotype-phenotype relationship in a bacterial population based on polymorphic genes. The genes are assumed to be in linkage equilibrium and there is no genetic interaction in our model. Given a certain population size, we simulate data to estimate how many true associations get detected.

Parametrisation of the genotype-phenotype relationship

Let's assume we have $n = 30$ independent causal genes (as there are elements towards polygenic inheritance) among $N = 4000$ (order of magnitude of $ Mycobacterium tuberculosis $ genes) bi-allelic genes.

```r
n = 15            # Number of independent (i.e. in linkage equilibrium) causal loci
N = 4000          # Number of loci to be tested; N >= n (N - n non causal)
```

We'll assume a simple, additive genotype-phenotype relationship in a bacterium. In this context, the genetic architecture is defined by the allele frequency and the penetrance of each polymorphic gene. We assume that there is no linkage We denote each causal gene has a MAF f_i and an effect size β_i.

A simple genetic architecture could assume identical MAFs and identical β_i's.

```r
f = rep(.25, n)    # Minor allele frequency (let's assume constant)
beta = rep(1, n)   # Identical, arbitrary effect size for all genes
```

Here, we will use a genetic architecture that assumes an equally distributed mixture of rare, uncommon and common variants $f_i \in \{1\%, 5\%, 20\%\}$, each with a spectrum of effect sizes: $\beta_i \in [\![1, 2, .., 10]\!]$.

```
f_vector = c(0.01, 0.05, 0.20)
f = rep(f_vector,
        each = n / length(f_vector)) # Rare, uncommon and common variants
beta = rep(1:(n / length(f_vector)),
              times = length(f_vector))        # Mixture of effect sizes
```

Note that the magnitude of the effect sizes β_i's is determined up to a constant multiplier (so that `beta = rep(1, n)` and `beta = rep(10, n)` are equivalent).

We model a quantitative phenotype Y through an additive relationship

$$Y = \sum \beta_i d_i + \epsilon$$

where d_i (0 or 1) denotes the i-th genotype (presence/absence of gene) ϵ is a normal distribution with a mean of zero and variance of σ^2; it captures the part of the phenotype that is determined by unaccounted-for variables (be they genetic, environmental or otherwise) as well as the part of the phenotype that is, for all intents and purposes, purely stochastic.

$$\bar{Y} = \sum \beta_i f_i \quad V(Y) = \sum \beta_i f_i (1 - f_i) + \sigma^2$$

Narrow-sense heritability is given by

$$h^2 = \frac{\sum \beta_i f_i (1 - f_i)}{V(Y)}$$

For this phenotype, we assume a fair amount of heritability $h^2 = 30\%$, to account for host factors such as immune competency (or lack thereof) making up part of the effect. Another assumption is the confounding contribution of all resistances, which here will be set at 5%, so $h^2\_res = 25$. This heritability would gain from managing resistance-induced bias, such as designing a resistance covariate. Furthermore, the heterogeneity of the 'extra-pulmonary' phenotype can be seen as a further loss in heritability (again set at 5%).

```
h2 = 0.30
h2_res = 0.25
h2_extrap = 0.20
```

The amount of noise, σ^2, can be determined based on the assumed value for h^2:

$$\sigma^2 = \frac{1 - h^2}{h^2} \left(\sum \beta_i f_i (1 - f_i) \right)$$

```
vg = sum(beta^2 * f * (1 - f)) # Genetic variation
s2 = (1 - h2)/h2 * vg          # sigma^2
vt = vg + s2                   # Total phenotypic variation
sigma  = sqrt(s2)              # sigma

s2_res = (1- h2_res)/h2_res * vg
vt_res = vg + s2_res
sigma_res  = sqrt(s2_res)

s2_extrap = (1- h2_extrap)/h2_extrap * vg
vt_extrap = vg + s2_extrap                  # Total phenotypic variation
sigma_extrap  = sqrt(s2_extrap)             # sigma
```

Of course, other genetic architectures could be tested:

- 10%, 15%, 40%, 60% heritability
- .02, .1, .3, .4 MAF
- 4, 8, 16, 32, 64 independent causal loci
- Include the alpha model for effect size https://www.nature.com/articles/s41467-019-08424-6 (https://www.nature.com/articles/s41467-019-08424-6)

We assume $K = 407$ for the set of genomes with no resistance and $K\_res = 1000$ for those with or without resistance. A majority of the strains in K_res but not in K would be publicly available genomes from published studies, some of which have already been collected and passed our genomic quality control pipeline. The rest would be newly included samples from patients suffering from meningitis, sequenced by the CNR.

```
K = 407
K_res = 750
K_extrap = 1000
```

Detection is based on a type~I error rate of $\alpha = 0.05\ (5\%)$ or False Discovery Rate $FDR = 50\%$.

```
alpha = .05
fdr_thr = .5
```

Simulations

We perform $R = 10,000$ replicates:

```
R = 10000       # Number of replicates
```

Each replicate will build a matrix of genotypes and a phenotype including noise. Genetic associations are sought using a simple t test; the p value is recorded.

For each simulation, we report:

- How many true hits are retrieved based on a Bonferroni correction
- How many hits are reported using the FDR threshold and
- How many true hits are reported using the FDR threshold
- The minimal and the maximal p values

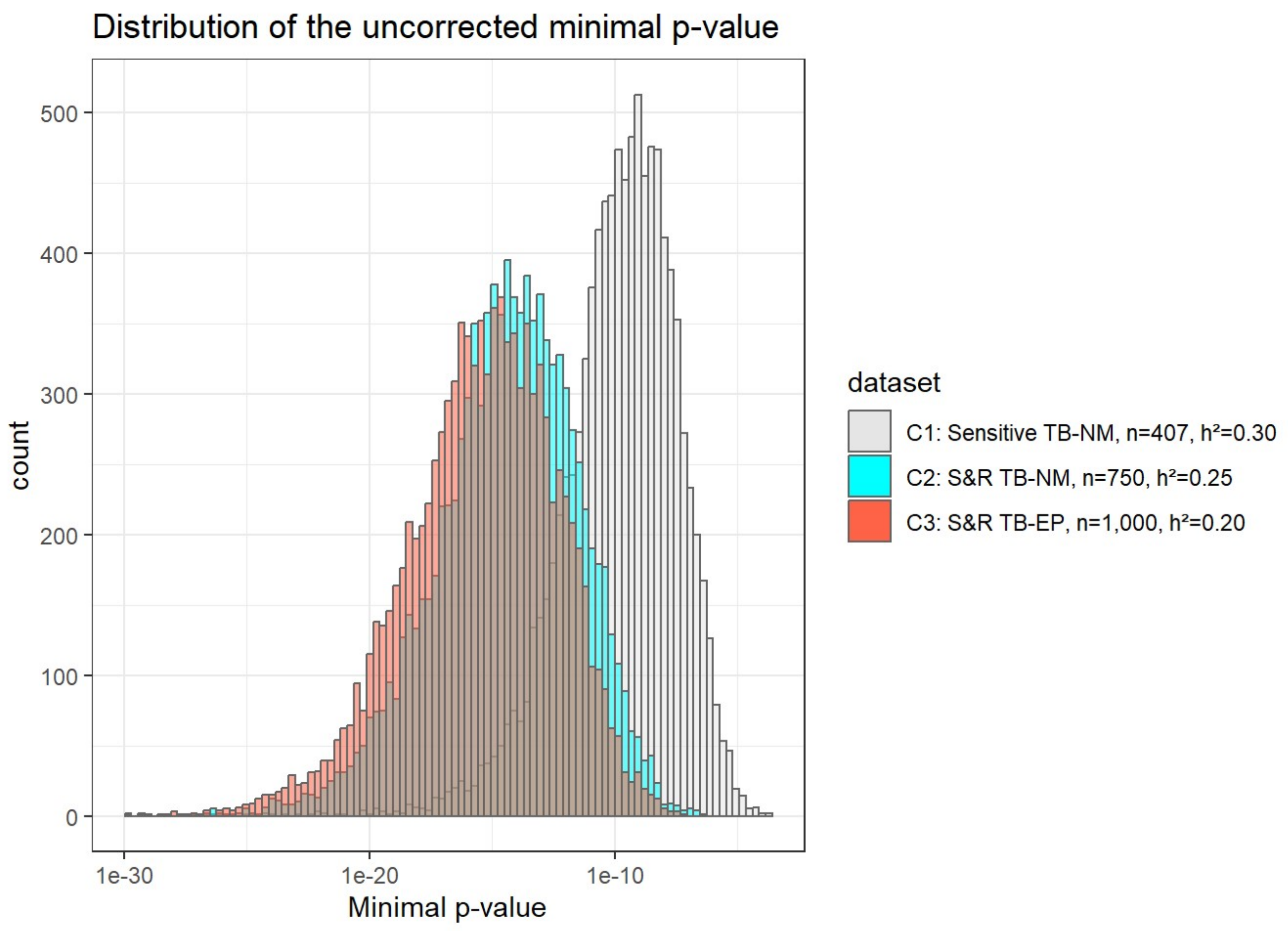
```
p <- numeric(n)  # Vector of n p-values

sim <- foreach(r = 1:R, .combine = rbind) %dopar%
  {
    x = t(matrix(rbinom(n * K, size=1, prob=f), nrow=n))
    yg = colSums(beta * t(x))
    noise = rnorm(K, mean=0, sd=sigma)
    y = yg + noise

    for (j in 1:n) {
        a = y[x[, j] == 0]
        b = y[x[, j] == 1]
        if (length(a) >=2 && length(b) >= 2) {
            test = t.test(a, b)
            p[j] = test$p.value
        } else {
            p[j] = 1
        }

        fdr = p.adjust(c(p, runif(N - n)), method='fdr')

        fdr_true_detected = sum(which(fdr < fdr_thr) <= n)

        fdr_detected = sum(fdr < fdr_thr)
    }

    c(minp=min(p),
      maxp=max(p),
      bonf_true=sum(p < alpha/N),
      fdr_detect=fdr_detected,
      fdr_true=fdr_true_detected,
      h2=var(yg)/var(y))
  }

# Same but with resistance dataset
sim_res <- foreach(r = 1:R, .combine = rbind) %dopar%
  {
    x = t(matrix(rbinom(n * K_res, size=1, prob=f), nrow=n))
    yg = colSums(beta * t(x))
    noise = rnorm(K_res, mean=0, sd=sigma_res)
    y = yg + noise

    for (j in 1:n) {
        a = y[x[, j] == 0]
        b = y[x[, j] == 1]
        if (length(a) >= 2 && length(b) >= 2) {
            test = t.test(a, b)
            p[j] = test$p.value
        } else {
            p[j] = 1
        }

        fdr = p.adjust(c(p, runif(N - n)), method='fdr')
        fdr_true_detected = sum(which(fdr < fdr_thr) <= n)

        fdr_detected = sum(fdr < fdr_thr)
    }

    c(minp=min(p),
      maxp=max(p),
      bonf_true=sum(p < alpha/N),
      fdr_detect=fdr_detected,
      fdr_true=fdr_true_detected,
      h2=var(yg)/var(y))
  }

# Same but with resistance dataset
sim_extrap <- foreach(r = 1:R, .combine = rbind) %dopar%
  {
    x = t(matrix(rbinom(n * K_extrap, size=1, prob=f), nrow=n))
    yg = colSums(beta * t(x))
    noise = rnorm(K_extrap, mean=0, sd=sigma_extrap)
    y = yg + noise

    for (j in 1:n) {
        a = y[x[, j] == 0]
        b = y[x[, j] == 1]
        if (length(a) >= 2 && length(b) >= 2) {
            test = t.test(a, b)
            p[j] = test$p.value
        } else {
            p[j] = 1
        }

        fdr = p.adjust(c(p, runif(N - n)), method='fdr')

        fdr_true_detected = sum(which(fdr < fdr_thr) <= n)

        fdr_detected = sum(fdr < fdr_thr)
    }

    c(minp=min(p),
      maxp=max(p),
      bonf_true=sum(p < alpha/N),
      fdr_detect=fdr_detected,
      fdr_true=fdr_true_detected,
      h2=var(yg)/var(y))
  }
```

```
sim <-
  sim |>
  as_tibble() |>
  mutate(dataset = "C1: Sensitive TB-NM, n=407, h²=0.30")

sim_res <-
  sim_res |>
  as_tibble() |>
  mutate(dataset = "C2: S&R TB-NM, n=750, h²=0.25")

sim_extrap <-
  sim_extrap |>
  as_tibble() |>
  mutate(dataset = "C3: S&R TB-EP, n=1,000, h²=0.20")

sim_all <- rbind(sim, sim_res, sim_extrap)
```

Analysis

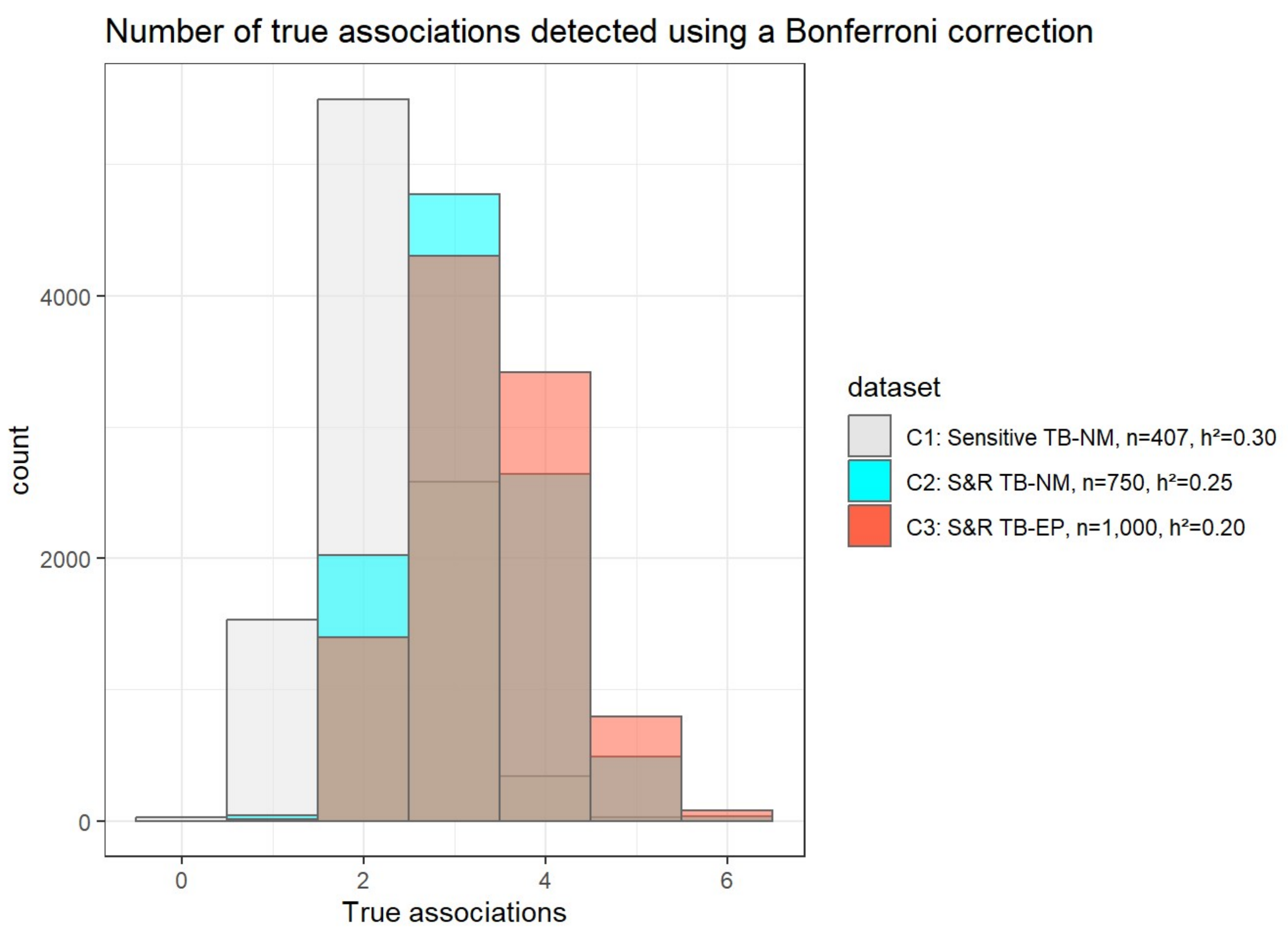
No multiple test correction

```
sim_all |>
  ggplot(aes(minp,
             group = dataset,
             fill = dataset,
             alpha = 0.7)) +
  geom_histogram(bins=100,
                 colour = "grey40",
                 position = "identity") +
  guides(size = "legend", alpha = "none") +
  scale_x_log10(limits = c(1e-30, NA),
                breaks = c(1e-2, 1e-10, 1e-20, 1e-30)) +
  scale_fill_manual(values = c("grey90", "cyan", "tomato")) +
  xlab("Minimal p-value") +
  ggtitle("Distribution of the uncorrected minimal p-value")
```



Bonferroni

```
sim_all |>
  as_tibble() |>
  ggplot(aes(bonf_true,
             colour = dataset,
             fill = dataset,
             alpha = 0.5)) +
  geom_histogram(binwidth=1,
                 colour = "grey40",
                 position = "identity") +
  guides(size = "legend", alpha = "none") +
  scale_x_continuous(breaks = seq(0, max(sim_all$bonf_true), 2)) +
  scale_fill_manual(values = c("grey90", "cyan", "tomato")) +
  labs(seq(0, max(sim_all$bonf_true), 2)) +
  xlab("True associations") +
  ggtitle('Number of true associations detected using a Bonferroni correction')
```



Here is a table of the minimal number of true associations that are likely to be found using Bonferroni:

```
sim_all |>
    as_tibble() |>
    group_by(dataset) |>
    count(bonf_true) |>
    arrange(desc(bonf_true)) |>
    mutate(cum_pc=signif(100 * cumsum(n/sum(n)),
                         digits=2)) |>
    arrange(dataset,
            desc(cum_pc),
            bonf_true) |>
  print(n = 20)
```

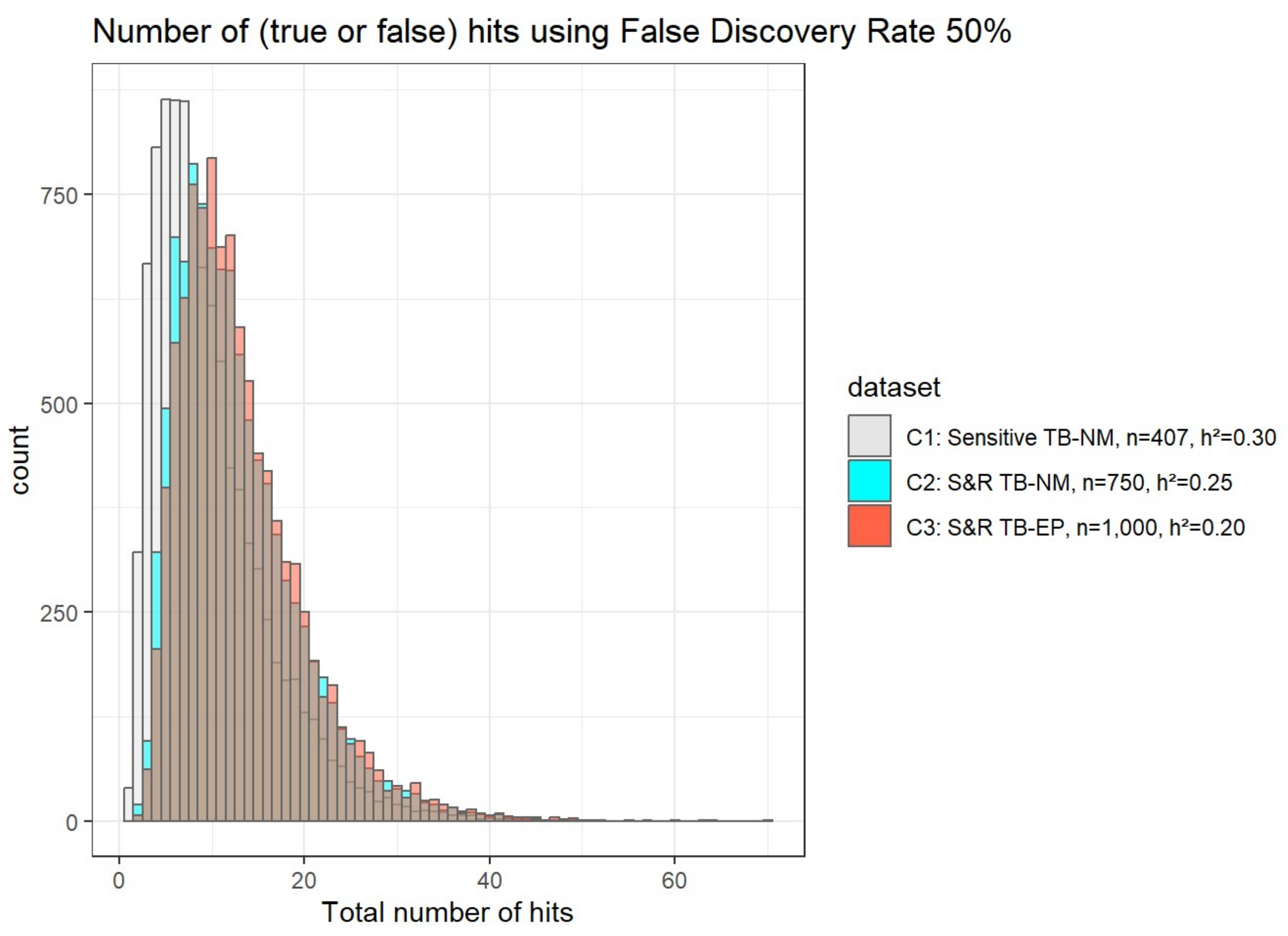
```
# A tibble: 18 × 4
# Groups:   dataset [3]
   dataset                               bonf_true     n cum_pc
   <chr>                                     <dbl> <int>  <dbl>
 1 C1: Sensitive TB-NM, n=407, h²=0.30         0    28 100
 2 C1: Sensitive TB-NM, n=407, h²=0.30         1  1532 100
 3 C1: Sensitive TB-NM, n=407, h²=0.30         2  5498  84
 4 C1: Sensitive TB-NM, n=407, h²=0.30         3  2580  29
 5 C1: Sensitive TB-NM, n=407, h²=0.30         4   336   3.6
 6 C1: Sensitive TB-NM, n=407, h²=0.30         5    26   0.26
 7 C2: S&R TB-NM, n=750, h²=0.25               1    42 100
 8 C2: S&R TB-NM, n=750, h²=0.25               2  2025 100
 9 C2: S&R TB-NM, n=750, h²=0.25               3  4772  79
10 C2: S&R TB-NM, n=750, h²=0.25               4  2640  32
11 C2: S&R TB-NM, n=750, h²=0.25               5   489   5.2
12 C2: S&R TB-NM, n=750, h²=0.25               6    32   0.32
13 C3: S&R TB-EP, n=1,000, h²=0.20             1    13 100
14 C3: S&R TB-EP, n=1,000, h²=0.20             2  1396 100
15 C3: S&R TB-EP, n=1,000, h²=0.20             3  4304  86
16 C3: S&R TB-EP, n=1,000, h²=0.20             4  3415  43
17 C3: S&R TB-EP, n=1,000, h²=0.20             5   795   8.7
18 C3: S&R TB-EP, n=1,000, h²=0.20             6    77   0.77
```

Note: The cum_pc column can be interpreted as the percentage of replicates having at least “bonf_true” true associations

False Discovery Rate

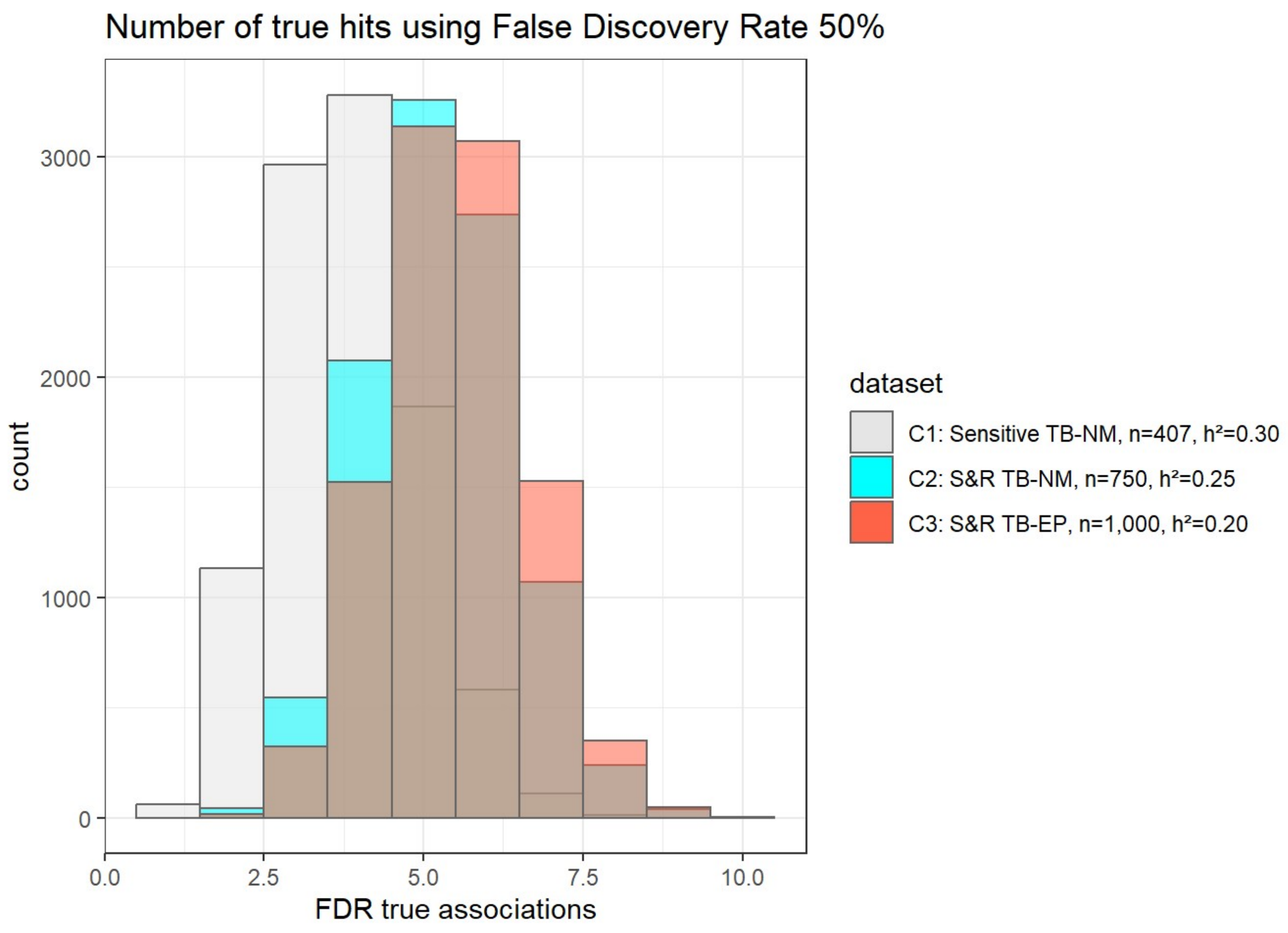
How many hits, in total (true or false positives), would come out of the FDR analysis:

```
sim_all |>
  as_tibble() |>
  ggplot(aes(fdr_detect,
             fill = dataset,
             alpha = 0.7)) +
  geom_histogram(binwidth=1,
                 colour = "grey40",
                  position="identity") +
  guides(size = "legend", alpha = "none") +
  scale_fill_manual(values = c("grey90", "cyan", "tomato")) +
  xlab("Total number of hits") +
  ggtitle("Number of (true or false) hits using False Discovery Rate 50%")
```



How many true hits would come out of the FDR analysis:

```
sim_all |>
  as_tibble() |>
  ggplot(aes(fdr_true,
             fill = dataset,
             alpha = 0.5)) +
  geom_histogram(binwidth=1,
                 colour = "grey40",
                 position = "identity") +
  guides(size = "legend", alpha = "none") +
  scale_fill_manual(values = c("grey90", "cyan", "tomato")) +
  xlab("FDR true associations") +
  ggtitle("Number of true hits using False Discovery Rate 50%")
```



```
sim_all |>
  as_tibble() |>
  group_by(dataset) |>
  count(fdr_true) |>
  arrange(desc(fdr_true)) |>
  mutate(cum_pc=signif(100 * cumsum(n/sum(n)), digits=2)) |>
  arrange(dataset,
          desc(cum_pc),
          fdr_true) |>
  print(n = 50)
```

```
# A tibble: 26 × 4
# Groups:   dataset [3]
   dataset                             fdr_true     n cum_pc
   <chr>                                  <dbl> <int>  <dbl>
 1 C1: Sensitive TB-NM, n=407, h²=0.30        1    61 100
 2 C1: Sensitive TB-NM, n=407, h²=0.30        2  1130  99
 3 C1: Sensitive TB-NM, n=407, h²=0.30        3  2963  88
 4 C1: Sensitive TB-NM, n=407, h²=0.30        4  3281  58
 5 C1: Sensitive TB-NM, n=407, h²=0.30        5  1865  26
 6 C1: Sensitive TB-NM, n=407, h²=0.30        6   579   7
 7 C1: Sensitive TB-NM, n=407, h²=0.30        7   110   1.2
 8 C1: Sensitive TB-NM, n=407, h²=0.30        8    11   0.11
 9 C2: S&R TB-NM, n=750, h²=0.25              2    44 100
10 C2: S&R TB-NM, n=750, h²=0.25              3   544 100
11 C2: S&R TB-NM, n=750, h²=0.25              4  2074  94
12 C2: S&R TB-NM, n=750, h²=0.25              5  3255  73
13 C2: S&R TB-NM, n=750, h²=0.25              6  2737  41
14 C2: S&R TB-NM, n=750, h²=0.25              7  1068  13
15 C2: S&R TB-NM, n=750, h²=0.25              8   239   2.8
16 C2: S&R TB-NM, n=750, h²=0.25              9    38   0.39
17 C2: S&R TB-NM, n=750, h²=0.25             10     1   0.01
18 C3: S&R TB-EP, n=1,000, h²=0.20            2    17 100
19 C3: S&R TB-EP, n=1,000, h²=0.20            3   321 100
20 C3: S&R TB-EP, n=1,000, h²=0.20            4  1521  97
21 C3: S&R TB-EP, n=1,000, h²=0.20            5  3139  81
22 C3: S&R TB-EP, n=1,000, h²=0.20            6  3072  50
23 C3: S&R TB-EP, n=1,000, h²=0.20            7  1527  19
24 C3: S&R TB-EP, n=1,000, h²=0.20            8   351   4
25 C3: S&R TB-EP, n=1,000, h²=0.20            9    49   0.52
26 C3: S&R TB-EP, n=1,000, h²=0.20           10     3   0.03
```