MHT Simulation

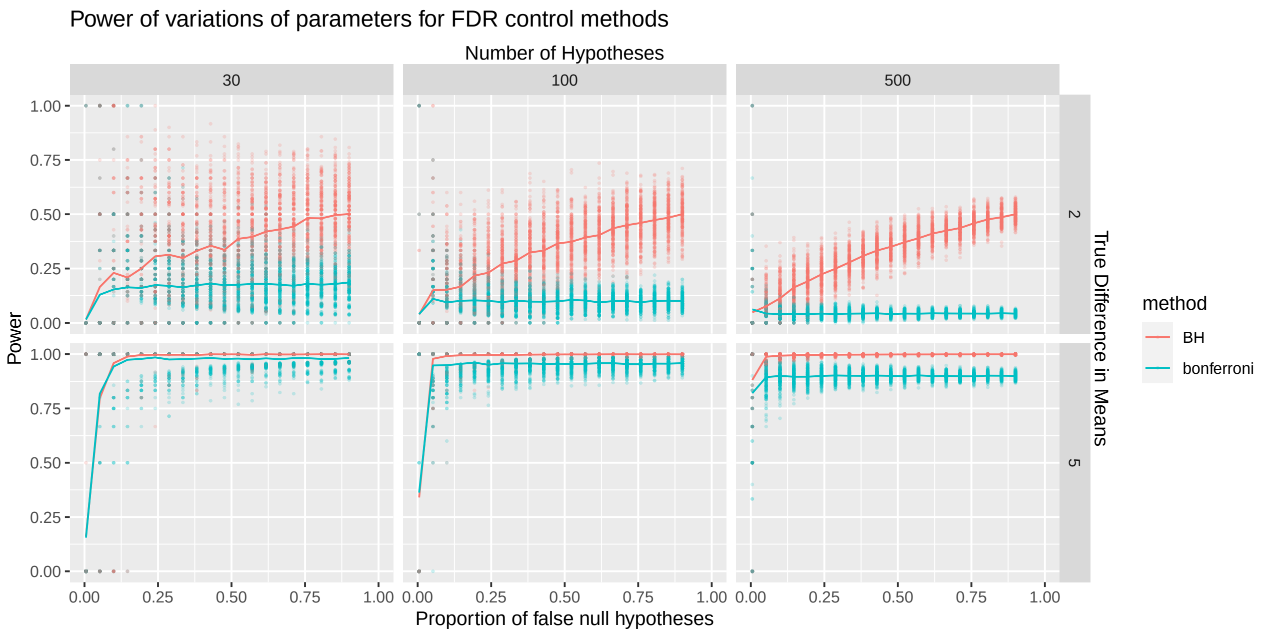
Connor Demorest

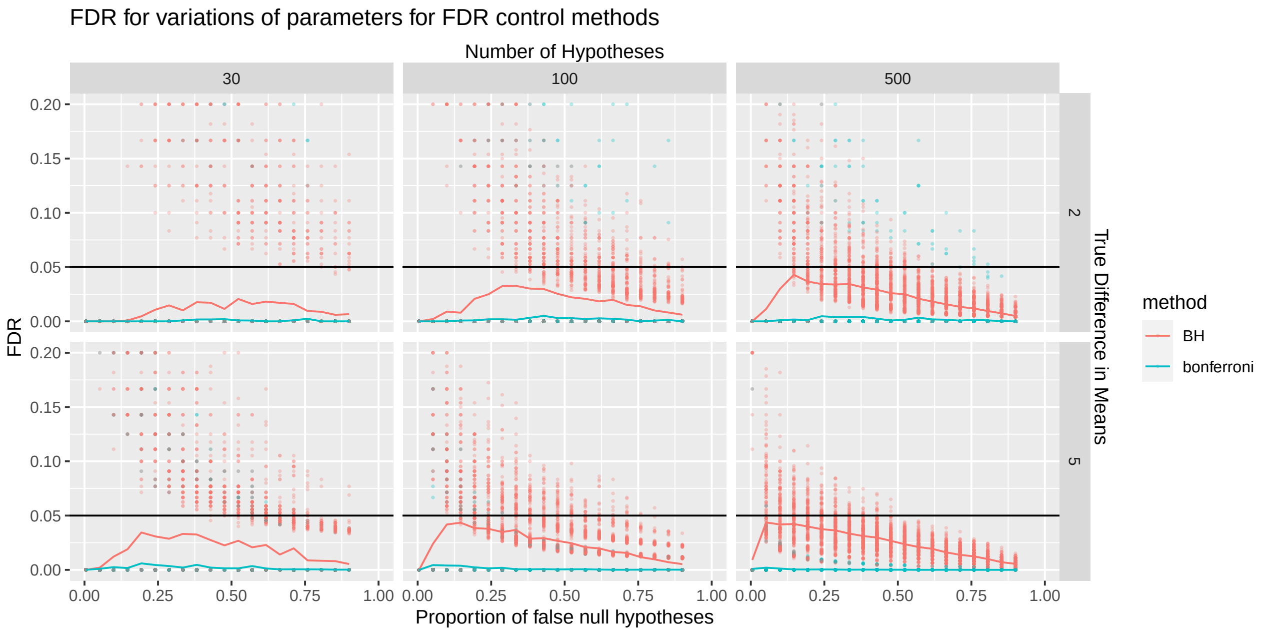
9/4/2021

Simulation study:

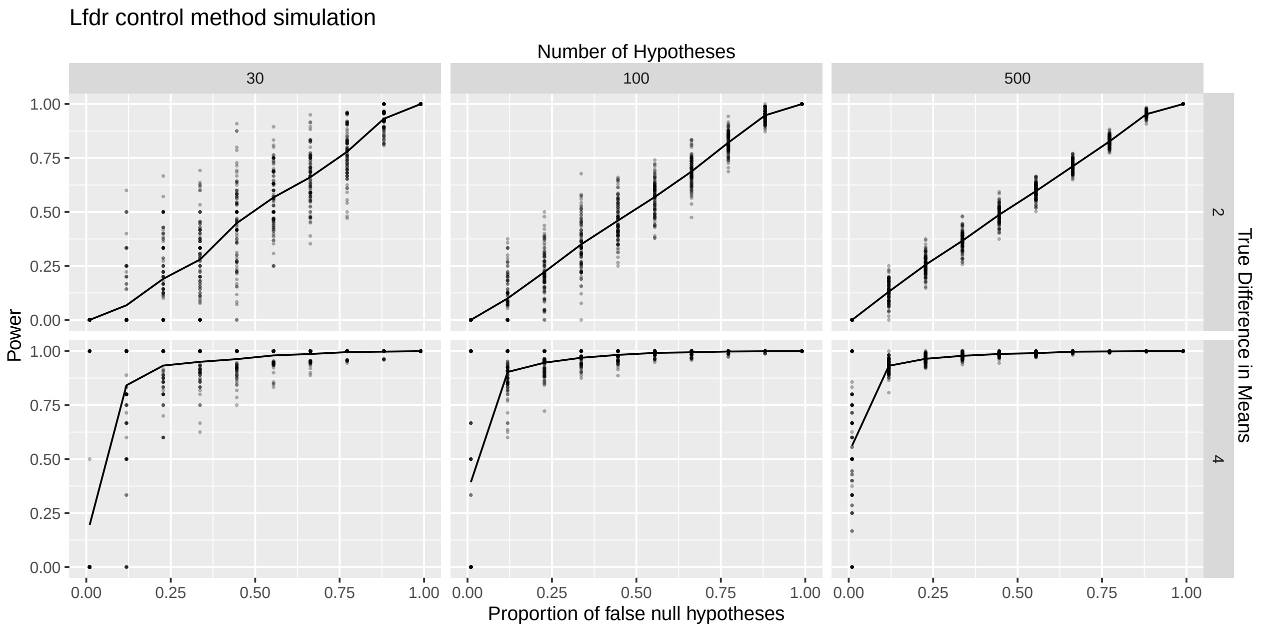
1) Take 100 samples from N(0,1) and N(5,1) mixed with proportion p in (0,1). Take $\alpha = 0.05$. $X_i = (1-p) * N(0,1) + p * N(3,1)$
2) Find p values of X
3) Use BH, etc
4) Get FDP and Power
5) Repeat many times to find FDR and avg power across values of p

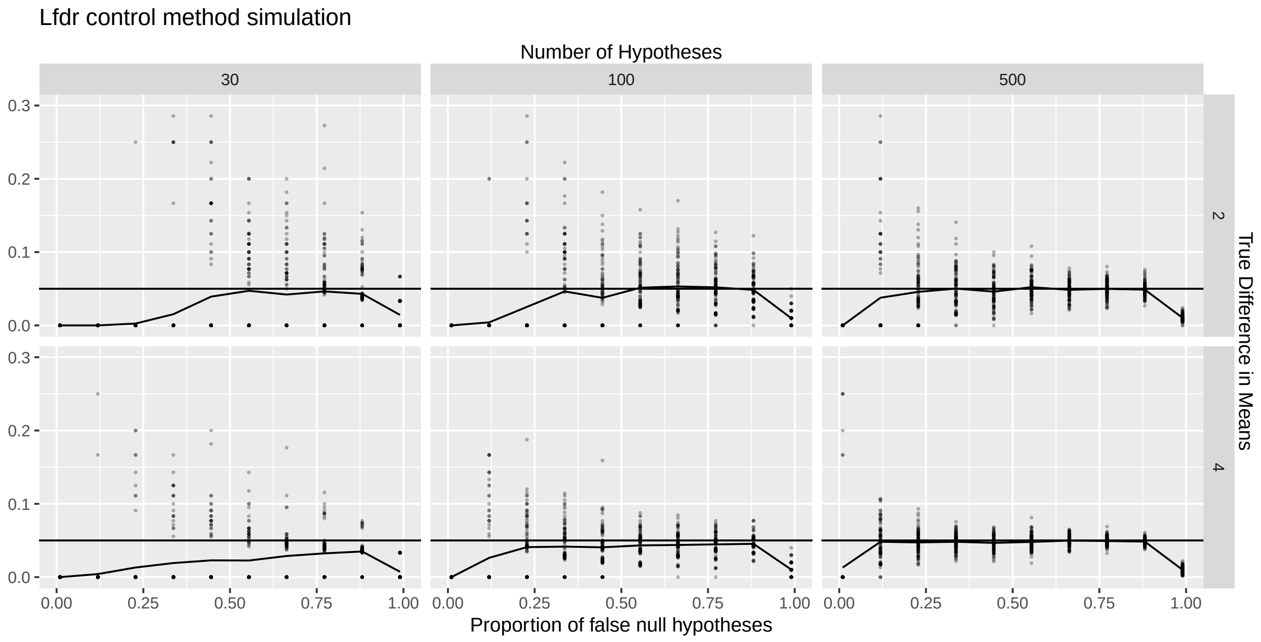
As a function:





Using Lfdr and PFDR





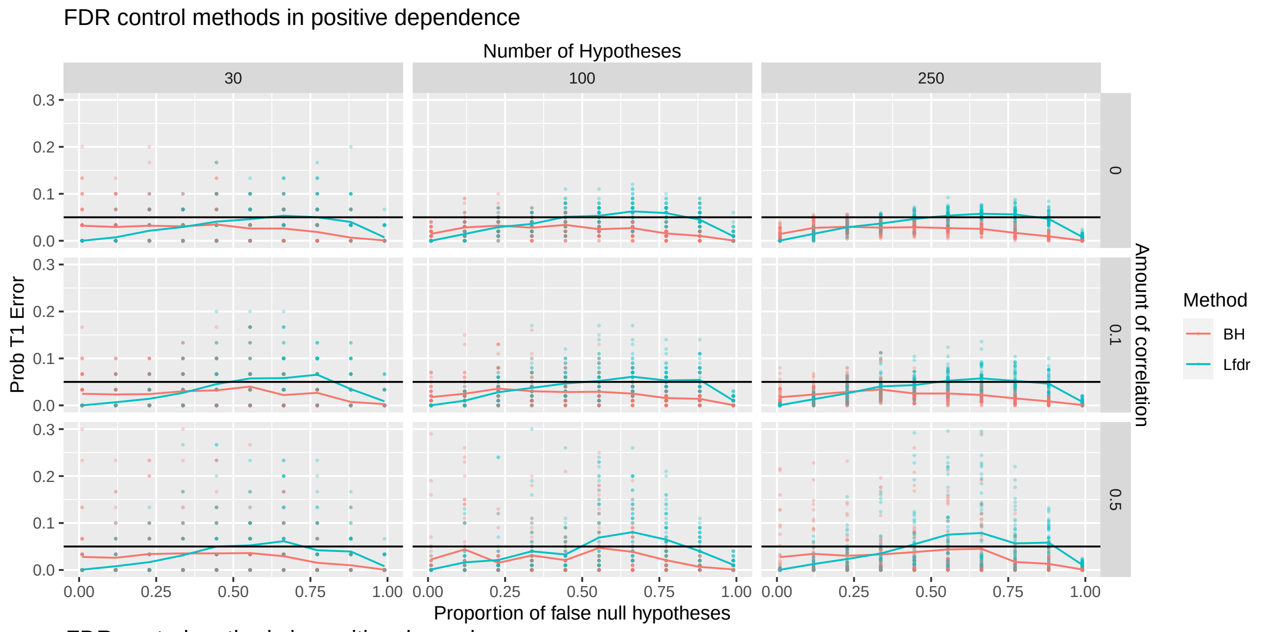
Using the same setting as the Lfdr procedure, compare the performance of the Lfdr and the BH procedure,

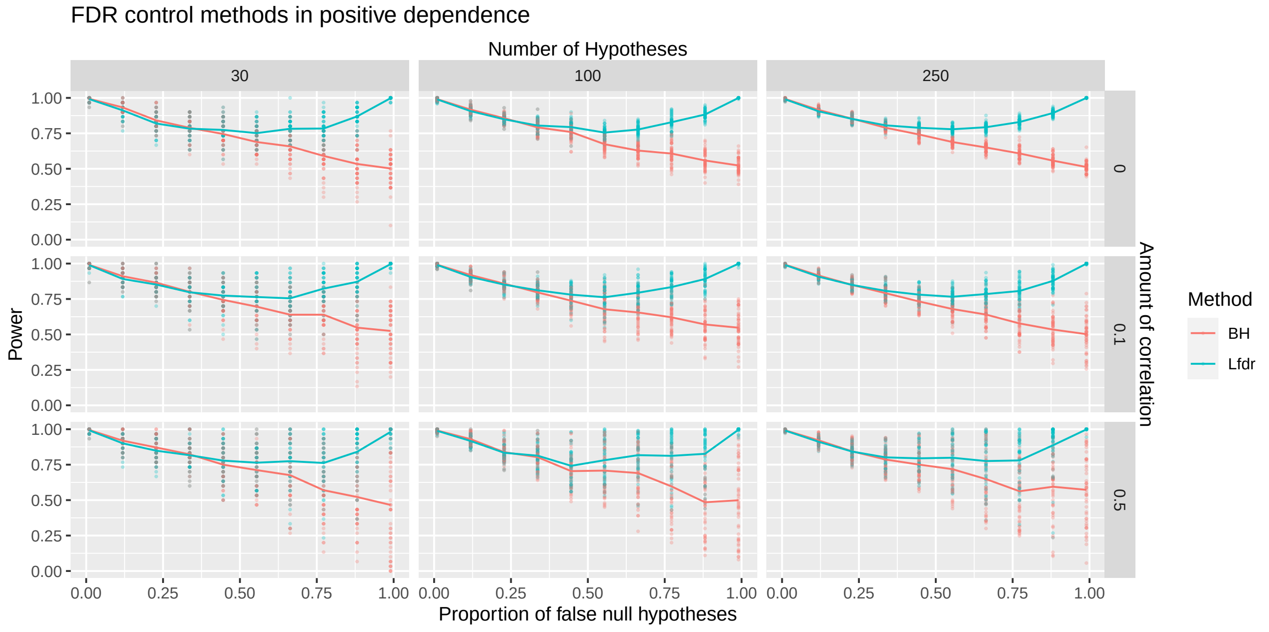
a) When the variances are same, i.e., 1 in both null and non-null groups -> *see above*

b) Draw the samples from a multivariate normal distribution. For example, if you have 100 samples, out of which $\pi = 0.5$ (say), proportion are from the non-null distribution, draw one sample from the following: Define mu_vector = (1-theta)f(0) + (theta)f(2), i.e. a 100x1 vector of 0 (null mean) and 2(non-null mean). Sigma = a 100x100 matrix, with diagonal elements 1 and off diagonal elements = rho (a positive value between 0 and 1). Draw one sample from MVNorm_100(μ = mu_vector, σ = Sigma) Find the p-values, and Apply the BH and Lfdr process on this sample to figure out the discoveries, false and true discoveries. This would show the performance of the two procedures when the p-values are positively dependent.

```
## Warning: Removed 19 rows containing non-finite values (stat_summary).
```

```
## Warning: Removed 19 rows containing missing values (geom_point).
```





Should I have been doing two-sided tests?

Code:

```
knitr::opts_chunk$set(echo = F, fig.width = 10, fig.height = 5,
                      cache = TRUE)
library(tidyverse)
p = 0.4
a = 0.05
num_samples = 100
# True if true mean =/= 0
A = rbernoulli(num_samples, p)
t_nulls = rnorm(num_samples, 0, 1)
```

```r
f_nulls = rnorm(num_samples, 2, 1)
samples = ifelse(A, f_nulls, t_nulls)
p_vals = pnorm(samples, lower.tail = F)

# Test discoveries
B = (p_vals <= a)
# FP = A' & B
## False_positive = (A & B)
# FDP = P(A' & B)/P(B)
FDP = ifelse(is.na(sum(!A & B)/sum(B)), 0, sum(!A & B)/sum(B))

# Power = P(reject Null | Null false) = P(reject & Null false)/P(Null false)
Power = ifelse(is.na(sum(A & B)/sum(A)), 0, sum(A & B)/sum(A))
# m === number of total hypotheses
func = function(p = 0.4, a = 0.05, m = 1000, alt_hyp = 3, method = "none") {
  # A === True discoveries
  A = rbernoulli(m, p)
  t_nulls = rnorm(m, 0, 1)
  f_nulls = rnorm(m, alt_hyp, 1)
  samples = ifelse(A, f_nulls, t_nulls)
  p_vals = pnorm(samples, lower.tail = F)
  adj_p_vals = p.adjust(p = p_vals, method = method)

  # Test discoveries
  B = (adj_p_vals <= a)
  # FP = A' & B
  False_positive = ifelse(!A & B, 1, 0)
  # FDP = P(A' & B)/P(B)
  FDP = ifelse(is.na(sum(!A & B)/sum(B)), 0, sum(!A & B)/sum(B))

  # Power = P(reject Null | Null false) = P(reject & Null false)/P(Null false)
  Power = ifelse(is.na(sum(A & B)/sum(A)), 0, sum(A & B)/sum(A))
  return(c(FDP, Power))
}
# New function: do.many, where it just does 'many' and returns averages?
# Calculate FDR and Avg Power

# many = replicate(1000, func(method = "none")) %>% apply(., 1, FUN = mean) %>% print

df = expand_grid(p = rep(seq(0.005, 0.9, length.out = 20), each = 200),
                 method = c("bonferroni", "BH"),
                 alt_hyp = c(2,5),
                 m = c(30, 100, 500))
many = t(mapply(func,
                p = df$p,
                method = df$method,
                alt_hyp = df$alt_hyp,
                m = df$m)) %>%
  as_tibble %>%
  rename(FDR = V1, Power = V2) %>%
  mutate(df)

# do_many = function(num_reps = 1000, p = 0.4, a = 0.05, num_samples = 1000, method = "none") {
```

```r
#   many = replicate(num_reps, func(p = p, a = a, num_samples = num_samples, method = method)) %>%
#     apply(., 1, FUN = mean)
#   return(many)
# }
#
# df = expand_grid(p = seq(0.1, 0.9, by = 0.01), method = c("none", "BH"))
# df2 = t(mapply(do_many, p = df$p, method = df$method, num_reps = 100)) %>%
#   as_tibble %>%
#   rename(FDR = V1, Power = V2) %>%
#   mutate(df)

# Increased p gives more true discoveries
ggplot(data = many, aes(x = p, y = Power, color = method)) +
  geom_point(size = 0.3, alpha = 0.2) +
  stat_summary(fun = "mean", geom = "line") +
  facet_grid(alt_hyp ~ m) +
  labs(x = "Proportion of false null hypotheses",
       title = "Power of variations of parameters for FDR control methods") +
  scale_y_continuous(sec.axis = sec_axis(~ . , name = "True Difference in Means",
                                          breaks = NULL, labels = NULL), limits = c(0,1)) +
  scale_x_continuous(sec.axis = sec_axis(~ . , name = "Number of Hypotheses",
                                          breaks = NULL, labels = NULL), limits = c(0,1))

ggplot(data = many, aes(x = p, y = FDR, color = method)) +
  geom_point(size = 0.3, alpha = 0.3) +
  stat_summary(fun = "mean", geom = "line") +
  facet_grid(alt_hyp ~ m) +
  geom_abline(intercept = 0.05, slope = 0) +
  labs(x = "Proportion of false null hypotheses",
       title = "FDR for variations of parameters for FDR control methods") +
  scale_y_continuous(sec.axis = sec_axis(~ . , name = "True Difference in Means",
                                          breaks = NULL, labels = NULL), limits = c(0,0.2)) +
  scale_x_continuous(sec.axis = sec_axis(~ . , name = "Number of Hypotheses",
                                          breaks = NULL, labels = NULL), limits = c(0,1))
# Basically re-using code from above
# p known here!
p = 0.4
alpha = 0.05
num_samples = 1000
## True if true mean =/= 0
# theta = rbernoulli(num_samples, p)
# f0 = rnorm(num_samples, 0, 1)
# f1 = rnorm(num_samples, 5, 1)
# X = ifelse(theta, f1, f0)
# f = (1-p)*dnorm(X, 0, 1) + p*dnorm(X, 5, 1)

# Lfdr = (1-pi)f0(X)/f(x)
PFDR_fun = function(p = 0.4, alpha = 0.05, num_samples = 1000, alt_hyp = 4) {
  df = tibble(theta = rbernoulli(num_samples, p),
              X = ifelse(theta, rnorm(num_samples, alt_hyp, 1), rnorm(num_samples, 0, 1)),
              f = (1-p)*dnorm(X, 0, 1) + p*dnorm(X, alt_hyp, 1),
              Lfdr = (1-p)*dnorm(X, 0, 1)/f) %>%
    arrange(Lfdr) %>%
    mutate(q = cumsum(Lfdr)/(1:length(Lfdr)),
           `q<a` = q < alpha, # Total discoveries
           Type1error = `q<a`&!theta, # Type 1 errors
           Type2error = theta&!`q<a`) %>% # Type 2 errors
    summarise(theta = sum(theta), disc = sum(`q<a`), t1 = sum(Type1error), t2 = sum(Type2error), p, alt_
  return(df)
}
# PFDR_fun()
# reps = plyr::rdply(100, PFDR_fun(), .id = NULL) %>% summarise(PFDR = t1/disc)
#
# mean(as.matrix(reps))

#### Old code ####
# PFDR = rep(NA, length(Lfdr))
# Q = rep(NA, length(Lfdr))
# for(t in 1:length(Lfdr)) {
#   Q[t] = (sum(sort(Lfdr)[1:t]/t))# %>% print
#   PFDR[t] = ((sum(sort(Lfdr)[1:t]/t)) < alpha)# %>% print
# }
#####

pr = seq(0.01, 0.99, length.out = 10)
alt = c(2, 4)
samps = c(30, 100, 500)
pr_alt_samps = expand.grid(pr, alt, samps)

# If no discoveries, then no type 1 errors and fdr = 0
reps = replicate(100,
                 mapply(PFDR_fun,
                        p = pr_alt_samps[,1],
                        alt_hyp = pr_alt_samps[,2],
                        num_samples = pr_alt_samps[,3]) %>% t(),
                 simplify = F) %>%
  do.call(rbind, .) %>%
  as_tibble() %>%
  mutate(across(.col = everything(), as.double),
         PFDR = ifelse(is.na(t1/disc), 0, t1/disc),
         Power = 1 - ifelse(is.na(t2/theta), 1, t2/theta))

# ### Make plots! Change values of p, 10 are plenty
# ggplot(reps, mapping = aes(x = p, y = PFDR)) +
#   geom_point(size = 0.3, alpha = 0.3, mapping = aes(color = "FDR")) +
#   geom_hline(yintercept = alpha, color = "blue") +
#   stat_summary(fun = "mean", geom = "line", mapping = aes(color = "FDR")) +
#   geom_point(size = 0.3, alpha = 0.3,
#              mapping = aes(x = p, y = Power, color = "Power")) +
#   stat_summary(fun = "mean", geom = "line",
#                mapping = aes(x = p, y = Power, color = "Power")) +
#   facet_grid(alt_hyp ~ num_samples) +
#   geom_abline(intercept = 0.05, slope = 0) +
#   labs(x = "Proportion of false null hypotheses",
#        title = "Lfdr control method simulation",
#        y = "",
```

```r
#        color = "") +
#   scale_y_continuous(sec.axis = sec_axis(~ . , name = "True Difference in Means",
#                                          breaks = NULL, labels = NULL), limits = c(0,1)) +
#   scale_x_continuous(sec.axis = sec_axis(~ . , name = "Number of Hypotheses",
#                                          breaks = NULL, labels = NULL), limits = c(0,1))

ggplot(reps, mapping = aes(x = p, y = Power)) +
  # geom_point(size = 0.3, alpha = 0.3, mapping = aes(color = "FDR")) +
  # geom_hline(yintercept = alpha, color = "blue") +
  # stat_summary(fun = "mean", geom = "line", mapping = aes(color = "FDR")) +
  geom_point(size = 0.3, alpha = 0.3) +
  stat_summary(fun = "mean", geom = "line") +
  facet_grid(alt_hyp ~ num_samples) +
  labs(x = "Proportion of false null hypotheses",
       title = "Lfdr control method simulation",
       y = "Power") +
  scale_y_continuous(sec.axis = sec_axis(~ . , name = "True Difference in Means",
                                         breaks = NULL, labels = NULL), limits = c(0,1)) +
  scale_x_continuous(sec.axis = sec_axis(~ . , name = "Number of Hypotheses",
                                         breaks = NULL, labels = NULL), limits = c(0,1))

### Make plots! Change values of p, 10 are plenty
ggplot(reps, mapping = aes(x = p, y = PFDR)) +
  geom_point(size = 0.3, alpha = 0.3) +
  geom_hline(yintercept = alpha, color = "blue") +
  stat_summary(fun = "mean", geom = "line") +
  facet_grid(alt_hyp ~ num_samples) +
  geom_abline(intercept = 0.05, slope = 0) +
  labs(x = "Proportion of false null hypotheses",
       title = "Lfdr control method simulation",
       y = "",
       color = "") +
  scale_y_continuous(sec.axis = sec_axis(~ . , name = "True Difference in Means",
                                         breaks = NULL, labels = NULL), limits = c(0,0.3)) +
  scale_x_continuous(sec.axis = sec_axis(~ . , name = "Number of Hypotheses",
                                         breaks = NULL, labels = NULL), limits = c(0,1))
# p = 0.4
# num_samples = 100
# theta = rbernoulli(num_samples, p)
# f0 = rnorm(num_samples, 0, 1)
# f1 = rnorm(num_samples, 2, 1)
# mu_vec = ifelse(theta, f1, f0)
# rho = 0.2
# Sigma = matrix(nrow = num_samples, ncol = num_samples, data = rho)
# diag(Sigma) = 1
#
# MVN_100 = MASS::mvrnorm(n = 1, mu = mu_vec, Sigma = Sigma)
# MVN_p = pnorm(MVN_100, mean = 0, sd = 1, lower.tail = F)
# BH_p = p.adjust(MVN_p, method = "BH")
#
# df = tibble(theta, BH_p,
#             mu_vec,
#             f = (1-p)*dnorm(mu_vec, 0, 1) + p*dnorm(mu_vec, 2, 1),
#             Lfdr = (1-p)*dnorm(mu_vec, 0, 1)/f) %>%
#   arrange(Lfdr) %>%
#   mutate(q = cumsum(Lfdr)/(1:length(Lfdr)),
#          `q<a` = q < alpha, # Total discoveries
#          Type1error = `q<a`&!theta, # Type 1 errors
#          Type2error = theta&!`q<a`)

# Takes in rho, N, proportions, and the alternative hypothesis
# Outputs the T1 and T2 error rates for BH and Lfdr
BH_lfdr_fun = function(num_samples = 100, p = 0.4, rho = 0, alt_hyp = 3) {
  # Sigma has correlation structure rho and 1 on diagonal
  Sigma = matrix(nrow = num_samples, ncol = num_samples, data = rho)
  diag(Sigma) = 1
  # For known p, simulate the true hypothesis state
  theta = rbernoulli(num_samples, p)
  # Maybe I don't need these next three lines?
  f0 = rnorm(num_samples, 0, 1)
  f1 = rnorm(num_samples, alt_hyp, 1)
  mu_vec = ifelse(theta, f1, f0)
  # Generate positively correlated data
  MVN = MASS::mvrnorm(n = 1, mu = mu_vec, Sigma = Sigma)
  # Before, I was using the wrong values for MVN below here! Eek!
  # Get p-values for BH and adjust them
  MVN_p = pnorm(MVN, mean = 0, sd = 1, lower.tail = F)
  BH_p = p.adjust(MVN_p, method = "BH")
  # Copy/paste code from before to get Lfdr
  df = tibble(theta,
              BH_p,
              f = (1-p)*dnorm(MVN, 0, 1) + p*dnorm(MVN, alt_hyp, 1),
              Lfdr = (1-p)*dnorm(MVN, 0, 1)/f) %>%
    arrange(Lfdr) %>%
    mutate(q = cumsum(Lfdr)/(1:length(Lfdr)),
           `q<a` = q < 0.05,
           # Calculate errors for BH and Lfdr methods
           Lfdr_Type1error = `q<a`&!theta,
           Lfdr_Type2error = theta&!`q<a`,
           BH_Type1error = (BH_p < 0.05) & !theta,
           BH_Type2error = theta & !(BH_p < 0.05))
  # Return df of mean for Lfdr and BH error rates
  # Fancy version of apply if you're not exposed to map before
  return(map_df(df[, 7:10], mean))
}

df = expand.grid(p = rep(seq(0.01, 0.99, length.out = 10), times = 50),
                 num_samples = rep(c(30, 100, 250), times = 1),
                 rho = rep(c(0, 0.1, 0.5), times = 1))

# Slow as heck without running in parallel
parallel::mcmapply(FUN = BH_lfdr_fun,
                   mc.cores = parallel::detectCores(),
                   p = df$p,
                   num_samples = df$num_samples,
                   rho = df$rho,
```

```r
                   alt_hyp = 2) -> a

# Move data around into a nice way to plot using ggplot
BH_Lfdr = t(a) %>%
  as_tibble(column_name = rownames(a)) %>%
  cbind(df) %>%
  mutate_all(as.double) %>%
  pivot_longer(cols = ends_with("error"),
               names_to = c("Method", "Error"),
               names_sep = "_",
               values_to = "Prob") %>%
  mutate(Error = str_sub(Error, 1, 5))

# Plot things
ggplot(BH_Lfdr %>% filter(Error == "Type1"),
       mapping = aes(x = p, y = Prob, color = Method)) +
  geom_point(size = 0.3, alpha = 0.3) +
  stat_summary(fun = "mean", geom = "line") +
  geom_hline(yintercept = 0.05) +
  facet_grid(rho ~ num_samples) +
  labs(x = "Proportion of false null hypotheses",
       title = "FDR control methods in positive dependence",
       y = "Prob T1 Error") +
  scale_y_continuous(sec.axis = sec_axis(~ . , name = "Amount of correlation",
                                         breaks = NULL, labels = NULL), limits = c(0,0.3)) +
  scale_x_continuous(sec.axis = sec_axis(~ . , name = "Number of Hypotheses",
                                         breaks = NULL, labels = NULL), limits = c(0,1))

ggplot(BH_Lfdr %>% filter(Error == "Type2"),
       mapping = aes(x = p, y = 1-Prob, color = Method)) +
  geom_point(size = 0.3, alpha = 0.3) +
  stat_summary(fun = "mean", geom = "line") +
  facet_grid(rho ~ num_samples) +
  labs(x = "Proportion of false null hypotheses",
       title = "FDR control methods in positive dependence",
       y = "Power") +
  scale_y_continuous(sec.axis = sec_axis(~ . , name = "Amount of correlation",
                                         breaks = NULL, labels = NULL), limits = c(0,1)) +
  scale_x_continuous(sec.axis = sec_axis(~ . , name = "Number of Hypotheses",
                                         breaks = NULL, labels = NULL), limits = c(0,1))
```