ANCOVA Comparison Simulations: Nonlinear Model

Kellie Ottoboni

2017-10-09

The potential outcomes model we used before turned out to be a linear function of X and Z, conditional on the latent variable v. For these simulations, we changed the outcome model to be nonlinear. Instead of an additive treatment effect, now the effect is multiplicative. We generated the latent variable and errors in the same way, but let

$$X_{ij} = \frac{e^{v_{ij}} + e^{v_{ij}/2}}{2} + \varepsilon_{ij}$$
$$Y_{ij}(Z_{ij}) = (1+\gamma)^{Z_{ij}} X_{ij} + \delta ij.$$

The treatment effect for individual (i, j) is thus γX_{ij}.

Data-generation, tests, and plotting functions

```
gen_y <- function(gamma, x, error, Z) {
    multiplier <- (1 + gamma)^Z
    y <- multiplier * x + error
    return(y)
}

gen_x <- function(gamma, v, error) {
    x <- 0.5 * (exp(v) + exp(v/2)) + error
    return(x)
}

generate_simulated_data <- function(gamma, effect, errors, n = c(16,
    16, 16)) {
    # Input: gamma = multiplier for the magnitude of the
    # treatment effect effect = 'same effect' or 'heterogeneous'
    # errors = 'normal' or 'heavy' n = number of individuals at
    # each stratum Returns: a dataframe containing columns named
    # Y1 (response), Y0 (baseline), Z (treatment), gamma_vec
    # (treatment effect per individual), stratumID (stratum),
    # stratum_effect (beta coefficient per individual), and
    # epsilon (errors)

    stratumID <- rep(1:3, times = n)
    N <- sum(n)

    # What is the treatment effect?
    if (effect == "same effect") {
        v <- runif(N, min = -4, max = 4)
    } else if (effect == "heterogeneous") {
        v <- rep(0, N)
```

```
        v[stratumID == 1] <- runif(n[1], min = -4, max = -1)
        v[stratumID == 2] <- runif(n[2], min = -1, max = 1)
        v[stratumID == 3] <- runif(n[3], min = 1, max = 4)
    } else {
        stop("invalid parameter effect")
    }

    # Generate errors
    if (errors == "normal") {
        epsilon <- rnorm(N)
        delta <- rnorm(N)
    } else if (errors == "t") {
        epsilon <- rt(N, df = 2)
        delta <- rt(N, df = 2)
    } else if (errors == "lognormal") {
        epsilon <- rlnorm(N)
        delta <- rlnorm(N)
    } else if (errors == "exponential") {
        epsilon <- rexp(N) - 1
        delta <- rexp(N) - 1
    } else {
        stop("invalid errors parameter")
    }

    # Generate covariates
    Z <- rep(0:1, length.out = N)
    Y0 <- gen_y0(gamma, v, epsilon)
    Y1 <- gen_y1(gamma, v, delta, Z)
    return(data.frame(Y1, Y0, Z, v, stratumID, epsilon, delta))
}

generate_simulated_pvalues <- function(dataset, reps = 1000) {
    # Inputs: dataset = a dataframe containing columns named Y1
    # (response), Y0 (baseline), Z (treatment), and stratumID
    # (stratum) Returns: a vector of p-values first element is
    # the p-value from the ANCOVA second element is the p-value
    # from the stratified two-sample permutation test third
    # element is the p-value from the linear model test,
    # permuting treatment fourth element is the p-value from the
    # Freedman-Lane linear model test, permuting residuals

    # ANCOVA
    modelfit <- lm(Y1 ~ Y0 + Z + factor(stratumID), data = dataset)
    resanova <- summary(aov(modelfit))
    anova_pvalue <- resanova[[1]]["Z", "Pr(>F)"]

    # Stratified permutation test of Y1
    observed_diff_means <- mean(dataset$Y1[dataset$Z == 1]) -
        mean(dataset$Y1[dataset$Z == 0])
    diff_means_distr <- stratified_two_sample(group = dataset$Z,
        response = dataset$Y1, stratum = dataset$stratumID, reps = reps)
    perm_pvalue <- t2p(observed_diff_means, diff_means_distr,
        alternative = "two-sided")
```

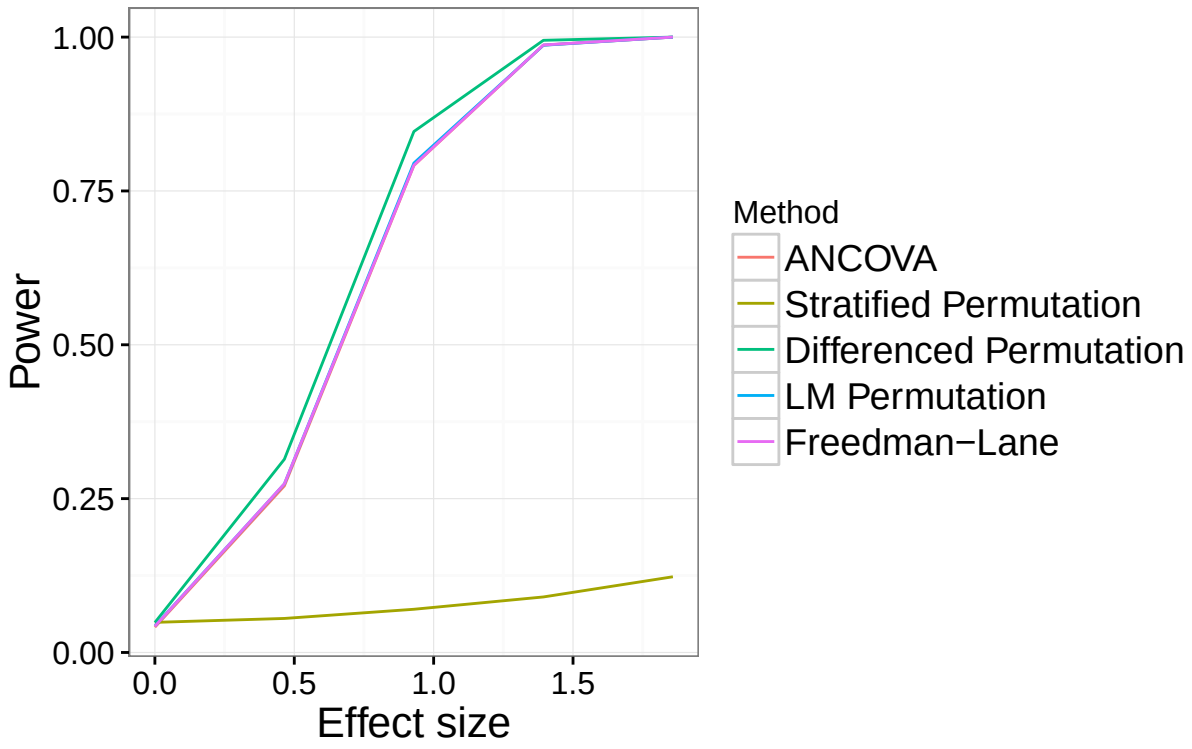
```
    # Diffed permutation test of Y1-Y0
    dataset$diff <- dataset$Y1 - dataset$Y0
    observed_diff_means2 <- mean(dataset$diff[dataset$Z == 1]) -
        mean(dataset$diff[dataset$Z == 0])
    diff_means_distr2 <- stratified_two_sample(group = dataset$Z,
        response = dataset$diff, stratum = dataset$stratumID,
        reps = reps)
    perm_pvalue2 <- t2p(observed_diff_means2, diff_means_distr2,
        alternative = "two-sided")

    # Permutation of treatment in linear model
    observed_t1 <- summary(modelfit)[["coefficients"]]["Z", "t value"]

    # Freedman-Lane linear model residual permutation
    lm2_no_tr <- lm(Y1 ~ Y0 + factor(stratumID), data = dataset)
    dataset$lm2_resid <- residuals(lm2_no_tr)
    lm2_yhat <- fitted(lm2_no_tr)

    lm1and2_t_distr <- replicate(reps, {
        dataset[, c("Z_perm", "lm2_resid_perm")] <- permute_within_groups(dataset[,
            c("Z", "lm2_resid")], dataset$stratumID)
        lm1_perm <- lm(Y1 ~ Y0 + Z_perm + factor(stratumID),
            data = dataset)

        dataset$response_fl <- lm2_yhat + dataset$lm2_resid_perm
        lm2_perm <- lm(response_fl ~ Y0 + Z + factor(stratumID),
            data = dataset)

        c(summary(lm1_perm)[["coefficients"]]["Z_perm", "t value"],
            summary(lm2_perm)[["coefficients"]]["Z", "t value"])
    })
    lm_pvalue <- t2p(observed_t1, lm1and2_t_distr[1, ], alternative = "two-sided")
    fl_pvalue <- t2p(observed_t1, lm1and2_t_distr[2, ], alternative = "two-sided")

    return(c(ANCOVA = anova_pvalue, `Stratified Permutation` = perm_pvalue,
        `Differenced Permutation` = perm_pvalue2, `LM Permutation` = lm_pvalue,
        `Freedman-Lane` = fl_pvalue))
}
```

```
compute_power <- function(pvalues) {
    sapply((0:99)/100, function(p) mean(pvalues <= p, na.rm = TRUE))
}

plot_power_curves <- function(power_mat, title) {
    melt(power_mat) %>% mutate(pvalue = Var1/100) %>% mutate(Method = Var2) %>%
        ggplot(aes_string(x = "pvalue", y = "value", color = "Method")) +
        geom_line() + geom_abline(intercept = 0, slope = 1, linetype = "dashed") +
        xlab("P-value") + ylab("Power") + ggtitle(title) + theme_bw() +
        theme(axis.text.x = element_text(size = 12), axis.text.y = element_text(size = 12),
            axis.title = element_text(size = 16), title = element_text(size = 16),
            legend.title = element_text(size = 12), legend.text = element_text(size = 14),
            strip.text.x = element_text(size = 12))
}
```

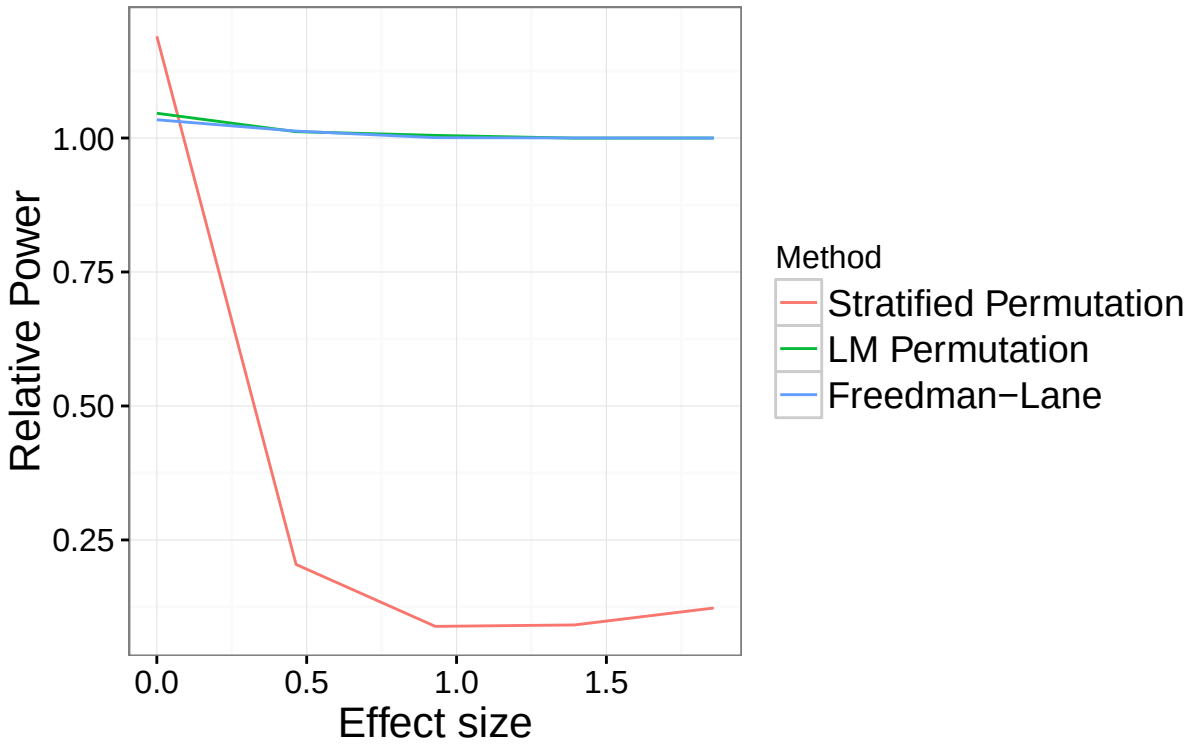
```
plot_power_gamma <- function(powermat_list, gamma_vec, alpha,
    title) {
    gamma_power_alpha <- t(sapply(powermat_list, function(x) x[floor(alpha *
        nrow(x)), ]))
    colnames(gamma_power_alpha) <- c("ANCOVA", "Stratified Permutation",
        "Differenced Permutation", "LM Permutation", "Freedman-Lane")
    melt(gamma_power_alpha, value.name = "Power") %>% mutate(Method = Var2) %>%
        mutate(Effect = rep(gamma_vec, 5)) %>% ggplot(aes(x = Effect,
        y = Power)) + geom_line(aes(color = Method)) + xlab("Effect size") +
        theme_bw() + theme(axis.text.x = element_text(size = 12),
        axis.text.y = element_text(size = 12), axis.title = element_text(size = 16),
        title = element_text(size = 16), legend.title = element_text(size = 12),
        legend.text = element_text(size = 14), strip.text.x = element_text(size = 12))
}

plot_power_ratio_gamma <- function(powermat_list, gamma_vec,
    alpha, title) {
    gamma_power_alpha <- t(sapply(powermat_list, function(x) x[floor(alpha *
        nrow(x)), ]))
    colnames(gamma_power_alpha) <- c("ANCOVA", "Stratified Permutation",
        "Differenced Permutation", "LM Permutation", "Freedman-Lane")
    gamma_power_alpha <- apply(gamma_power_alpha, 2, function(x) x/gamma_power_alpha[,
        "ANCOVA"])
    melt(gamma_power_alpha, value.name = "Power") %>% mutate(Method = Var2) %>%
        mutate(Effect = rep(gamma_vec, 5)) %>% filter(Method !=
        "Differenced Permutation") %>% filter(Method != "ANCOVA") %>%
        ggplot(aes(x = Effect, y = Power)) + geom_line(aes(color = Method)) +
        xlab("Effect size") + ylab("Relative Power") + theme_bw() +
        theme(axis.text.x = element_text(size = 12), axis.text.y = element_text(size = 12),
            axis.title = element_text(size = 16), title = element_text(size = 16),
            legend.title = element_text(size = 12), legend.text = element_text(size = 14),
            strip.text.x = element_text(size = 12))
}
```

Homogeneous treatment effects

```
load("nonlinear_model/vary_gamma_results.Rda")
gamma_vec <- seq(0, 0.2, by = 0.05)
gamma_power <- lapply(gamma_res, function(x) apply(x, 2, compute_power))
plot_power_gamma(gamma_power, tr_effect, 0.05, "Power at level 5%, Constant Treatment Effect")
```




```
plot_power_ratio_gamma(gamma_power, tr_effect, 0.05, "Power at level 5%, Constant Treatment Effect")
```

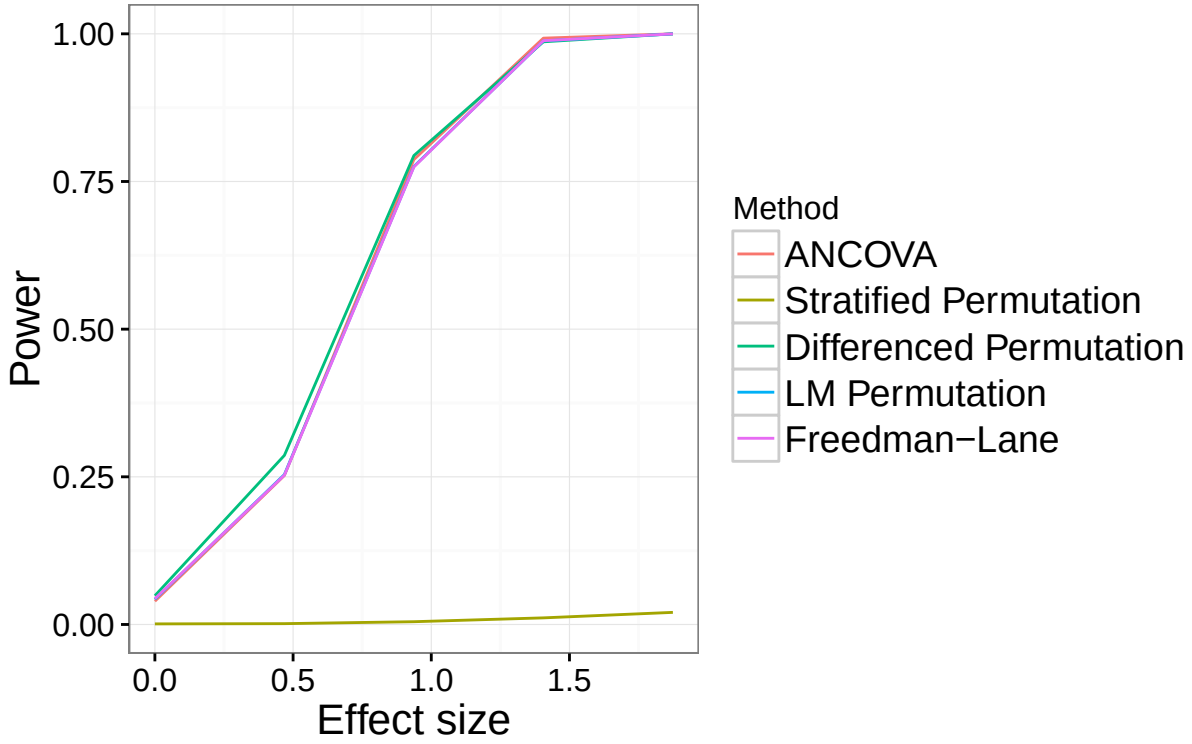


First, we let the v_{ij} have the same distribution across strata and generated normally distributed errors. We varied γ from 0 to 0.2 in steps of 0.05. In the random populations that were generated, this corresponded to population average treatment effects of 0, 0.46, 0.93, 1.39, and 1.86, respectively. Figure 1 shows the empirical power (rate of rejection in the 10,000 simulations) at level 5% for these increasing effect sizes. When the effect size was zero, all five tests had the correct level of 5%. The stratified, unadjusted permutation test

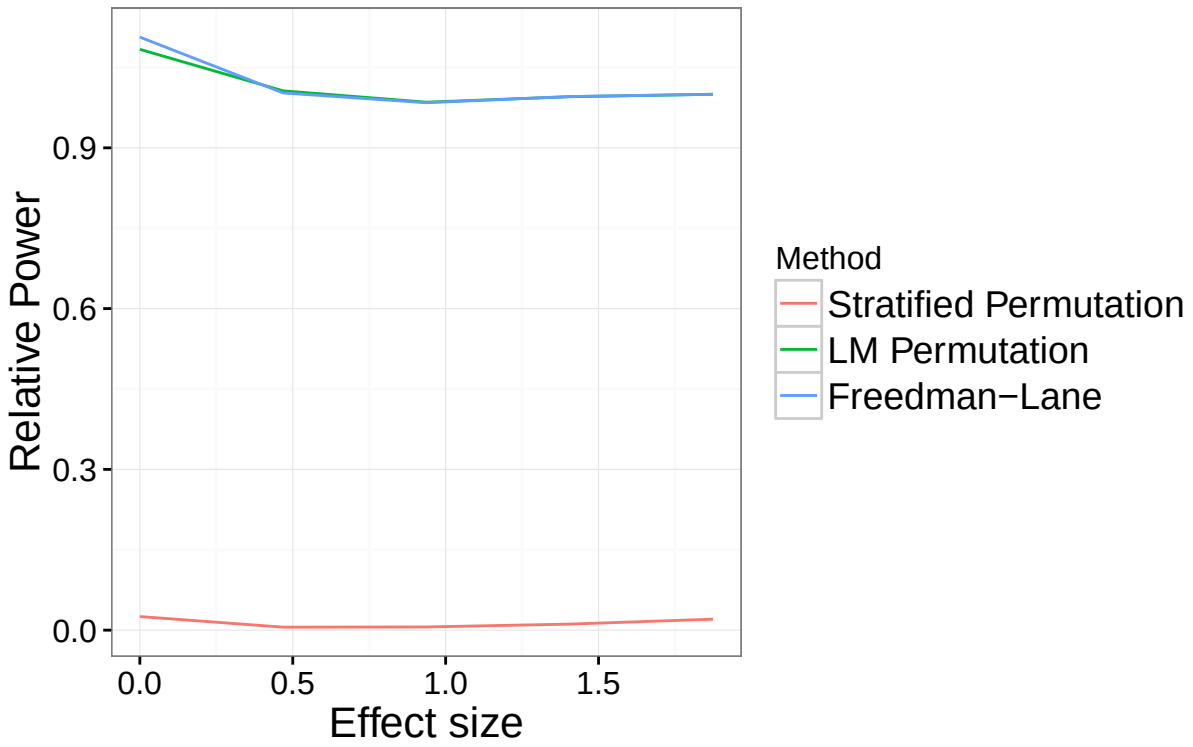
failed here: the treatment effect is entangled with the baseline covariate, so failing to account for the baseline obscures the treatment effect. The parametric and permutation linear model tests perform comparably and are plotted on top of each other, while the differenced permutation test has slightly more power.

Heterogeneous treatment effect

```
load("nonlinear_model/vary_gamma_het_results.Rda")
gamma_vec <- seq(0, 0.2, by = 0.05)
gamma_power_het <- lapply(gamma_res_het, function(x) apply(x,
    2, compute_power))
plot_power_gamma(gamma_power_het, tr_effect_het, 0.05, "Power at level 5%, Heterogeneous Treatment Effe
```




```
plot_power_ratio_gamma(gamma_power_het, tr_effect_het, 0.05,
    "Power at level 5%, Heterogeneous Treatment Effect")
```



Next, we varied the distribution of v_{ij} across strata and generated normally distributed errors. Once again, we varied γ from 0 to 0.2 in steps of 0.05. The corresponding population average treatment effects were of 0, 0.47, 0.94, 1.41, and 1.87, respectively. Figure 2 shows the empirical power at level 5% for these increasing effect sizes. Four of the five tests had the correct level of 5%, while the stratified permutation test rejected only 1% of the time. The same pattern appeared here: the stratified permutation test rarely rejects, while the differenced permutation test dominates all of the linear model based tests.