Harmful Compared to What? The Problem of Gaming and Ambiguous Causal Questions - Online Supplement

Kristoffer Magnusson

2023-11-30

Table of contents

Introduction

This is the online supplement for *Harmful Compared to What? The Problem of Gaming and Ambiguous Causal Questions* (Magnusson, Johansson, & Przybylski, 2023, in preparation), containing the code for all calculations and figures presented in the article.

- Preprint
- GitHub repository
- OSF repository

A HTML version of this document can also be viewed at https://rpsychologist.github.io/gaming-causal-inference-paper

Exchangeability example

This function was created to simulate a binary confounder.

```r
#' Simulate data with a binary exposure & confounder
#'
#' Will generate a dataset with a binary exposure and confounder with an
#' observed difference of a specific magnitude, while the true average
#' causal effect is zero. The outcome is continuous.
#'
#' @param n the number of participants per group
#' @param m_young the mean outcome among the young people
#' @param p_a1_young the proportion of young people among the exposed (A = 1)
#' @param d the (biased) standardized mean difference between the exposed
#' and non-exposed
simulate_confounding <- function(n, m_young, p_a1_young, d) {
    # this is to make sure the overall group diff is equal to d
    m_old <- d / (2 * p_a1_young - 1)
    M <- c(m_young, m_old)
    p <- c(1 - p_a1_young, p_a1_young)
    var_M <- sum((M - sum(M * p))^2 * p)
    SD <- sqrt(1 - var_M)

    params <- data.frame(
        n = c(
            n * (1 - p_a1_young),
            n * p_a1_young,
            n * p_a1_young,
            n * (1 - p_a1_young)
        ),
        A = c(0, 0, 1, 1),
        U = c(2, 1, 2, 1),
        mean = c(
            m_young,
            m_old,
            m_young,
            m_old
        ),
        SD = SD
    )

    pmap(
        params,
        function(n, A, U, mean, SD) {
            data.frame(
```

```
                    A = A,
                    U = U,
                    y = rnorm(n, mean, SD)
                )
            }
        ) %>%
        bind_rows() %>%
        mutate(
            A = factor(A, labels = c("No gaming", "Gaming")),
            U = factor(U, labels = c("Old", "Young"))
        )
    }
```

When then simulate a large data set with a high proportion (85%) young people among the exposed.

```
set.seed(1337)
d <- simulate_confounding(
    n = 1e5,
    m_young = 0,
    p_a1_young = 0.85,
    d = 0.5
)
```

We can see a summary of the simulated data in Table 1.

```
d %>%
    group_by(A) %>%
    summarize(
        "Prop. young" = mean(U == "Young"),
        "Prop. old" = mean(U == "Old"),
        "Mean" = mean(y),
        "SD" = sd(y),
    ) %>%
    kable(digits = 2)
```

The proportion of gamers is,

```
mean(d$A == "Gaming")
```

```
[1] 0.5
```

Table 1: Summary of the simulated data

(a) Young/old is the binary confounder, A is the exposure

A	Prop. young	Prop. old	Mean	SD
No gaming	0.15	0.85	0.61	1
Gaming	0.85	0.15	0.10	1

and the biased difference between the exposed and non-exposed is:

```
mean(d[d$A == "Gaming", "y"]) - mean(d[d$A == "No gaming", "y"])
```

```
[1] -0.5034944
```

Figure 1 shows that the causal effect within each strata is zero, and that the observed difference in outcomes is caused by the confounder.

```
p <- ggplot(
    d,
    aes(
        x = A,
        y = y,
        color = U,
        group = interaction(U, A)
    )
) +
    geom_violinhalf(
        aes(
            fill = U,
            color = U
        ),
        alpha = 0.40,
        position = "identity",
        scale = "count",
        flip = c(1, 2),
    ) +
    stat_summary(
        aes(fill = U),
        color = "white",
        shape = 21,
        geom = "point",
```

```r
      fun = "mean",
      size = 3,
  ) +
  stat_summary(
      aes(fill = U),
      color = "white",
      shape = 21,
      geom = "point",
      fun = "mean",
      size = 3,
  ) +
  stat_summary(
      aes(color = U, group = U),
      geom = "line",
      fun = "mean",
      linewidth = 0.5
  ) +
  stat_summary(
      aes(group = NA),
      geom = "point",
      fun = "mean",
      fill = "black",
      color = "white",
      shape = 21,
      size = 3,
  ) +
  stat_summary(
      aes(group = NA),
      geom = "line",
      fun = "mean",
      col = "black",
      linewidth = 1,
  ) +
  labs(y = "Mental health", x = "Observed outcomes") +
  scale_color_manual(values = c("Young" = col1, "Old" = col2)) +
  scale_fill_manual(values = c("Young" = col1, "Old" = col2)) +
  guides(
      colour = guide_legend("Confounder"),
      fill = guide_legend("Confounder")
  ) +
  lims(y = c(-2.5, 2.5)) +
```

```
        theme_minimal() +
        theme(
            panel.grid.minor.x = element_blank(),
            text = element_text(family = "ArialMT")
        )
p
ggsave(
    "figures/figure-confounding.svg",
    p,
    width = 5,
    height = 4
)
```
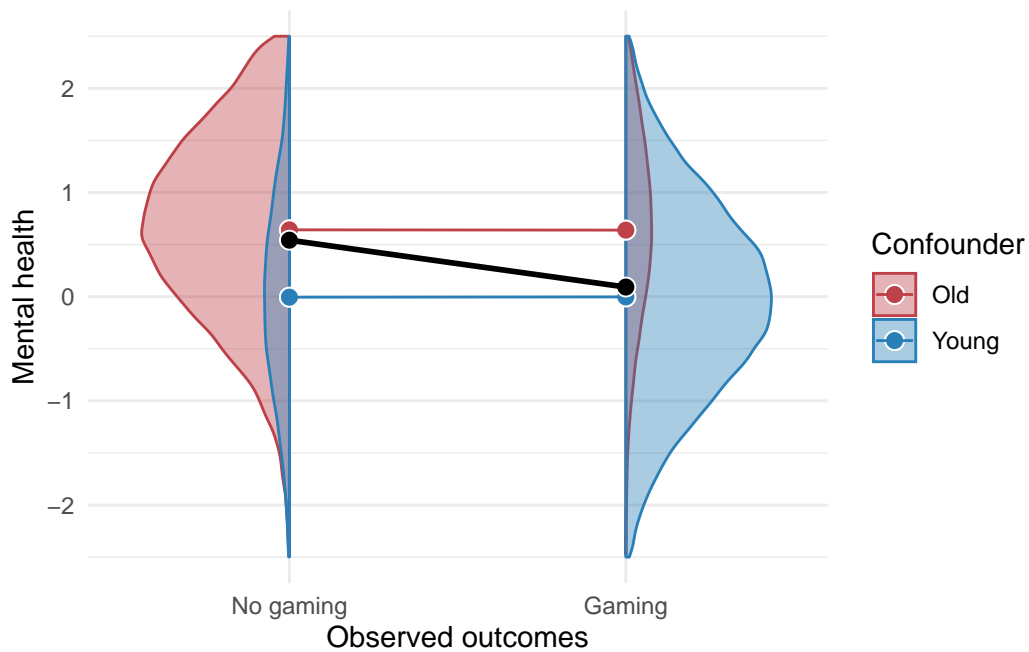

Figure 1: Illustration of a third variable violating the exchangeability assumption

Unsurprisingly, we get the same biased effect if we run a regression without including the confounder U, as shown in Table 2.

```
broom::tidy(lm(y ~ A, data = d)) %>%
    kable(digits = 2)
```

Table 2: Linear regression *without* adjusting for the confounder

term	estimate	std.error	statistic	p.value
(Intercept)	0.61	0	191.73	0
AGaming	-0.50	0	-112.35	0

Adjusting for `U` removes the confounding, as shown in Table 3.

```
broom::tidy(lm(y ~ A + U, data = d)) %>%
    kable(digits = 2)
```

Table 3: Linear regression adjusting for the confounder

term	estimate	std.error	statistic	p.value
(Intercept)	0.71	0.00	223.38	0.00
AGaming	-0.01	0.01	-0.88	0.38
UYoung	-0.71	0.01	-117.24	0.00

Exposure-version confounding example

Now we will focus on exposure-version confounding. First, we write a function to generate simulated data with 4 versions, 2 versions for the non-exposed and 2 versions for the exposed, both exposure and version is influenced by the binary confounder (W).

```
#' Simulate data with exposure-version confounding
#'
#' Will generate a dataset with a binary exposure-version confounder with an
#' observed difference of a specific magnitude, while the true average
#' causal effect is zero. Both the confounder and exposure is binary,
#' whereas the outcome is a continuous variable.
#'
#' @param n the number of participants per group
#' @param pr_W the probability of a participant being young
#' @param pr_A_W the probability of being exposed among the young
#' @param pr_K_W_a0 the probability of K = 2 among the non-exposed young
#' @param pr_K_W_a1 the probability of K = 2 among the exposed young
#' @param sd_a0 the total SD for the non-exposed
#' @param sd_a1 the total SD for the exposed
#' @param d the (biased) standardized mean difference between exposed
```

```r
#' and non-exposed
simulate_data <- function(
    n,
    pr_W,
    pr_A_W,
    pr_K_W_a0,
    pr_K_W_a1,
    sd_a0,
    sd_a1,
    d
) {
    # confounder
    W <- rbinom(n, 1, pr_W)
    # treatment
    A <- rbinom(
        n,
        1,
        ifelse(W == 1, pr_A_W, 1 - pr_A_W)
    )
    # versions when A = 0
    K_a0 <- rbinom(
        n,
        1,
        ifelse(W == 1, pr_K_W_a0, 1 - pr_K_W_a0)
    ) + 1
    # versions when A = 1
    K_a1 <- rbinom(
        n,
        1,
        ifelse(W == 1, pr_K_W_a1, 1 - pr_K_W_a1)
    ) + 1

    # Code below appears more complicated than it is,
    # it's just a bunch of conditional probabilities to get the correct
    # proportions and variances. We do this so the total variance
    # will be sd_a0² and sd_a1², and the observed standardized
    # difference will be equal to `d`.

    # Pr(A)
    pr_A <- pr_A_W * pr_W + (1 - pr_A_W) * (1 - pr_W)
    # Solve for params in non-gamers group
```

```r
# Pr(W = 1 | A = 0)
pr_w_a0 <- pr_W * (1 - pr_A_W) / (1 - pr_A)
# Pr(K == 2 | A = 0)
p_k2_a0 <- pr_w_a0 * pr_K_W_a0 + (1 - pr_w_a0) * (1 - pr_K_W_a0)
# Pr(K == 1 | A = 0)
p_k1_a0 <- 1 - p_k2_a0
# E(Y0 | K = 1)
M_Y0_k1 <- -0.5
# E(Y0 | K = 2)
M_Y0_k2 <- 0.5
pr <- c(p_k1_a0, p_k2_a0)
# solve for variance
M <- c(M_Y0_k1, M_Y0_k2)
y_a0 <- sum(M * pr)
var_M <- sum((M - y_a0)^2 * pr)
sd_error_A0 <- sqrt(sd_a0 - var_M)

# Solve for params in treatment group
# Pr(W = 1 | A = 1)
pr_w_a1 <- pr_W * pr_A_W / (pr_A)
# Pr(K == 2 | A = 1)
p_k2_a1 <- pr_w_a1 * pr_K_W_a1 + (1 - pr_w_a1) * (1 - pr_K_W_a1)
p_k1_a1 <- 1 - p_k2_a1
pr <- c(p_k1_a1, p_k2_a1)
SD_pooled <- sqrt((sd_a0^2 * (1 - pr_A) + sd_a1^2 * pr_A))
es <- d * SD_pooled
delta <- (-2 * p_k1_a1 * M_Y0_k2 + M_Y0_k2 - (y_a0 - es)) /
    (2 * p_k1_a1 - 1)
M <- c(M_Y0_k1 - delta, M_Y0_k2 + delta)
y_a1 <- sum(M * pr)
var_M <- sum((M - y_a1)^2 * pr)
sd_error_A1 <- sqrt(sd_a1^2 - var_M)

# generate outcomes
Y1 <- model.matrix(~ 0 + as.factor(K_a1)) %*%
    c(M_Y0_k1 - delta, M_Y0_k2 + delta) +
    rnorm(n, 0, sd_error_A1)
Y0 <- model.matrix(~ 0 + as.factor(K_a0)) %*%
    c(M_Y0_k1, M_Y0_k2) +
    rnorm(n, 0, sd_error_A0)
# combine data
```

```r
    list(
        data = data.frame(
            y = ifelse(A == 0, Y0, Y1),
            y1 = Y1,
            y0 = Y0,
            K_a0,
            K_a1,
            K = ifelse(A == 0, K_a0, K_a1),
            A = A,
            W = W
        ),
        params = list(
            "pr_A" = pr_A,
            "sd_error_A0" = sd_error_A0,
            "sd_error_A1" = sd_error_A1,
            "y_a0" = y_a0,
            "y_a1" = y_a1,
            "p_k1_a0" = p_k1_a0,
            "p_k2_a0" = p_k2_a0,
            "pr_w_a0" = pr_w_a0,
            "pr_w_a1" = pr_w_a1,
            "p_k1_a1" = p_k1_a1,
            "p_k2_a1" = p_k2_a1,
            "delta" = delta,
            "SD_pooled" = SD_pooled,
            "es" = es
        )
    )
}
```

Let's simulate a large data set, where 50% are exposed, among the exposed 80% are young, and among the exposed young 90% get K = 2, and among the non-exposed young 90% also get K = 2.

```r
pr_W <- 0.5 # Pr(W)
pr_A_W <- 0.8 # Pr(A | W)
pr_K_W_a0 <- 0.1 # Pr(K | W, A = 0)
pr_K_W_a1 <- 0.1 # Pr(K | W, A = 1)
sd_a0 <- 1
sd_a1 <- 1.2
res <- simulate_data(
    n = 1e5,
```

```
    pr_W = pr_W,
    pr_A_W = pr_A_W,
    pr_K_W_a0 = pr_K_W_a0,
    pr_K_W_a1 = pr_K_W_a1,
    d = 0.5,
    sd_a0 = 1,
    sd_a1 = 1.2
)
d <- res$data
```

Check if the simulated data match our inputs.

```
# non-gamers
d %>%
    filter(A == 0) %>%
    summarize(
        mean(K == 1),
        mean(K == 2),
        mean(y),
        sd(y),
        mean(W == 0)
    ) %>%
    pivot_longer(
        everything(),
        names_to = "parameter",
        values_to = "value"
    ) %>%
    mutate(
        theta = with(
            res$params,
            c(
                p_k1_a0,
                p_k2_a0,
                y_a0,
                sd_a0,
                1 - pr_w_a0
            )
        )
    ) %>%
    kable(digits = 2)
```

Table 4: Summary of the non-exposed group

parameter	value	theta
mean(K == 1)	0.26	0.26
mean(K == 2)	0.74	0.74
mean(y)	0.24	0.24
sd(y)	1.00	1.00
mean(W == 0)	0.80	0.80

```r
d %>%
    filter(A == 1) %>%
    summarize(
        mean(K == 1),
        mean(K == 2),
        mean(y),
        sd(y),
        mean(W == 0)
    ) %>%
    pivot_longer(
        everything(),
        names_to = "parameter",
        values_to = "value"
    ) %>%
    mutate(
        theta = with(
            res$params,
            c(
                p_k1_a1,
                p_k2_a1,
                y_a1,
                sd_a1,
                1 - pr_w_a1
            )
        )
    ) %>%
    kable(digits = 2)
```

Table 5: Summary of the exposed group

parameter	value	theta
mean(K == 1)	0.74	0.74
mean(K == 2)	0.26	0.26
mean(y)	-0.32	-0.31
sd(y)	1.20	1.20
mean(W == 0)	0.20	0.20

We also check so that the proportions match, stratified by age and version.

```
d %>%
    group_by(A) %>%
    summarize(
        "Young K1" = sum(K == 1 & W == 1) / sum(W == 1),
        "Young K2" = sum(K == 2 & W == 1) / sum(W == 1),
        "Old K1" = sum(K == 1 & W == 0) / sum(W == 0),
        "Old K2" = sum(K == 2 & W == 0) / sum(W == 0),
        n()
    ) %>%
    kable(digits = 2)
```

Table 6: Summary stratified by age and version

A	Young K1	Young K2	Old K1	Old K2	n()
0	0.9	0.1	0.1	0.9	50044
1	0.9	0.1	0.1	0.9	49956

We also see that the overall values are correct, including the biased effect.

```
pr_A <- mean(d$A)
pooled_SD <- sqrt(
    sd(d[d$A == 0, "y"])^2 * (1 - pr_A) +
    sd(d[d$A == 1, "y"])^2 * pr_A
)
m_diff <- mean(d[d$A == 1, "y"]) - mean(d[d$A == 0, "y"])

data.frame(
    parameter = c(
        "Pr(A)",
```

```
        "mean diff",
        "pooled SD",
        "Cohen's d"
    ),
    value = c(
        pr_A,
        m_diff,
        pooled_SD,
        m_diff / pooled_SD
    ),
    theta = with(
        res$params,
        c(
            pr_A,
            y_a1 - y_a0,
            SD_pooled,
            (y_a1 - y_a0) / SD_pooled
        )
    )
) %>%
    kable(digits = 2)
```

Table 7: Summary overall

parameter	value	theta
Pr(A)	0.50	0.50
mean diff	-0.55	-0.55
pooled SD	1.11	1.10
Cohen's d	-0.50	-0.50

Naive analysis

We get a biased estimate if we naively look at the group differences

```
broom::tidy(lm(y ~ A, data = d)) %>%
    kable(digits = 2)
```
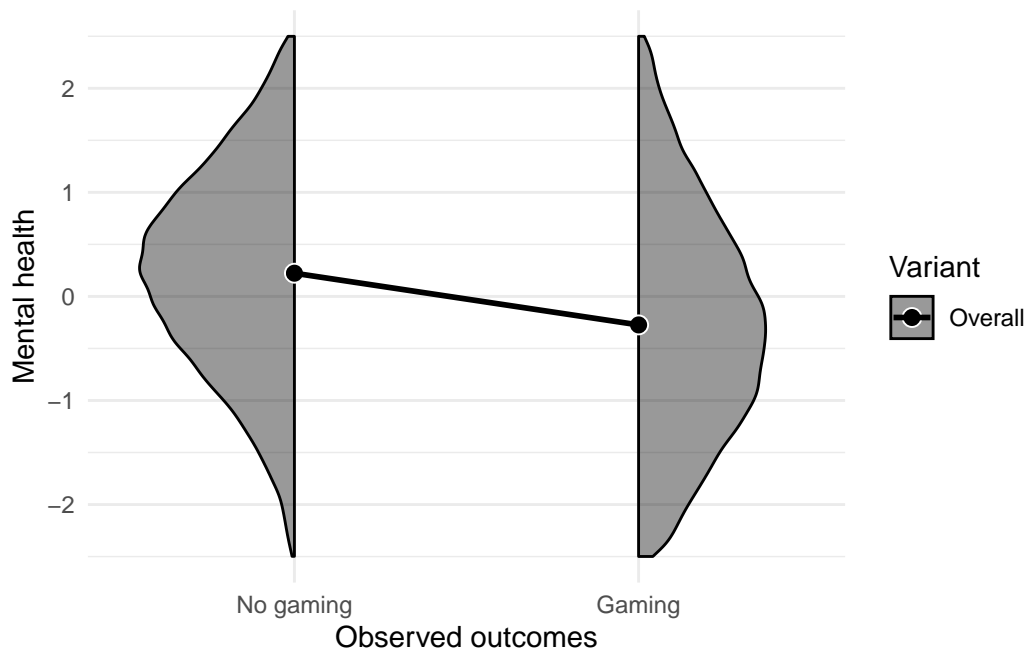
Figure 2: Overall distribution of outcomes ignoring exposure-versions

Table 8: Linear regression ignoring the exposure-version confounder

| term	estimate	std.error	statistic	p.value
(Intercept)	0.24	0.00	47.66	0
A	-0.55	0.01	-79.22	0

If we look at some diagnostics plots things look pretty normal.

```
d_samp <- d %>%
    group_by(A) %>%
    sample_n(1000) %>%
    ungroup()
fit <- lm(y ~ factor(A), data = d_samp)
performance::check_model(fit, check = c("qq", "pp_check"))
d_samp <- d_samp %>%
    mutate(
        yhat = fitted(fit),
        res_sqrt = sqrt(abs(rstandard(fit)))
    )
```

```
ggplot(
    d_samp,
    aes(x = factor(A), res_sqrt)
    ) +
    geom_boxplot() +
    theme_minimal()
```
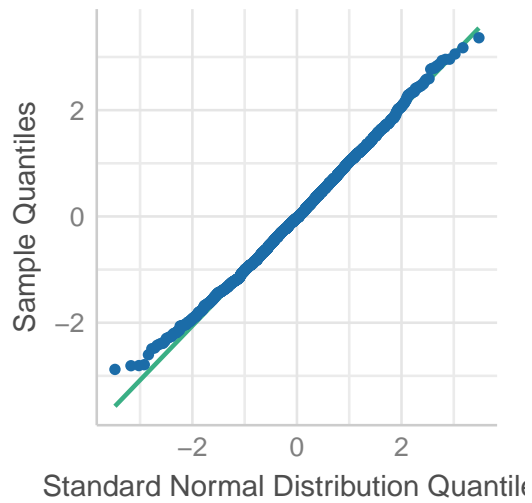
Posterior Predictive Check
Model−predicted lines should resemble obs
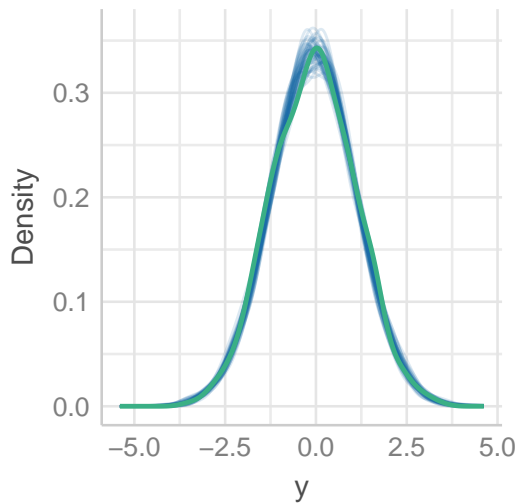
Normality of Residuals
Dots should fall along the line
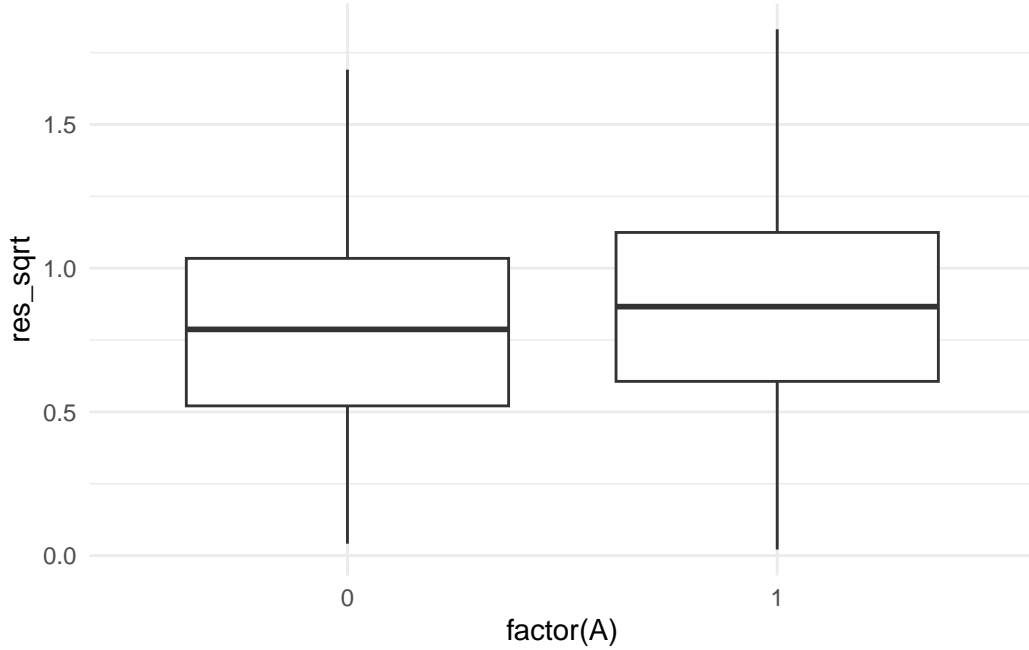
Figure 3: Diagnostic plots

Figure 4: Diagnostic plots

In Figure 5 we plot the exposure-versions, and we and see that the overall comparison is a weighted average of the different exposure versions in each population. The overall comparison is biased when there is exposure-version confounding, however, we could compare specific versions, e.g.,

$$\mathbb{E}(Y \mid \text{Non-gamers}, K = 2) - \mathbb{E}(Y \mid \text{Gamers}, K = 1).$$

In Figure 6 we look at the data within each strata of the confounder, and see that the overall impact among the older group is positive, whereas it is negative among the younger group. This is entirely driven by young and old people being exposed to different versions of the exposure.

Average treatment effect (ATE)

If we ignore versions, the average causal effect can be written as,
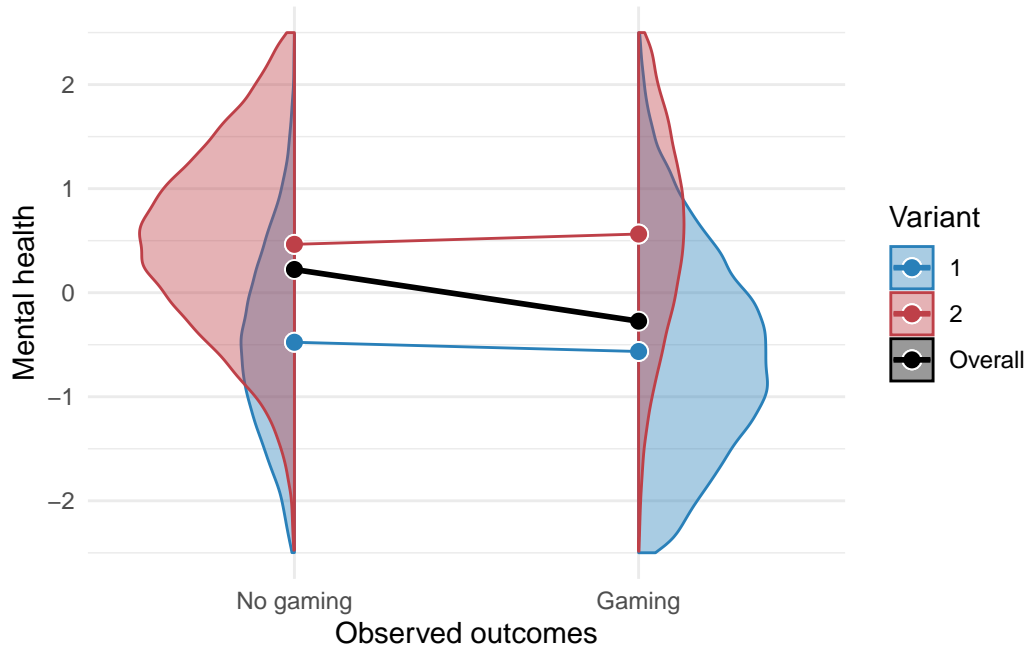
Figure 5: The distribution of the different exposure-versions. Circles represent averages, black circles are averaged over both versions
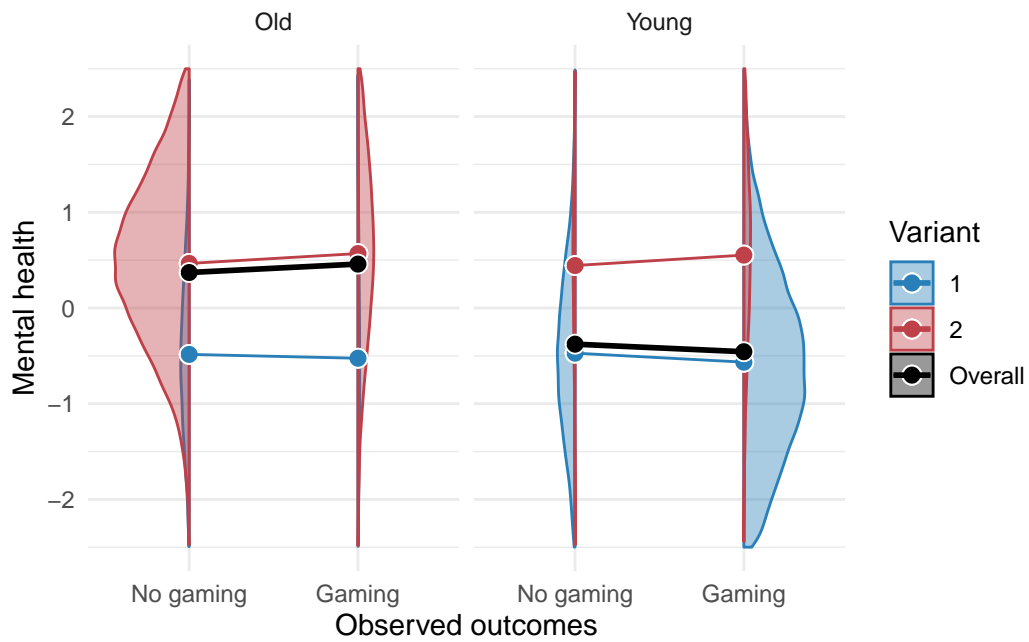

Figure 6: The distribution of exposure-versions within each level of the confounder. Circles represent averages, black circles are averaged over both versions.

$$\text{ATE} = \mathbb{E}(Y^{\text{Gaming}} - Y^{\text{No gaming}}) \tag{1}$$

$$= \mathbb{E}(Y^{\text{Gaming}}) - \mathbb{E}(Y^{\text{No gaming}}) \tag{2}$$

$$= \mathbb{E}(Y^{\text{Gaming}} \mid A = \text{Gaming}) - \mathbb{E}(Y^{\text{No gaming}} \mid A = \text{No gaming}) \tag{3}$$

$$= \mathbb{E}(Y^{\text{Obs}} \mid A = \text{Gaming}) - \mathbb{E}(Y^{\text{Obs}} \mid A = \text{No gaming}) \tag{4}$$

Going from step 2 to step 3 requires exchangeability, whereas going from step 3 to 4 assumes consistency.

If we add versions to the definition, then the average causal effect of gaming can be written (VanderWeele and Hernán 2013),

$$
\begin{aligned}
\text{ATE} &= \mathbb{E}(Y^{\text{Gaming}}) - \mathbb{E}(Y^{\text{No gaming}}) \\
&= \sum_k \mathbb{E}(Y \mid \text{Gaming}, K = k)\Pr(K = k \mid \text{Gaming}) \\
&\quad - \sum_k \mathbb{E}(Y \mid \text{No gaming}, K = k)\Pr(K = k \mid \text{No gaming})
\end{aligned}
$$

that is, we are comparing the outcomes if everyone in the population were gamers versus if no one were playing games, weighted by the distribution of the exposure versions.

We can get the correct estimate by adjusting for age, for instance, by including age as a covariate or using inverse probability weighting (IPW), as shown in Table 9 and Table 10.

```
broom::tidy(
        lm(y ~ A + W, data = d)
    ) %>%
    kable(digits = 2)
```

Table 9: Outcome regression adjusting for age.

term	estimate	std.error	statistic	p.value
(Intercept)	0.42	0.00	84.91	0.00
A	0.01	0.01	0.73	0.46
W	-0.93	0.01	-112.52	0.00

```
# IPW
mod1 <- glm(
    A ~ W,
```

```
    data = d,
    family = binomial(link = "logit")
)
d2 <- d %>%
    mutate(
        p = predict(mod1, type = "response")
    ) %>%
    mutate(ipw = (A / p) + ((1 - A) / (1 - p)))

broom::tidy(
    lm(y ~ A, weights = ipw, data = d2)
    ) %>%
    kable(digits = 2)
```

Table 10: IPW regression adjusting for age.

term	estimate	std.error	statistic	p.value
(Intercept)	0.00	0.01	-0.85	0.40
A	0.01	0.01	0.80	0.42

In Figure 7 we plot the counterfactuals, and we see that the ATE is 0. However, we'd get a much more nuanced answer if we had data on the exposure versions.

Average treatment effect on the treated (ATT).

It's possible that most people do not have the ATE in mind when thinking about the causal effect of video games. It might be more intuitive to imagine what would happened to the population of gamers if they did not play games. This estimand is called *the average treatment effect on the treated* (ATT). If the "no gaming" group was well-defined (i.e. lacked multiple versions), then we could get the ATT simply by comparing the average mental health the among gamers with the average mental health among the non-gamers.

$$ATT = \mathbb{E}(Y^{\text{Gaming}} - Y^{\text{No gaming}} \mid A = \text{Gaming})$$
$$= \mathbb{E}(Y^{\text{Gaming}} \mid A = \text{Gaming}) - \mathbb{E}(Y^{\text{No gaming}} \mid A = \text{Gaming}),$$

$\mathbb{E}(Y^{\text{No gaming}} \mid A = \text{Gaming})$ is not something we could observe.

With our data we can calculate the ATT using g-computation, the estimated ATT is shown in Table 11.
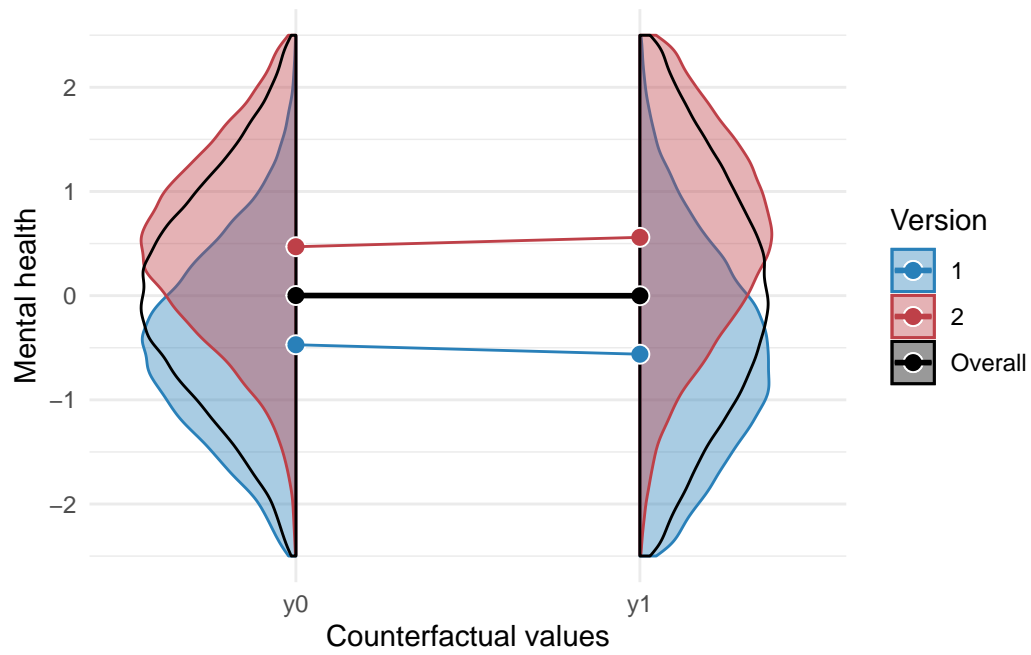
Figure 7: The counterfactual distributions of the different exposure-versions for the whole population under both levels of the exposure, the overall difference represents the average treatment effect.

```r
lm_fit <- lm(y ~ A * W, data = d)
d_tx <- d %>% filter(A == 1)
d_pred <- tibble(
    d_tx,
    y_a0 = predict(
        lm_fit,
        newdata = mutate(d_tx, A = 0)
    ),
    y_a1 = d_tx$y,
    att = y_a1 - y_a0
)
```

```r
d_pred %>%
    summarize(
        ATT = mean(y_a1 - y_a0)
    ) %>%
    mutate(
        ATT_true = d %>%
            filter(A == 1) %>%
            summarize(ATT = mean(y1 - y0)) %>%
            pull(ATT)
    ) %>%
    kable(digits = 3)
```

Table 11: ATT estimate calculated using g-computation

ATT	ATT_true
-0.074	-0.085

We can plot what's going on here. In Figure 8, `y1` is the observed data among the exposed. `y0` would not be observed, but since this is simulated data, we can plot it anyway. The dashed `y0` curves show the observed data. We see that if we compare the gamers with the non-gamers we get a counterfactual comparison that is wrong, the versions of `y0` is weighted incorrectly. This is caused by the exposure-version confounding.

```r
tmp <- d %>%
    filter(A == 1) %>%
    pivot_longer(
        cols = c(y1, y0)
    ) %>%
```

```r
    mutate(
        K = case_when(
            name == "y1" ~ K_a1,
            name == "y0" ~ K_a0
        ),
        y = ifelse(name == "y0", y, NA),
        group = paste0(name, "k", K)
    )
p <- ggplot(
    data = tmp,
    aes(
        x = factor(name),
        value, color = interaction(K, A),
        group = interaction(K, A)
    )
) +
    geom_violinhalf(
        data = filter(tmp, name == "y0"),
        aes(
            fill = factor(K_a0),
            color = factor(K_a0),
            group = interaction("y0", K_a0)
        ),
        alpha = 0.4,
        position = "identity",
        scale = "count",
        flip = TRUE,
    ) +
    geom_violinhalf(
        data = filter(d, A == 0),
        aes(
            x = "y0",
            y = y, color = factor(K),
            group = interaction("y0", K)
        ),
        fill = NA,
        linetype = "dashed",
        alpha = 0.50,
        position = "identity",
        scale = "count",
        flip = TRUE,
```

```r
) +
geom_violinhalf(
    data = filter(tmp, name == "y1"),
    aes(
        fill = factor(K_a1),
        color = factor(K_a1),
        group = interaction("y1", K_a1)
    ),
    alpha = 0.4,
    position = "identity",
    scale = "count",
) +
stat_summary(
    aes(fill = factor(K)),
    color = "white",
    shape = 21,
    geom = "point",
    fun = "mean",
    size = 3,
) +
stat_summary(
    data = filter(d, A == 0),
    aes(
        x = "y0",
        y = y,
        color = factor(K)
    ),
    geom = "point",
    shape = 1,
    fun.y = "mean",
    size = 5,
) +
stat_summary(
    aes(
        color = factor(K),
        group = factor(K)
    ),
    geom = "line",
    fun.y = "mean",
    linewidth = 0.5
) +
```

```r
stat_summary(
    aes(
        group = NA,
        fill = "Overall"
    ),
    geom = "point",
    fun = "mean",
    color = "white",
    shape = 21,
    size = 3,
) +
stat_summary(
    data = filter(d, A == 0),
    aes(
        x = "y0", y = y,
        group = NA,
        color = "Overall",
        fill = "Overall"
    ),
    geom = "point",
    shape = 1,
    fun.y = "mean",
    size = 5,
) +
stat_summary(
    aes(group = NA, color = "Overall"),
    geom = "line",
    fun = "mean",
    linewidth = 1,
) +
lims(y = c(-2.5, 2.5)) +
labs(
    y = "Mental health",
    x = "Counterfactual"
) +
scale_color_manual(
    values = c(
        "1" = col1,
        "2" = col2,
        "Overall" = "black"
    )
```

```r
    ) +
    scale_fill_manual(
        values = c(
            "1" = col1,
            "2" = col2,
            "Overall" = "black"
        )
    ) +
    guides(
        colour = guide_legend("Version"),
        fill = guide_legend("Version")
    ) +
    theme_minimal() +
    theme(
        panel.grid.minor.x = element_blank(),
        text = element_text(family = "ArialMT")
    )
ggsave(
    "figures/figure-version-confounding-4.svg",
    p,
    width = 5,
    height = 4
)
p
```
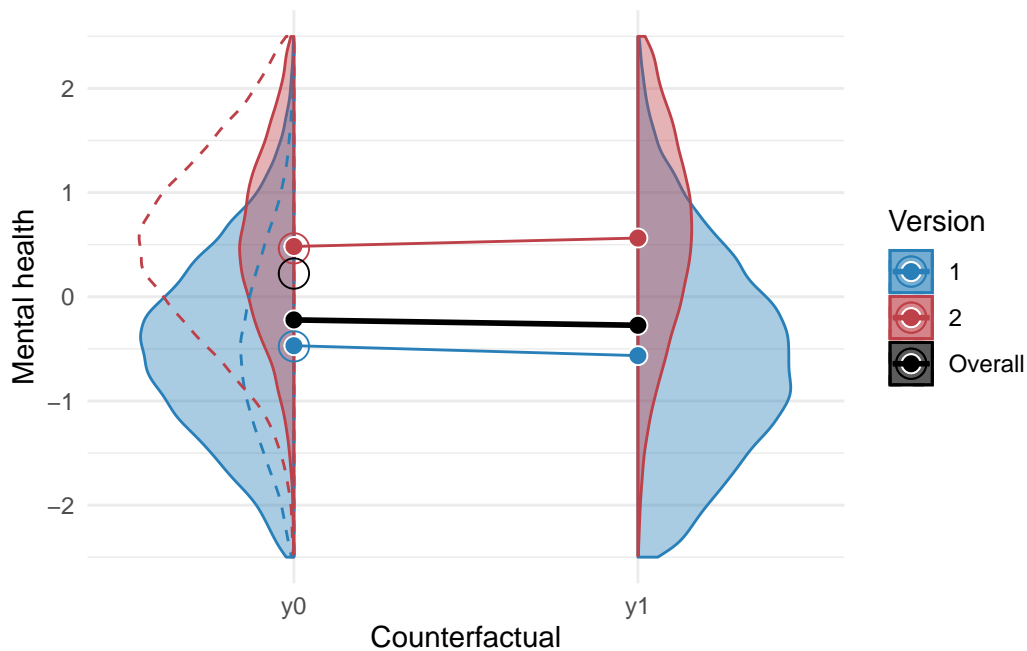
Figure 8: The counterfactual distributions of the exposure-versions among only the gamers. The overall difference represents the average treatment effect on the treated. The dotted lines represent the distributions among non-gamers, and are added as a reference only. Circles represent averages, black circles are averaged over both versions

References

VanderWeele, Tyler J., and Miguel A. Hernán. 2013. "Causal Inference Under Multiple Versions of Treatment." *Journal of Causal Inference* 1 (1): 1–20. https://doi.org/10.1515/jci-2012-0002.