Power Analysis for Trust Experiment

```
suppressMessages({
  library(tidyverse)
  library(fixest)
  library(pwr)
})

set.seed(42)

trounds <- read_csv("../data/generated/trust_rounds.csv", show_col_types = FALSE) %>%
  mutate(
    experiment = factor(ifelse(
      experiment == "ftrust",
      "Business Framing", "Neutral Framing"
    ), c("Neutral Framing", "Business Framing")),
    pct_returned = sent_back_amount/(3*sent_amount)
  )
tparticipants <- read_csv("../data/generated/trust_participants.csv", show_col_types = FAL
  mutate(
    experiment = factor(ifelse(
      experiment == "ftrust",
      "Business Framing", "Neutral Framing"
    ), c("Neutral Framing", "Business Framing"))
  )
```

Descriptive statistics of pretest data to standardize the power tests

```
mn_sent_start <- mean(trounds$sent_amount[trounds$round == 1])
sd_sent <- sd(trounds$sent_amount)
mn_pct_returned_start <- mean(trounds$pct_returned[trounds$round == 1])
sd_pct_returned <- sd(trounds$pct_returned)

message(sprintf(
  "Mean start (SD all) sent: %.2f (%.2f)", mn_sent_start, sd_sent
))
```

```
Mean start (SD all) sent: 50.80 (14.70)
```

```
message(sprintf(
    "Mean start (SD all) pct_returned: %.2f (%.2f)",
    mn_pct_returned_start, sd_pct_returned
))
```

```
Mean start (SD all) pct_returned: 0.45 (0.09)
```

```
# Some pretest regressions to see how rounds affect our DVs:

ols_sent_amount <- lm(sent_amount ~ round, data = trounds)
summary(ols_sent_amount)
```

```

Call:
lm(formula = sent_amount ~ round, data = trounds)

Residuals:
    Min      1Q  Median      3Q     Max
-10.328  -7.865  -5.402  -2.939  47.061

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  52.1175     0.9926  52.507  < 2e-16 ***
round         0.8211     0.1602   5.124 3.59e-07 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 14.52 on 995 degrees of freedom
Multiple R-squared:  0.02571,   Adjusted R-squared:  0.02473
F-statistic: 26.26 on 1 and 995 DF,  p-value: 3.59e-07
```

```r
summary(lm(sent_amount ~ round*experiment, data = trounds))
```

```
Call:
lm(formula = sent_amount ~ round * experiment, data = trounds)

Residuals:
    Min      1Q  Median      3Q     Max
-16.146  -7.409  -3.382  -1.342  47.026

Coefficients:
                                 Estimate Std. Error t value Pr(>|t|)
(Intercept)                       50.5267     1.3481  37.479  < 2e-16 ***
round                              0.4079     0.2173   1.877  0.06077 .
experimentBusiness Framing         3.1375     1.9078   1.645  0.10038
round:experimentBusiness Framing   0.8403     0.3080   2.728  0.00648 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 13.95 on 993 degrees of freedom
Multiple R-squared:  0.102, Adjusted R-squared:  0.09926
F-statistic: 37.59 on 3 and 993 DF,  p-value: < 2.2e-16
```

```r
summary(lm(log(sent_amount) ~ round, data = trounds))
```

```
Call:
lm(formula = log(sent_amount) ~ round, data = trounds)

Residuals:
     Min       1Q   Median       3Q      Max
-0.14855 -0.11526 -0.08196 -0.04867  0.64448

Coefficients:
```

```
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 3.949593   0.014419 273.911  < 2e-16 ***
round       0.011098   0.002328   4.768 2.14e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2109 on 995 degrees of freedom
Multiple R-squared:  0.02234,   Adjusted R-squared:  0.02135
F-statistic: 22.73 on 1 and 995 DF,  p-value: 2.141e-06
```

```r
summary(lm(log(sent_amount) ~ round*experiment, data = trounds))
```

```
Call:
lm(formula = log(sent_amount) ~ round * experiment, data = trounds)

Residuals:
     Min       1Q   Median       3Q      Max
-0.23224 -0.11412 -0.04967 -0.02206  0.64900

Coefficients:
                                 Estimate Std. Error t value Pr(>|t|)
(Intercept)                      3.923034   0.019564 200.524   <2e-16 ***
round                            0.005523   0.003153   1.752   0.0802 .
experimentBusiness Framing       0.052481   0.027686   1.896   0.0583 .
round:experimentBusiness Framing 0.011352   0.004470   2.540   0.0112 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2025 on 993 degrees of freedom
Multiple R-squared:  0.1007,    Adjusted R-squared:  0.098
F-statistic: 37.07 on 3 and 993 DF,  p-value: < 2.2e-16
```

```r
fe_sent_amount <- feols(
  sent_amount ~ experiment | round, cluster = c("round", "group_id"),
  data = trounds
)
summary(fe_sent_amount)
```

```
OLS estimation, Dep. Var.: sent_amount
```

```
Observations: 997
Fixed-effects: round: 10
Standard-errors: Clustered (round & group_id)
                           Estimate Std. Error t value Pr(>|t|)
experimentBusiness Framing  7.74488    2.66715 2.90381  0.04395 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
RMSE: 13.9     Adj. R2: 0.094154
             Within R2: 0.071864
```

```
fixef(fe_sent_amount)$round
```

```
       1        2        3        4        5        6        7        8
46.92756 49.12756 50.92756 52.62756 54.02756 55.12756 54.72756 54.70203
       9       10
54.75253 54.75253
```

```
# Clearly not exponential but decreasing positive trend for rounds.
# Sticking to a linear trend for the estimation.

summary(lm(pct_returned ~ round, data = trounds))
```

```
Call:
lm(formula = pct_returned ~ round, data = trounds)

Residuals:
     Min       1Q   Median       3Q      Max
-0.11316 -0.11304  0.05355  0.05363  0.22035

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 4.463e-01  6.252e-03  71.385   <2e-16 ***
round       1.951e-05  1.009e-03   0.019    0.985
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.09146 on 995 degrees of freedom
Multiple R-squared:  3.756e-07, Adjusted R-squared:  -0.001005
F-statistic: 0.0003737 on 1 and 995 DF,  p-value: 0.9846
```

```r
summary(lm(pct_returned ~ round*experiment, data = trounds))
```

```
Call:
lm(formula = pct_returned ~ round * experiment, data = trounds)

Residuals:
     Min       1Q   Median       3Q      Max
-0.13101 -0.09782  0.03774  0.06534  0.23831

Coefficients:
                                   Estimate Std. Error t value Pr(>|t|)
(Intercept)                       0.4650347  0.0087245  53.302  < 2e-16 ***
round                            -0.0006938  0.0014061  -0.493  0.62181
experimentBusiness Framing       -0.0373835  0.0123467  -3.028  0.00253 **
round:experimentBusiness Framing  0.0013951  0.0019932   0.700  0.48413
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.09031 on 993 degrees of freedom
Multiple R-squared:  0.02694,   Adjusted R-squared:  0.024
F-statistic: 9.165 on 3 and 993 DF,  p-value: 5.519e-06
```

```r
fe_pct_returned <- feols(
  pct_returned ~ experiment | round, cluster = c("round", "group_id"),
  data = trounds
)
summary(fe_pct_returned)
```

```
OLS estimation, Dep. Var.: pct_returned
Observations: 997
Fixed-effects: round: 10
Standard-errors: Clustered (round & group_id)
                           Estimate Std. Error  t value Pr(>|t|)
experimentBusiness Framing -0.029721   0.010969 -2.70962 0.053558 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
RMSE: 0.090137     Adj. R2: 0.016834
                 Within R2: 0.02646
```

```r
fixef(fe_pct_returned)$round
```

```
        1         2         3         4         5         6         7         8
0.4631936 0.4631936 0.4595162 0.4592082 0.4593048 0.4609390 0.4610133 0.4616848
        9        10
0.4618798 0.4622539
```

```r
# No round trend in pct_returned based on pretest data
```

Equation-based Power Analysis based on Pretest Data

Round Based Analysis

```r
pwr.t.test(500, 5/sd_sent)
```

```

     Two-sample t test power calculation

              n = 500
              d = 0.3400633
      sig.level = 0.05
          power = 0.9996772
    alternative = two.sided

NOTE: n is number in *each* group
```

```r
pwr.t.test(d = 5/sd_sent, power = 0.8)
```

```

     Two-sample t test power calculation

              n = 136.7099
              d = 0.3400633
      sig.level = 0.05
          power = 0.8
    alternative = two.sided

NOTE: n is number in *each* group
```

```r
pr <- pwr.t.test(n = 500, power = 0.8)
sprintf("MDE Trust sent: %.2f", pr$d * sd_sent)
```

```
[1] "MDE Trust sent: 2.61"
```

```r
sprintf("MDE Trust pct_returned: %.4f", pr$d * sd_pct_returned)
```

```
[1] "MDE Trust pct_returned: 0.0162"
```

Participant and Dyad Based Analysis

```r
mn_payoff_part <- mean(tparticipants$payoff)
sd_payoff_part <- sd(tparticipants$payoff)
sprintf("Mean (SD) of part payoff: %.2f (%.2f)", mn_payoff_part, sd_payoff_part)
```

```
[1] "Mean (SD) of part payoff: 1063.05 (250.36)"
```

```r
pwr.t.test(100, (0.1*mn_payoff_part)/sd_payoff_part)
```

```

     Two-sample t test power calculation

              n = 100
              d = 0.4246088
      sig.level = 0.05
          power = 0.8479971
    alternative = two.sided

NOTE: n is number in *each* group
```

```r
pr <- pwr.t.test(n = 100, power = 0.8)
sprintf(
  "MDE Payoff part: %.2f (%.1f %% of mean)", pr$d * sd_payoff_part,
  100*(pr$d * sd_payoff_part)/mn_payoff_part
)
```

```
[1] "MDE Payoff part: 99.68 (9.4 % of mean)"
```

```r
dyads <- tparticipants %>%
  group_by(experiment, session_code, group_id) %>%
  summarise(sum_payoff = sum(payoff),.groups = "drop")

mn_payoff_dyads <- mean(dyads$sum_payoff)
sd_payoff_dyads <- sd(dyads$sum_payoff)
sprintf("Mean (SD) of dyad payoff: %.2f (%.2f)", mn_payoff_dyads, sd_payoff_dyads)
```

```
[1] "Mean (SD) of dyad payoff: 2126.10 (262.65)"
```

```r
pwr.t.test(50, (0.1*mn_payoff_dyads)/sd_payoff_dyads)
```

```

     Two-sample t test power calculation

              n = 50
              d = 0.8094766
      sig.level = 0.05
          power = 0.9796931
    alternative = two.sided

NOTE: n is number in *each* group
```

```r
pr <- pwr.t.test(n = 50, power = 0.8)
sprintf(
  "MDE Payoff dyads: %.2f (%.1f %% of mean)", pr$d * sd_payoff_dyads,
  100*(pr$d * sd_payoff_dyads)/mn_payoff_dyads
)
```

```
[1] "MDE Payoff dyads: 148.62 (7.0 % of mean)"
```

Simulation for regression based tests

```r
if (file.exists("../data/generated/trust_sim_results.csv")) {
  trust_sim_results <- read_csv("../data/generated/trust_sim_results.csv", show_col_types
} else {
```

```r
sim_data <- function(parms, runs = 50, rounds = 10) {
  cl <- function(val, vmin = 0, vmax = 100) {
    if (val > vmax) return(as.integer(vmax))
    if (val < vmin) return(as.integer(vmin))
    as.integer(round(val))
  }
  tr <- function(rd, exp, g, parms) {
    tsent_start = ifelse(
      exp == "ftrust",
      parms$tsent_start + parms$tsent_start_teffect,
      parms$tsent_start
    )
    tsent_grate = ifelse(
      exp == "ftrust",
      parms$tsent_grate + parms$tsent_grate_teffect,
      parms$tsent_grate
    )
    tpct_returned_start = ifelse(
      exp == "ftrust",
      parms$tpct_returned_start + parms$tpct_returned_start_teffect,
      parms$tpct_returned_start
    )
    tpct_returned_grate = ifelse(
      exp == "ftrust",
      parms$tpct_returned_grate + parms$tpct_returned_grate_teffect,
      parms$tpct_returned_grate
    )

    tibble(
      experiment = factor(ifelse(
        exp == "ftrust",
        "Business Framing", "Neutral Framing"
      ), c("Neutral Framing", "Business Framing")),
      group_id = g,
      round = rd,
      sent_amount = cl(
        tsent_start + (rd-1)*tsent_grate + rnorm(1, 0, parms$tsent_evar),
        0, 100
      ),
      sent_back_amount = cl(
        3*(tpct_returned_start + (rd-1)*tpct_returned_grate +
```

```
            rnorm(1, 0, parms$tpct_returned_evar))*sent_amount ,
        0, 3*sent_amount
      ),
      pct_returned = sent_back_amount/(3*sent_amount)
    )
  }
  bind_rows(
    lapply(
      c("trust", "ftrust"),
      function(e) bind_rows(
        lapply(
          1:runs,
          function(g) bind_rows(lapply(1:rounds, tr, e, g, parms))
        )
      )
    )
  )
}

run_trust_sim <- function(te) {
  parms <- tibble(
    tsent_start = mn_sent_start,
    tsent_start_teffect = te$teffect_sent,
    tsent_grate = coef(ols_sent_amount)[2],
    tsent_grate_teffect = te$teffect_sent_grate,
    tsent_evar = sd_sent,
    tpct_returned_start = mn_pct_returned_start,
    tpct_returned_start_teffect = te$teffect_pct_returned,
    tpct_returned_grate = 0,
    tpct_returned_grate_teffect = te$teffect_pct_returned_grate,
    tpct_returned_evar = sd_pct_returned
  )
  smp <- sim_data(parms)
  ci_trust_sent_fe <- confint(
    feols(sent_amount ~ experiment | round, cluster = c("round", "group_id"), data = smp
  )
  ci_trust_sent_round_fe <- confint(
    feols(sent_amount ~ experiment*round, cluster = c("round", "group_id"), data = smp)
  )
  ci_trust_pct_returned_fe <- confint(feols(
    pct_returned ~ experiment | round, cluster = c("round", "group_id"),
```

```
    data = smp %>% filter(sent_amount > 0)
  ))
  ci_trust_pct_returned_round_fe <- confint(feols(
    pct_returned ~ experiment*round, cluster = c("round", "group_id"),
    data = smp %>% filter(sent_amount > 0)
  ))
  tibble(
    sent_teffect_lb = pull(ci_trust_sent_fe[1]),
    sent_teffect_ub = pull(ci_trust_sent_fe[2]),
    sent_round_teffect_lb = ci_trust_sent_round_fe[4, 1],
    sent_round_teffect_ub = ci_trust_sent_round_fe[4, 2],
    pct_returned_teffect_lb = pull(ci_trust_pct_returned_fe[1]),
    pct_returned_teffect_ub = pull(ci_trust_pct_returned_fe[2]),
    pct_returned_round_teffect_lb = ci_trust_pct_returned_round_fe[4, 1],
    pct_returned_round_teffect_ub = ci_trust_pct_returned_round_fe[4, 2],
  )
}

sim_power_trust <- function(plan) {
  sim_results <- bind_rows(
    lapply(
      1:nrow(plan),
      function(x) {
        message(
          sprintf("Running trust sim, plan row %d of %d...", x, nrow(plan)),
          appendLF = F
        )
        rv <- bind_cols(plan[x,], run_trust_sim(plan[x,]))
        message("")
        rv
      }
    )
  )
}

plan <- bind_rows(
  expand_grid(
    n = 1:100,
    teffect_sent = 1:5,
    teffect_sent_grate = 0,
    teffect_pct_returned = c(0.005, 0.01, 0.015, 0.02, 0.025),
```

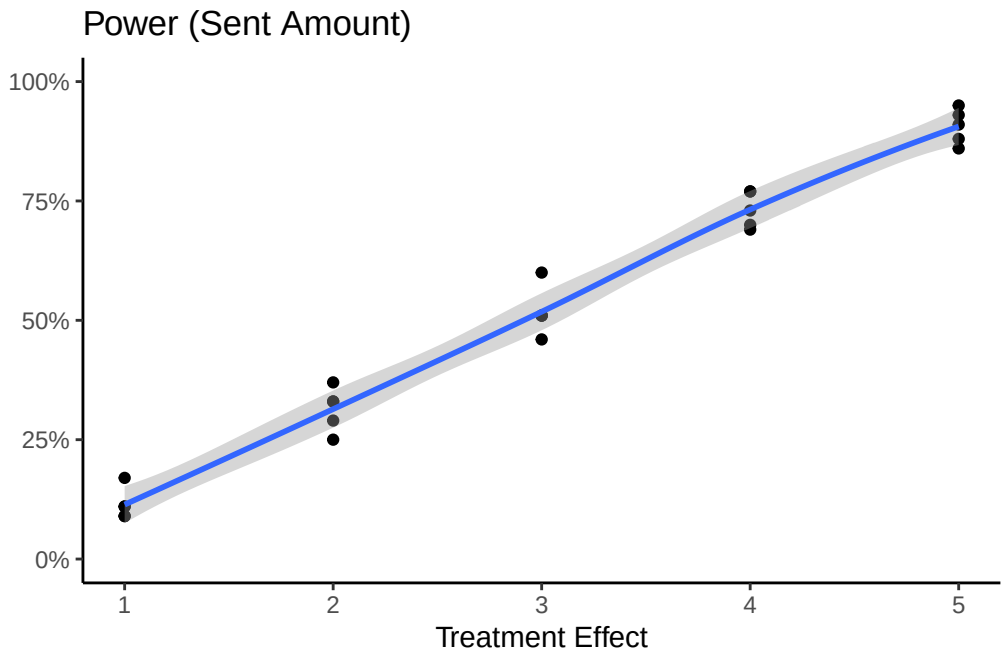
```
      teffect_pct_returned_grate = 0,
    ),
    expand_grid(
      n = 1:100,
      teffect_sent = 0,
      teffect_sent_grate = c(0.5, 1, 1.5, 2, 2.5),
      teffect_pct_returned = 0,
      teffect_pct_returned_grate = c(0.01, 0.02, 0.03, 0.04, 0.05)/5,
    )
  )

  message(sprintf("Starting trust power simulations (%d runs): %s", nrow(plan), Sys.time()))
  trust_sim_results <- sim_power_trust(plan)
  write_csv(trust_sim_results, "../data/generated/trust_sim_results.csv")
  message(sprintf("Done: %s", Sys.time()))
}

trust_power <- trust_sim_results %>%
  group_by(
    teffect_sent, teffect_sent_grate,
    teffect_pct_returned, teffect_pct_returned_grate
  ) %>%
  summarise(
    power_sent = mean(sent_teffect_lb > 0),
    power_sent_round = mean(sent_round_teffect_lb > 0),
    power_pct_returned = mean(pct_returned_teffect_lb > 0),
    power_pct_returned_round = mean(pct_returned_round_teffect_lb > 0),
    .groups = "drop"
  )

ggplot(
  trust_power %>% filter(teffect_sent_grate == 0),
  aes(x = teffect_sent, y = power_sent)
) + geom_point() + geom_smooth() +
  scale_y_continuous(limits = c(0, 1), labels = scales::percent) +
  labs(title = "Power (Sent Amount)", x = "Treatment Effect", y = "") +
  theme_classic()
```

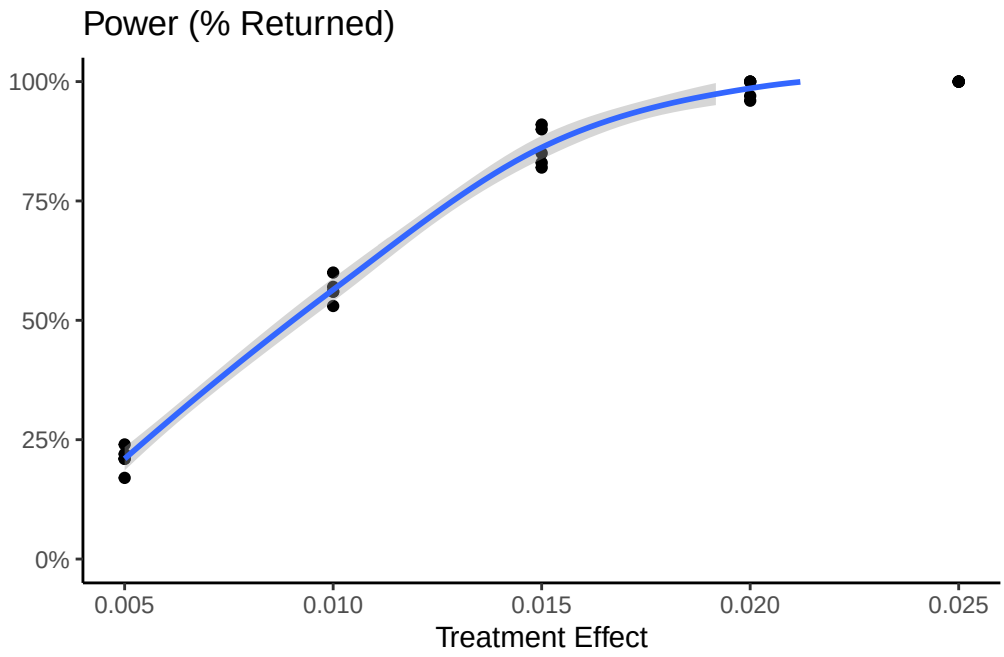
```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```




```
ggplot(
  trust_power %>% filter(teffect_sent_grate == 0),
  aes(x = teffect_pct_returned, y = power_pct_returned)
) + geom_point() + geom_smooth() +
  scale_y_continuous(limits = c(0, 1), labels = scales::percent) +
  labs(title = "Power (% Returned)", x = "Treatment Effect", y = "") +
  theme_classic()
```

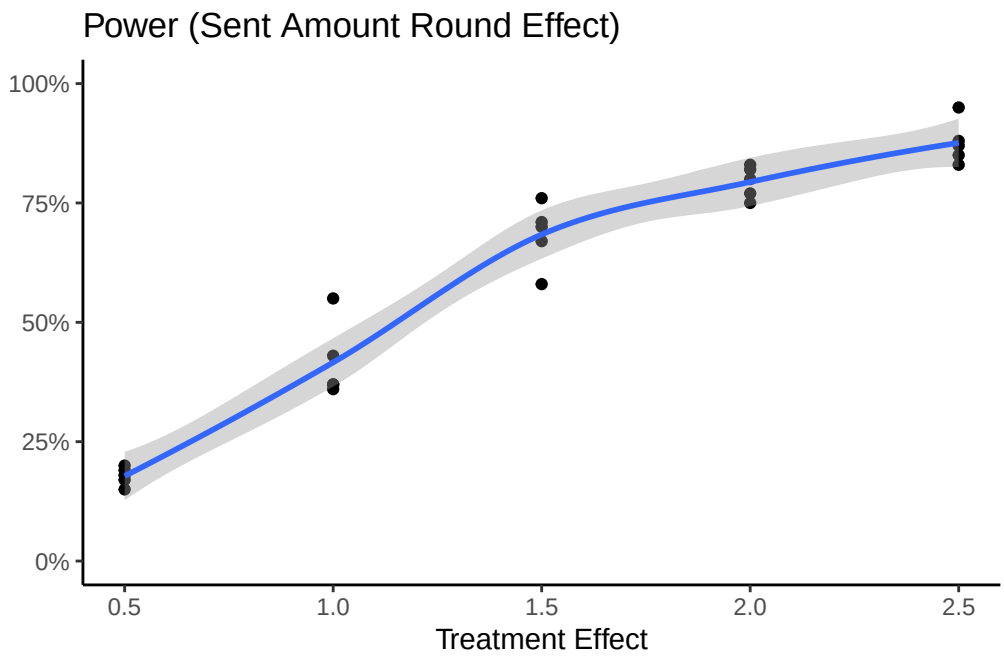
```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

```
Warning: Removed 15 rows containing missing values (`geom_smooth()`).
```




```
ggplot(
  trust_power %>% filter(teffect_sent == 0),
  aes(x = teffect_sent_grate, y = power_sent_round)
) + geom_point() + geom_smooth() +
  scale_y_continuous(limits = c(0, 1), labels = scales::percent) +
  labs(title = "Power (Sent Amount Round Effect)", x = "Treatment Effect", y = "") +
  theme_classic()
```

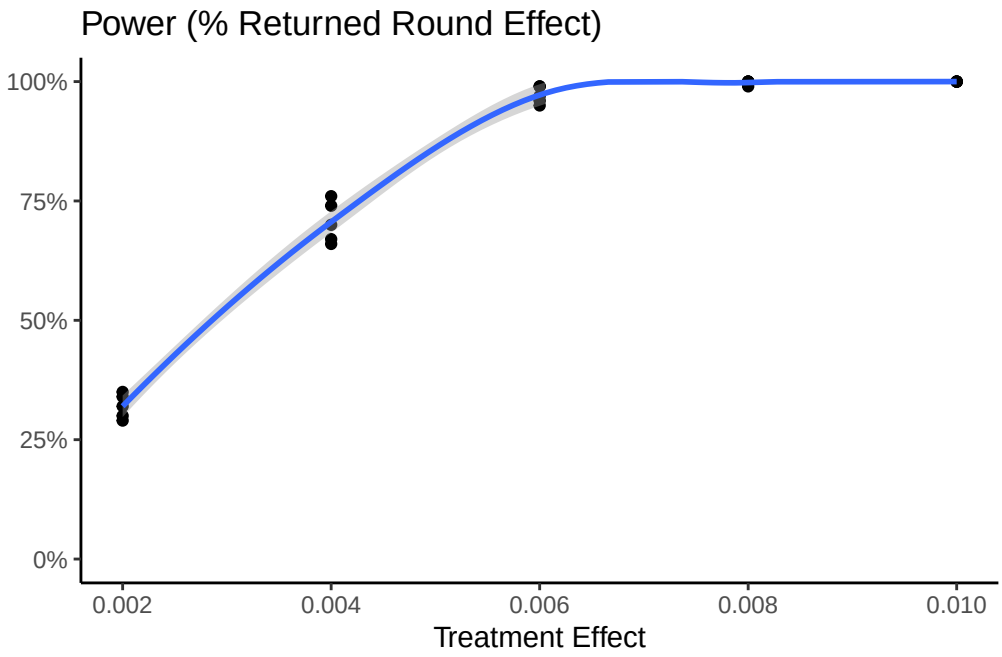
```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```




```
ggplot(
  trust_power %>% filter(teffect_sent == 0),
  aes(x = teffect_pct_returned_grate, y = power_pct_returned_round)
) + geom_point() + geom_smooth() +
  scale_y_continuous(limits = c(0, 1), labels = scales::percent) +
  labs(title = "Power (% Returned Round Effect)", x = "Treatment Effect", y = "") +
  theme_classic()
```

```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

```
Warning: Removed 22 rows containing missing values (`geom_smooth()`).
```


Power (% Returned Round Effect)
100%
75%
50%
25%
0%
0.002
0.004
0.006
0.008
0.010
Treatment Effect