# Causal Inference with `NNS.stack`: A Transparent Workflow on mtcars (with All the Caveats)

## Causal Inference with `NNS.stack` on the mtcars Dataset

### Transmission Effects on Fuel Efficiency

This document presents a causal-inference workflow for the `mtcars` dataset using `NNS.stack` as the outcome-modeling engine. The treatment is transmission type, `am`, the outcome is miles per gallon, `mpg`, and the observed pre-treatment covariates are weight (`wt`), horsepower (`hp`), and displacement (`disp`).

The workflow is intentionally simple and internally consistent:

1. fit the treated and control outcome surfaces once with `NNS.stack`,
2. generate counterfactual predictions once for the full sample,
3. construct the fitted treatment-effect distribution,
4. summarize ATE, ATT, ATC, and heterogeneous effects from that fitted contrast distribution, and
5. use `NNS.meboot` on the estimated treatment-effect distribution to construct interval summaries conditional on the learned outcome surfaces.

This is not an application of `NNS.caus`. The goal here is response-scale causal estimation in outcome units, not directional causal signaling.

```
library(NNS)
```

## 1. Causal Question and Estimands

Let:

- `A = am` denote treatment, where `A = 1` is manual transmission and `A = 0` is automatic transmission,
- `Y = mpg` denote fuel efficiency,
- `X = (wt, hp, disp)` denote observed pre-treatment covariates.

The causal question is:

> What is the causal effect of manual transmission on miles per gallon, after adjusting for observed pre-treatment covariates?

The estimands of interest are:

- **Average Treatment Effect (ATE)**: `E[Y(1) - Y(0)]`
- **Average Treatment Effect on the Treated (ATT)**: `E[Y(1) - Y(0) | A = 1]`
- **Average Treatment Effect on the Controls (ATC)**: `E[Y(1) - Y(0) | A = 0]`
- **Conditional Average Treatment Effect (CATE)**: `E[Y(1) - Y(0) | X = x]`

The fitted conditional treatment effect is

`tau_hat(x) = m1_hat(x) - m0_hat(x)`

where `m1_hat(x)` and `m0_hat(x)` are the estimated treated and control outcome surfaces.

## 2. Identification Assumptions

The causal interpretation requires the usual observational assumptions.

### 2.1 Consistency

For each unit, the observed outcome equals the potential outcome under the observed treatment.

### 2.2 No Unmeasured Confounding

Conditional on the observed pre-treatment covariates, treatment assignment is assumed independent of the potential outcomes.

### 2.3 Positivity / Overlap

For the covariate profiles under study, both treatment states must remain possible.

These assumptions are substantive. They are not guaranteed by the modeling procedure alone.

## 3. Data Setup

```
data(mtcars)

dat <- mtcars

Y <- dat$mpg
A <- dat$am

X <- data.frame(
  wt   = dat$wt,
  hp   = dat$hp,
  disp = dat$disp
)

analysis_dat <- data.frame(
  mpg   = Y,
  am    = A,
  wt    = X$wt,
  hp    = X$hp,
  disp  = X$disp
)

str(analysis_dat)
```

```
## 'data.frame':    32 obs. of  5 variables:
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ disp: num  160 160 108 258 360 ...
```

```
summary(analysis_dat$am)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.0000  0.4062  1.0000  1.0000
```

The sample is small and the treatment split is unbalanced, so any causal interpretation will necessarily depend on strong modeling assumptions and careful attention to overlap.

## 4. Propensity Score Model and Design Diagnostics

A causal analysis should examine overlap explicitly. For transparency, a logistic regression propensity-score model is used here.

```
ps_model <- glm(am ~ wt + hp + disp, data = analysis_dat, family = binomial())

analysis_dat$ps <- as.numeric(predict(ps_model, type = "response"))

summary(analysis_dat$ps)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.000000 0.005349 0.062641 0.406250 0.960551 0.999937
```

The estimated propensity scores already suggest a design problem: the treatment is highly predictable from the covariates, which is helpful for classification but harmful for causal identification because it reduces common support.

### 4.1 Groupwise Propensity Summary

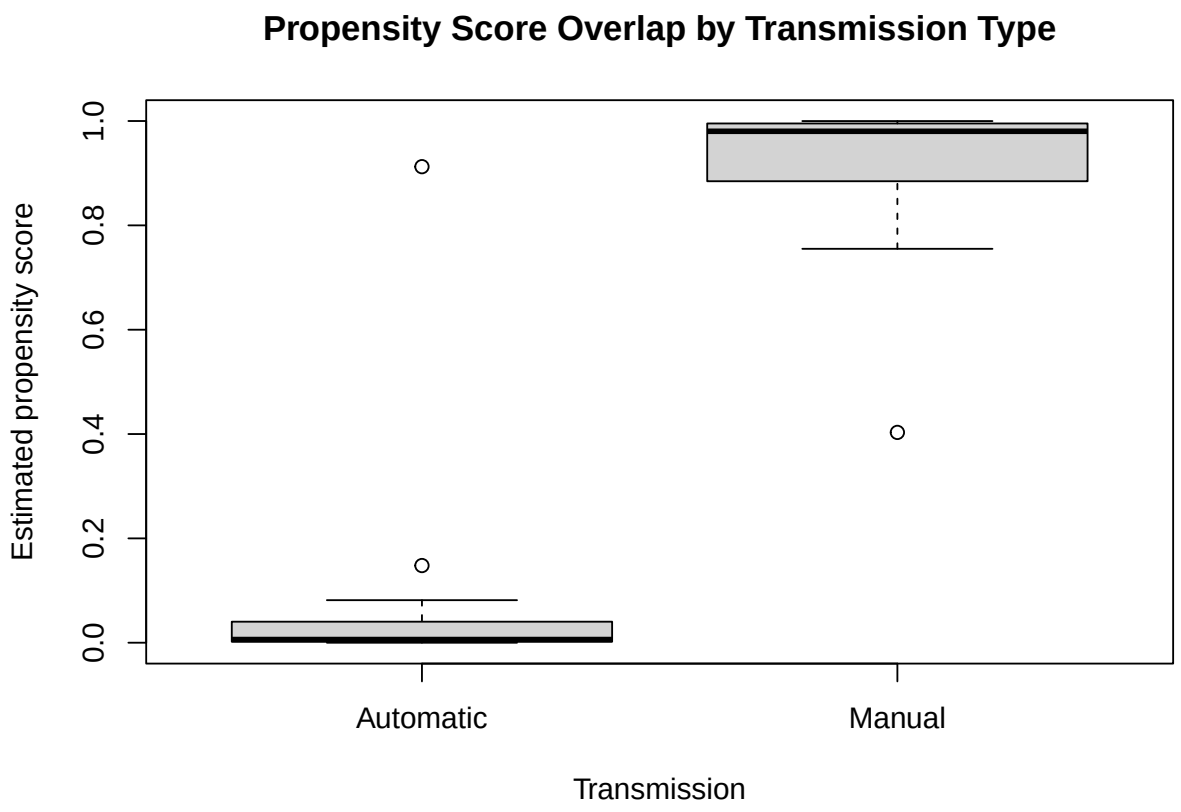
```
ps_by_group <- split(analysis_dat$ps, analysis_dat$am)

overlap_summary <- data.frame(
  Group  = c("Automatic", "Manual"),
  Min    = c(min(ps_by_group[["0"]]), min(ps_by_group[["1"]])),
  Q1     = c(quantile(ps_by_group[["0"]], 0.25), quantile(ps_by_group[["1"]], 0.25)),
  Median = c(median(ps_by_group[["0"]]), median(ps_by_group[["1"]])),
  Mean   = c(mean(ps_by_group[["0"]]), mean(ps_by_group[["1"]])),
  Q3     = c(quantile(ps_by_group[["0"]], 0.75), quantile(ps_by_group[["1"]], 0.75)),
  Max    = c(max(ps_by_group[["0"]]), max(ps_by_group[["1"]]))
)

overlap_summary
```

```
##       Group          Min          Q1      Median       Mean         Q3
## 1 Automatic 2.911885e-08 0.001944826 0.006101649 0.06943628 0.04029405
## 2    Manual 4.031203e-01 0.884557352 0.980314005 0.89851620 0.99529516
##         Max
## 1 0.9125089
## 2 0.9999370
```

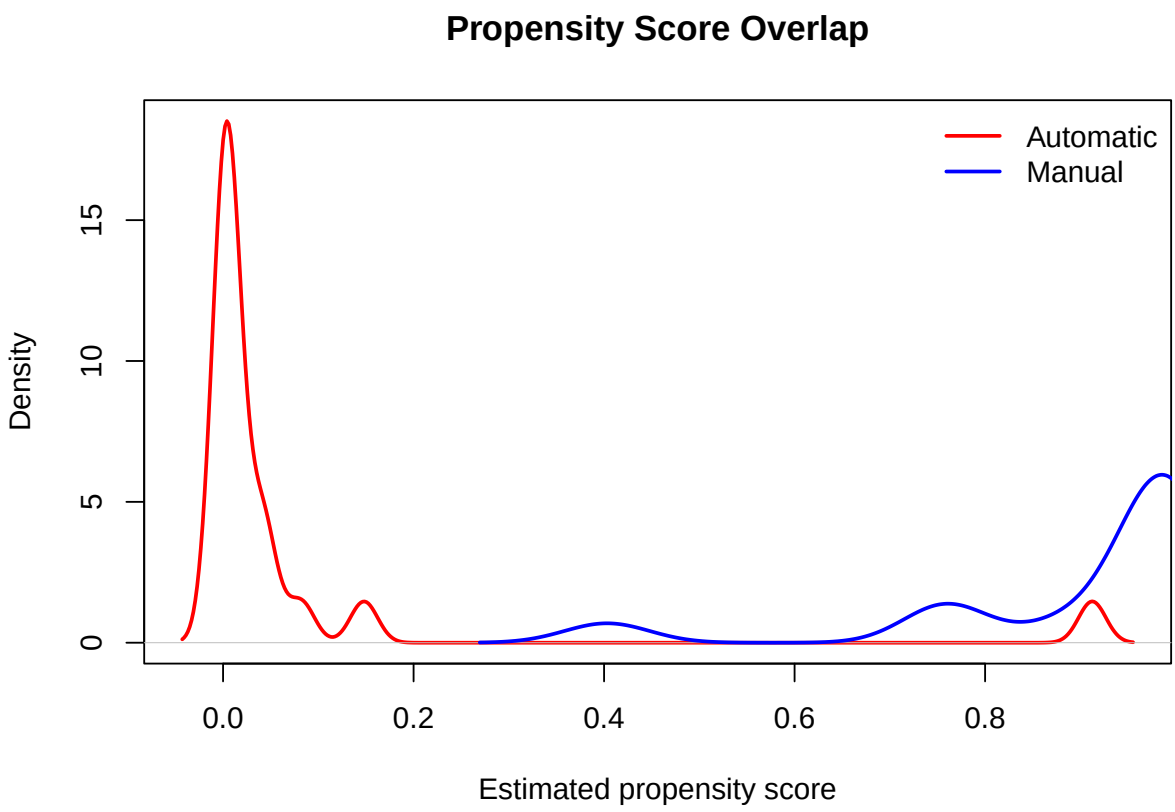
In causal inference, overlap means that both treatment states are plausible at similar covariate values. Strong separation between the propensity-score distributions is therefore a warning sign, because treatment effects then rely more on extrapolation than on direct treated-versus-control comparison.

### 4.2 Visual Overlap Diagnostics

```
boxplot(ps ~ am,
        data = analysis_dat,
        names = c("Automatic", "Manual"),
        xlab = "Transmission",
        ylab = "Estimated propensity score",
        main = "Propensity Score Overlap by Transmission Type")
```




```r
plot(density(analysis_dat$ps[analysis_dat$am == 0]),
     main = "Propensity Score Overlap",
     xlab = "Estimated propensity score",
     col = "red",
     lwd = 2)
lines(density(analysis_dat$ps[analysis_dat$am == 1]),
      col = "blue",
      lwd = 2)
legend("topright",
       legend = c("Automatic", "Manual"),
       col = c("red", "blue"),
       lwd = 2,
       bty = "n")
```




The overlap plots make the same point visually: automatic and manual cars occupy quite different regions of covariate space. In `mtcars`, this is not a correctable design flaw so much as a structural property of the underlying data-generating process. Manual and automatic transmissions were associated with fundamentally different types of vehicles: manuals tended to appear in lighter, smaller, sport-oriented cars, while automatics were more common in heavier, larger, family-oriented cars. As a result, the treatment assignment mechanism is not merely imbalanced but nearly built into the product categories themselves.

Seen in that light, the overlap problem should be understood as a property of the scientific question rather than simply a weakness of the analysis. Trimming would recover only a narrow and poorly representative region of common support, yielding a more local but less substantively meaningful estimand. The overlap failure here is not a correctable design flaw — it reflects the absence of genuine experimental variation in the data-generating process. Any causal estimate from `mtcars` on this question is fundamentally an extrapolation, and the choice of estimator determines the shape of that extrapolation, not whether it occurs.

### 4.3 Compact Balance Diagnostics

A compact balance table keeps the design stage visible. Standardized mean differences are shown before and after inverse-probability weighting.

```
w_ate <- ifelse(analysis_dat$am == 1,
                1 / analysis_dat$ps,
                1 / (1 - analysis_dat$ps))

weighted_mean <- function(x, w) sum(w * x) / sum(w)
weighted_var  <- function(x, w) sum(w * (x - weighted_mean(x, w))^2) / sum(w)

smd_unweighted <- function(x, a) {
```

```
  x1 <- x[a == 1]
  x0 <- x[a == 0]
  (mean(x1) - mean(x0)) / sqrt((var(x1) + var(x0)) / 2)
}

smd_weighted <- function(x, a, w) {
  x1 <- x[a == 1]
  x0 <- x[a == 0]
  w1 <- w[a == 1]
  w0 <- w[a == 0]
  (weighted_mean(x1, w1) - weighted_mean(x0, w0)) /
    sqrt((weighted_var(x1, w1) + weighted_var(x0, w0)) / 2)
}

balance_table <- data.frame(
  Covariate = colnames(X),
  SMD_Unweighted = sapply(X, smd_unweighted, a = A),
  SMD_IPW = sapply(X, smd_weighted, a = A, w = w_ate)
)

balance_table
```

```
##      Covariate SMD_Unweighted    SMD_IPW
## wt          wt     -1.9349029 -1.1427267
## hp          hp     -0.4732373 -0.2005179
## disp      disp     -1.4780339 -0.8336086
```

The weighted balance table shows some improvement, but substantial residual imbalance remains, especially for `wt` and `disp`. This is another reason to interpret the resulting treatment effects as illustrative rather than definitive.

## 5. Linear Benchmark

Before fitting the nonlinear outcome models, it is useful to establish a familiar linear benchmark.

```
fit_ols <- lm(mpg ~ am + wt + hp + disp, data = analysis_dat)
summary(fit_ols)
```

```
##
## Call:
## lm(formula = mpg ~ am + wt + hp + disp, data = analysis_dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4590 -1.6900 -0.3708  1.1301  5.5011
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 34.209443   2.822826  12.119 1.98e-12 ***
## am           2.159271   1.435176   1.505  0.14405
## wt          -3.046747   1.157119  -2.633  0.01383 *
## hp          -0.039323   0.012434  -3.163  0.00384 **
## disp         0.002489   0.010377   0.240  0.81222
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 2.581 on 27 degrees of freedom
## Multiple R-squared:  0.8402, Adjusted R-squared:  0.8166
## F-statistic:  35.5 on 4 and 27 DF,  p-value: 2.181e-10
```

```
coef_ci <- confint(fit_ols)["am", ]
ols_summary <- data.frame(
  Estimand = "OLS coefficient on am",
  Estimate = coef(fit_ols)["am"],
  Lower_95 = coef_ci[1],
  Upper_95 = coef_ci[2]
)

ols_summary
```

```
##                  Estimand Estimate   Lower_95 Upper_95
## am OLS coefficient on am 2.159271 -0.7854664 5.104008
```

The OLS coefficient provides a single homogeneous average contrast. It is a useful benchmark, but it cannot represent the heterogeneous treatment surface that the NNS workflow is designed to estimate.

## 6. Fit `NNS.stack` Outcome Surfaces Once

Fit separate stacked outcome models in the treated and control arms. Here `method = 1` is used so that the full multivariate covariate set is retained rather than routed through the dimension-reduction regression path.

```
X_treat   <- X[A == 1, , drop = FALSE]
Y_treat   <- Y[A == 1]

X_control <- X[A == 0, , drop = FALSE]
Y_control <- Y[A == 0]

stack_treat <- NNS.stack(
  IVs.train = X_treat,
  DV.train = Y_treat,
  IVs.test = X,
  method = 1,
  folds = 5,
  stack = TRUE,
  status = FALSE
)

stack_control <- NNS.stack(
  IVs.train = X_control,
  DV.train = Y_control,
  IVs.test = X,
  method = 1,
  folds = 5,
  stack = TRUE,
  status = FALSE
)
```

The treated and control models estimate the two potential-outcome surfaces:

- `m1(x) = E[Y | A = 1, X = x]`
- `m0(x) = E[Y | A = 0, X = x]`

```r
stack_tuning <- data.frame(
  Model = c("Manual surface", "Automatic surface"),
  NNS_reg_n_best = c(stack_treat$NNS.reg.n.best, stack_control$NNS.reg.n.best)
)

stack_tuning
```

```
##               Model NNS_reg_n_best
## 1    Manual surface              1
## 2 Automatic surface              1
```

The tuning results indicate how local each arm-specific surface is. In this example, the chosen neighborhood sizes are quite small, which is consistent with a flexible and locally adaptive fit.

## 7. Generate Counterfactual Predictions

Generate both potential-outcome predictions for the full observed covariate matrix and form the fitted treatment-effect distribution.

```r
mu1_hat <- stack_treat$stack
mu0_hat <- stack_control$stack

tau_hat <- mu1_hat - mu0_hat
```

## 8. Summarize Causal Effects from the Fitted Contrast Distribution

### 8.1 Sample-Wide Causal Estimands

```r
ATE_hat <- mean(tau_hat)
ATT_hat <- mean(tau_hat[A == 1])
ATC_hat <- mean(tau_hat[A == 0])

estimand_table <- data.frame(
  Estimand = c("ATE", "ATT", "ATC"),
  Estimate = c(ATE_hat, ATT_hat, ATC_hat)
)

estimand_table
```

```
##   Estimand    Estimate
## 1      ATE  1.41142173
## 2      ATT  3.55169533
## 3      ATC -0.05297599
```

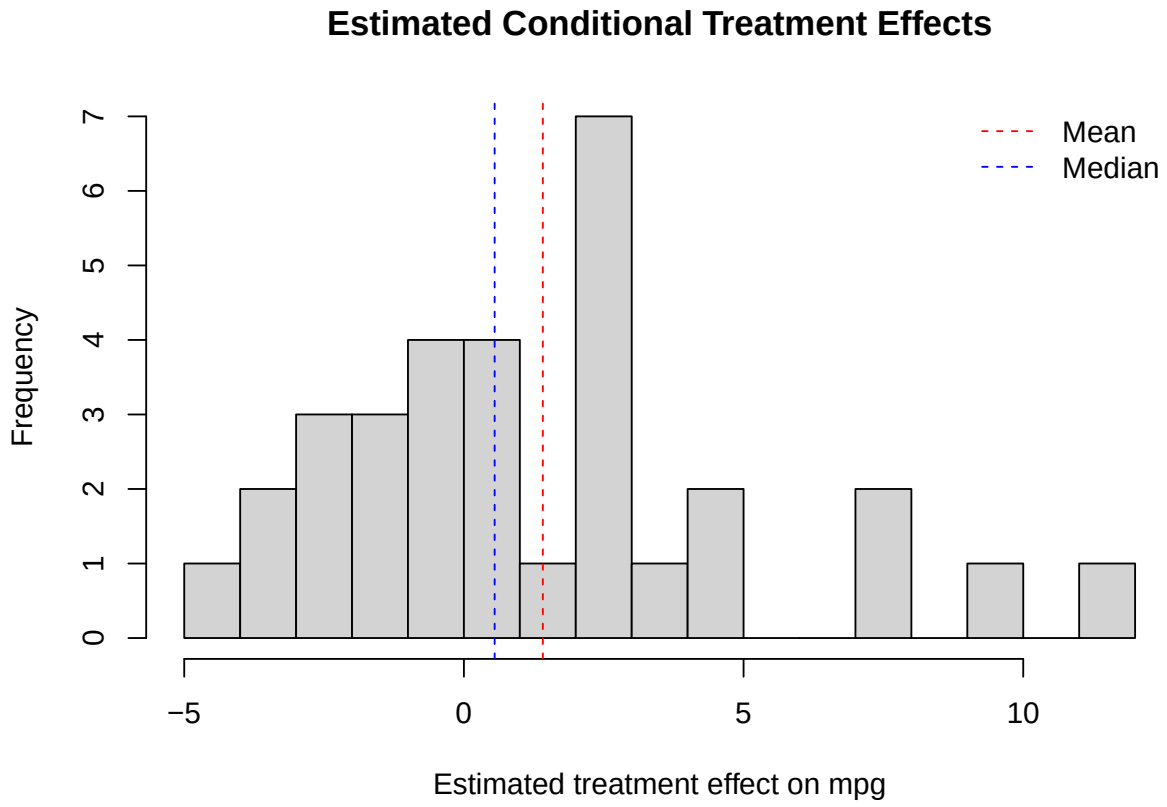
The fitted treatment-effect distribution implies a positive average effect of manual transmission on `mpg`, with the largest gains concentrated in the treated support. The much larger ATT relative to the ATC indicates that the estimated benefit depends strongly on the region of covariate space being evaluated.

### 8.2 Distribution of Estimated Individual Effects

```r
summary(tau_hat)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -4.776  -1.150   0.550   1.411   2.873  11.044
```

```r
hist(tau_hat,
     main = "Estimated Conditional Treatment Effects",
     xlab = "Estimated treatment effect on mpg",
     breaks = 12)
abline(v = mean(tau_hat), lty = 2, col = "red")
abline(v = median(tau_hat), lty = 2, col = "blue")
legend("topright",
       legend = c("Mean", "Median"),
       col = c("red", "blue"),
       lty = 2,
       bty = "n")
```




The distribution of `tau_hat` shows that the estimated treatment effect is heterogeneous rather than constant. Although the center of the distribution is positive, some covariate profiles still receive negative fitted effects, which is consistent with a nonlinear and support-dependent treatment surface.

The causal estimands are all being summarized from the same fitted effect distribution. This keeps the workflow internally consistent.

## 9. Reference-Point Contrast

A benchmark contrast at a fixed covariate profile can still be useful. Here the benchmark is defined at median covariate values.

```r
x_ref <- as.data.frame(lapply(X, median))
```

```r
mu1_ref <- NNS.stack(
  IVs.train = X_treat,
  DV.train = Y_treat,
  IVs.test = x_ref,
  method = 1,
  folds = 5,
  stack = TRUE,
  status = FALSE
)$stack

mu0_ref <- NNS.stack(
  IVs.train = X_control,
  DV.train = Y_control,
  IVs.test = x_ref,
  method = 1,
  folds = 5,
  stack = TRUE,
  status = FALSE
)$stack

ref_contrast <- mu1_ref - mu0_ref

benchmark_table <- data.frame(
  Contrast = "Reference-point contrast at median covariates",
  Estimate = ref_contrast
)

benchmark_table
```

```
##                                       Contrast Estimate
## 1 Reference-point contrast at median covariates      2.5
```

The benchmark contrast is a valid local summary, but it should not be treated as the sole causal estimate. The sample-wide ATE, ATT, ATC, and the full `tau_hat` distribution remain the main inferential objects.

## 10. Doubly Robust Comparison

To reduce dependence on the outcome model alone, combine the stacked outcome surfaces with the propensity model using AIPW.

```r
eps <- 0.01
ps_trim <- pmin(pmax(analysis_dat$ps, eps), 1 - eps)

AIPW_scores <- (mu1_hat - mu0_hat) +
  A * (Y - mu1_hat) / ps_trim -
  (1 - A) * (Y - mu0_hat) / (1 - ps_trim)

AIPW_ATE <- mean(AIPW_scores)

aipw_table <- data.frame(
  Estimand = c("ATE (G-computation)", "ATE (AIPW)"),
  Estimate = c(ATE_hat, AIPW_ATE)
)

aipw_table
```

```
##              Estimand Estimate
## 1 ATE (G-computation) 1.411422
## 2          ATE (AIPW) 1.668820
```

The AIPW estimate is directionally consistent with the g-computation estimate, which provides a useful robustness check. In a small and noisy sample like `mtcars`, agreement in sign is more informative than agreement in exact magnitude.

## 11. Bootstrap the Estimated Effect Distribution

For this workflow, the bootstrap is attached to the estimated treatment-effect distribution itself rather than to repeated re-estimation of `NNS.stack`. This is computationally much lighter and keeps the inferential target aligned with the fitted contrast object.

```r
set.seed(123)

B <- 1000
alpha <- 0.05

boot_tau <- NNS.meboot(
  x = tau_hat,
  reps = B,
  rho = 1,
  drift = FALSE
)

boot_tau_mat <- boot_tau["replicates", ]$replicates

boot_ate <- apply(boot_tau_mat, 2, mean)
boot_att <- apply(boot_tau_mat[A == 1, , drop = FALSE], 2, mean)
boot_atc <- apply(boot_tau_mat[A == 0, , drop = FALSE], 2, mean)
boot_med <- apply(boot_tau_mat, 2, median)
```

```r
ate_ci <- c(
  LPM.VaR(alpha / 2, 0, boot_ate),
  UPM.VaR(alpha / 2, 0, boot_ate)
)

att_ci <- c(
  LPM.VaR(alpha / 2, 0, boot_att),
  UPM.VaR(alpha / 2, 0, boot_att)
)

atc_ci <- c(
  LPM.VaR(alpha / 2, 0, boot_atc),
  UPM.VaR(alpha / 2, 0, boot_atc)
)

med_ci <- c(
  LPM.VaR(alpha / 2, 0, boot_med),
  UPM.VaR(alpha / 2, 0, boot_med)
)

causal_summary_table <- data.frame(
  Estimand = c("ATE (G-computation)",
```

```
               "ATT (G-computation)",
               "ATC (G-computation)",
               "ATE (AIPW)",
               "Reference-point contrast",
               "Median CATE"),
  Estimate = c(ATE_hat,
               ATT_hat,
               ATC_hat,
               AIPW_ATE,
               ref_contrast,
               median(tau_hat)),
  Lower_95 = c(ate_ci[1],
               att_ci[1],
               atc_ci[1],
               NA,
               NA,
               med_ci[1]),
  Upper_95 = c(ate_ci[2],
               att_ci[2],
               atc_ci[2],
               NA,
               NA,
               med_ci[2])
)

causal_summary_table
```

```
##                   Estimand    Estimate     Lower_95  Upper_95
## 1      ATE (G-computation)  1.41142173  1.178679666 1.6441637
## 2      ATT (G-computation)  3.55169533  2.340121029 3.5811884
## 3      ATC (G-computation) -0.05297599 -0.001657587 0.5941207
## 4               ATE (AIPW)  1.66882018           NA        NA
## 5 Reference-point contrast  2.50000000           NA        NA
## 6             Median CATE  0.55000000  0.208244250 2.3279959
```

```
bootstrap_counts <- data.frame(
  Requested_Replicates = B,
  Usable_ATE = length(boot_ate),
  Usable_ATT = length(boot_att),
  Usable_ATC = length(boot_atc),
  Usable_Median = length(boot_med)
)

bootstrap_counts
```

```
##   Requested_Replicates Usable_ATE Usable_ATT Usable_ATC Usable_Median
## 1                 1000       1000       1000       1000          1000
```

These interval summaries are attached to the estimated treatment-effect distribution itself, conditional on the fitted `NNS.stack` outcome surfaces. They should therefore be read as uncertainty around the learned causal contrast distribution, not as full model-refit uncertainty intervals.

## 12. Interpretation

Under consistency, no unmeasured confounding given the observed covariates, and sufficient overlap, this workflow estimates the fuel-efficiency effect of manual transmission through arm-specific `NNS.stack` outcome surfaces and standardization.

The workflow is deliberately consistent:

- one manual outcome surface,
- one automatic outcome surface,
- one fitted effect distribution,
- one set of causal summaries derived from that distribution,
- and one bootstrap layer attached directly to that estimated effect object.

For `mtcars`, the main substantive message is a positive fitted effect on average, but with clear dependence on covariate support and substantial design limitations. More precisely, this is an extrapolation-dependent causal exercise: the outcome-modeling step cannot avoid projecting treatment contrasts into regions where one arm is only weakly represented. The value of the example is therefore not that it delivers a high-credibility causal estimate for transmission in `mtcars`, but that it makes transparent how a nonlinear outcome-modeling workflow behaves when the scientific question itself is only weakly supported by common empirical variation.

## 13. Limitations

This remains an observational analysis. The main limitations are:

- The no-unmeasured-confounding assumption is substantive and not testable from the data alone.
- The propensity model may be misspecified.
- The bootstrap intervals are conditional on the fitted `NNS.stack` outcome surfaces rather than full model-refit uncertainty.
- Overlap and balance should still be examined carefully before making strong substantive claims.
- In a formal empirical paper, richer balance diagnostics, trimming, and sensitivity analysis would likely be added.

## 14. Conclusion

This document shows how `NNS.stack` can be used in a clean causal-inference workflow on the `mtcars` dataset.

The central idea is straightforward: fit the manual and automatic outcome surfaces once, generate counterfactual predictions, construct the fitted treatment-effect distribution, summarize causal estimands from that distribution, and then attach interval summaries to the estimated effect object using `NNS.meboot`.

That keeps the causal design explicit, preserves the nonlinear and heterogeneous strengths of the NNS framework, and avoids the unnecessary computational burden of repeatedly refitting the full learner inside every bootstrap replicate. In this dataset, the estimator does not eliminate extrapolation; it determines the form of the extrapolation.