Estimating the Variance of IP-Weighted Risk Ratios Using Influence Functions

Chong Zhang, Dan Scharfstein, Yizhe Xu*

2026-01-08

Contents

*Corresponding author: yizhe.xu@hsc.utah.edu

Introduction and Goal

This tutorial provides a complete implementation of variance estimation for an intention-to-treat effect estimated as inverse probability weighted (IPW) risk ratio using influence functions, where the risk is the subdistribution cumulative incidence function (CIF) to account for the competing risk of death.

The Problem

When estimating treatment effects using IPW, we face a **two-step estimation problem**:

1. **Step 1:** Estimate propensity scores from a logistic regression model
2. **Step 2:** Use these estimated propensity scores as weights to compute the CIF

Because the propensity scores are estimated (not known), their uncertainty must propagate into the variance of the CIF. The influence function approach provides an elegant analytical solution.

Why Influence Functions?

- **Computational efficiency:** Avoids bootstrapping, which is slow for large datasets
- **Analytical insight:** Clearly separates sources of uncertainty
- **Correct inference:** Properly accounts for the two-step estimation

Since $h^*(t) = h_{dementia}(t) + h_{death}(t)$, we only need:

- $\psi_{h_{dementia}}$ for the direct effect on CIF
- ψ_{h^*} for the indirect effect through survival

Notation and Setup

Key Notation

Symbol	Description
A_i	Treatment indicator (1 = treated, 0 = control)
X_i	Vector of baseline covariates (including intercept)
T_i	Observed event time
Δ_i	Event indicator (0 = censored, 1 = dementia, 2 = death)
γ	Propensity score model parameters
p_i	Propensity score: $P(A_i = 1 \mid X_i)$
$h^a(t)$	Cause-specific hazard of dementia at time t under treatment a
$h^{*a}(t)$	Overall hazard (dementia + death) at time t under treatment a
$F^a(t)$	Cumulative incidence function at time t under treatment a

Simulate Data

We generate data with:

- Baseline covariates affecting treatment assignment as well as time to events
- Treatment affecting both dementia and death risks
- Competing risks (dementia vs. death)
- Administrative censoring

```
# Parameters
n <- 2000           # Sample size
t_interest <- 20    # Time point of interest for inference

# Step 1: Generate baseline covariates
age <- rnorm(n, mean = 65, sd = 10)
female <- rbinom(n, 1, 0.5)
comorbidity <- rpois(n, lambda = 2)

# Design matrix with intercept (n x q)
X <- cbind(1, age, female, comorbidity)
colnames(X) <- c("intercept", "age", "female", "comorbidity")
q <- ncol(X)  # Number of PS parameters

cat("Design matrix X dimensions:", dim(X), "\n")
```

```
## Design matrix X dimensions: 2000 4
```

```r
cat("Number of PS parameters (q):", q, "\n")
```

```
## Number of PS parameters (q): 4
```

```r
# Step 2: Generate treatment assignment from propensity score model
# True model: logit P(A=1|X) = gamma_0 + gamma_1*age + gamma_2*female + gamma_3*comorbidity

gamma_true <- c(-1, 0.02, -0.3, 0.1)
names(gamma_true) <- colnames(X)

# True propensity scores
linear_pred_true <- X %*% gamma_true
ps_true <- plogis(linear_pred_true)

# Generate treatment assignment
A <- rbinom(n, 1, ps_true)

cat("True gamma:", gamma_true, "\n")
```

```
## True gamma: -1 0.02 -0.3 0.1
```

```r
cat("Treatment prevalence:", mean(A), "\n")
```

```
## Treatment prevalence: 0.5785
```

```r
cat("Propensity score range:", round(range(ps_true), 3), "\n")
```

```
## Propensity score range: 0.393 0.802
```

```r
# Step 3: Generate competing risks outcomes
# Using Weibull distributions; treatment reduces dementia risk
# Generate competing risks outcomes

  # Baseline hazard parameters (Weibull distribution)
  # Dementia: treatment reduces risk
  shape_dementia <- 1.5
  scale_dementia_0 <- 50   # Control group baseline
  scale_dementia_1 <- 60   # Treatment group baseline

  # Death: treatment has smaller effect
  shape_death <- 1.2
  scale_death_0 <- 55
  scale_death_1 <- 58

censor_time <- 40  # Administrative censoring

  # Generate latent event times with covariate effects
  U_dementia <- runif(n)
  U_death <- runif(n)
```

```r
  # Weibull scale parameter adjusted by covariates and treatment
  # Scale = baseline_scale * exp(covariate_effects)
  # Higher scale = lower hazard = longer times

  # Dementia scale: affected by age (increases risk), female (protective), comorbidity (increases risk)
  scale_adj_dementia <- exp(-0.02 * (age - 65) + 0.15 * female - 0.05 * comorbidity)
  scale_dementia <- ifelse(A == 1,
                           scale_dementia_1 * scale_adj_dementia,
                           scale_dementia_0 * scale_adj_dementia)

  # Death scale: different covariate effects
  scale_adj_death <- exp(-0.03 * (age - 65) - 0.10 * female - 0.08 * comorbidity)
  scale_death <- ifelse(A == 1,
                        scale_death_1 * scale_adj_death,
                        scale_death_0 * scale_adj_death)

  # Weibull survival time: T = scale * (-log(U))^(1/shape)
  T_dementia <- scale_dementia * (-log(U_dementia))^(1/shape_dementia)
  T_death <- scale_death * (-log(U_death))^(1/shape_death)


# Determine observed event (minimum of dementia and death times)
T_event <- pmin(T_dementia, T_death)
Delta_latent <- ifelse(T_dementia <= T_death, 1, 2)  # 1 = dementia, 2 = death

# Apply administrative censoring
T_obs <- pmin(T_event, censor_time)
Delta <- ifelse(T_event <= censor_time, Delta_latent, 0)  # 0 = censored

# Discretize time for easier computation
T_obs <- ceiling(T_obs)
T_obs[T_obs == 0] <- 1  # Minimum time is 1

data.frame(age=round(age), female, comorbidity, A, T_obs, Delta)[1:10,]
```

```
##    age female comorbidity A T_obs Delta
## 1   79      1           1 1    20     2
## 2   59      1           2 1    40     0
## 3   69      0           1 1    18     1
## 4   71      1           0 1    19     2
## 5   69      0           1 1    33     1
## 6   64      0           1 0    11     2
## 7   80      0           0 0    14     2
## 8   64      0           3 0    20     1
## 9   85      0           4 1    19     2
## 10  64      1           1 1    36     2
```

```r
max_time <- max(T_obs)

# Summary
cat("\n=== Data Summary ===\n")
```

```
##
```

```
## === Data Summary ===
```

```
cat("Sample size:", n, "\n")
```

```
## Sample size: 2000
```

```
cat("Treated:", sum(A), "(", round(100*mean(A), 1), "%)\n")
```

```
## Treated: 1157 ( 57.9 %)
```

```
cat("Dementia events:", sum(Delta == 1), "\n")
```

```
## Dementia events: 589
```

```
cat("Death events:", sum(Delta == 2), "\n")
```

```
## Death events: 939
```

```
cat("Censored:", sum(Delta == 0), "\n")
```

```
## Censored: 472
```

```
cat("Max observed time:", max_time, "\n")
```

```
## Max observed time: 40
```

Step 1: Estimating Propensity Scores

The Propensity Score Model

We model the probability of treatment using logistic regression:

$$\text{logit } P(A = 1 \mid X) = X\gamma$$

$$p_i = \frac{\exp(X_i\gamma)}{1 + \exp(X_i\gamma)}$$

```
# Fit logistic regression for propensity scores
ps_model <- glm(A ~ age + female + comorbidity, family = binomial)

# Extract estimates
gamma_hat <- coef(ps_model)
ps_hat <- fitted(ps_model)
```

```r
# Compare estimated to true coefficients
comparison <- data.frame(
  Parameter = names(gamma_hat),
  True = gamma_true,
  Estimated = round(gamma_hat, 4)
)
print(comparison)
```

```
##               Parameter  True Estimated
## intercept   (Intercept) -1.00   -0.9296
## age                 age  0.02    0.0175
## female           female -0.30   -0.1132
## comorbidity comorbidity  0.10    0.0895
```

Score Function for Propensity Score Model

The MLE $\hat{\gamma}$ solves the score equation $U_2(\gamma) = 0$, where:

$$U_2(\gamma) = \frac{1}{n}\sum_{i=1}^{n}(A_i - \hat{p}_i)X_i$$

Each observation contributes:

$$U_{2,i}(\gamma) = (A_i - \hat{p}_i)X_i$$

```r
# Score function U2: contribution from each observation
# U2_i = (A_i - p_hat_i) * X_i
# This is an n x q matrix

U2 <- (A - ps_hat) * X

cat("U2 dimensions:", dim(U2), "\n")
```

```
## U2 dimensions: 2000 4
```

```r
cat("(Each row is one observation's contribution to the score)\n\n")
```

```
## (Each row is one observation's contribution to the score)
```

```r
# Verify: column sums should be approximately 0 (score equation solved)
cat("Column sums of U2 (should be ~0):\n")
```

```
## Column sums of U2 (should be ~0):
```

```r
print(round(colSums(U2), 10))
```

```
##   intercept         age      female comorbidity
##       0e+00       4e-10       0e+00       0e+00
```

Derivative of Score Function

We need $D_\gamma^{-1} = E[\partial U_2/\partial\gamma']$.

Derivation:

Starting from $U_{2,i}(\gamma) = (A_i - p_i)X_i$:

$$\frac{\partial U_{2,i}}{\partial\gamma'} = -\frac{\partial p_i}{\partial\gamma'}X_i = -p_i(1-p_i)X_iX_i'$$

Taking the average:

$$D_\gamma^{-1} = -\frac{1}{n}\sum_{i=1}^{n} p_i(1-p_i)X_iX_i' = -\frac{1}{n}X'WX$$

where $W = \text{diag}(p_1(1-p_1), \ldots, p_n(1-p_n))$.

```
# Compute D_gamma_inv = E[dU2/dgamma'] = -(1/n) * X' * W * X
# where W = diag(p_hat * (1 - p_hat))

W_ps <- ps_hat * (1 - ps_hat)  # n x 1 vector of weights

# D_gamma_inv is q x q
D_gamma_inv <- -t(X) %*% (W_ps * X) / n

cat("D_gamma_inv dimensions:", dim(D_gamma_inv), "\n")
```

```
## D_gamma_inv dimensions: 4 4
```

```
cat("\nD_gamma_inv matrix:\n")
```

```
##
## D_gamma_inv matrix:
```

```
print(round(D_gamma_inv, 6))
```

```
##              intercept          age    female comorbidity
## intercept    -0.241069   -15.568960 -0.124935   -0.461900
## age         -15.568960 -1028.945758 -8.078593  -29.720176
## female       -0.124935    -8.078593 -0.124935   -0.238645
## comorbidity  -0.461900   -29.720176 -0.238645   -1.338910
```

```
# We'll need D_gamma (the inverse) later
D_gamma <- solve(D_gamma_inv)

cat("\nD_gamma (inverse) diagonal elements:", round(diag(D_gamma), 2), "\n")
```

```
##
## D_gamma (inverse) diagonal elements: -196.9 -0.04 -16.62 -2.21
```

Step 2: Computing IPW-Weighted Hazards and CIF

Inverse Probability Weights

For treatment group $a \in \{0, 1\}$, the IPW weight is:

$$w_i = \frac{I(A_i = a)}{P(A = a \mid X_i)}$$

- For treated ($a = 1$): $w_i = A_i / \hat{p}_i$
- For control ($a = 0$): $w_i = (1 - A_i)/(1 - \hat{p}_i)$

Key insight: The treatment indicator in the numerator automatically filters to the correct group.

```
# IPW weights for each treatment arm
weights_1 <- A / ps_hat           # Weights for treated
weights_0 <- (1 - A) / (1 - ps_hat)  # Weights for control

# Verify: control patients have zero weight in treated group (and vice versa)
cat("Treated group weights:\n")
```

```
## Treated group weights:
```

```
cat("  - For treated patients (A=1): range =", round(range(weights_1[A == 1]), 3), "\n")
```

```
##   - For treated patients (A=1): range = 1.308 2.334
```

```
cat("  - For control patients (A=0): all =", unique(weights_1[A == 0]), "\n")
```

```
##   - For control patients (A=0): all = 0
```

```
cat("\nControl group weights:\n")
```

```
##
## Control group weights:
```

```
cat("  - For control patients (A=0): range =", round(range(weights_0[A == 0]), 3), "\n")
```

```
##   - For control patients (A=0): range = 1.755 3.94
```

```
cat("  - For treated patients (A=1): all =", unique(weights_0[A == 1]), "\n")
```

```
##   - For treated patients (A=1): all = 0
```

Nonparametric Weighted Hazard Estimation

We estimate the cause-specific hazard at each discrete time t:

$$\hat{h}^a(t) = \frac{\sum_i w_i \cdot I(T_i \geq t) \cdot I(T_i = t, \Delta_i = 1)}{\sum_i w_i \cdot I(T_i \geq t)}$$

This is: (weighted events at time t) / (weighted number at risk at time t).

No proportional hazards assumption is required.

```
# Initialize storage for hazards
# NOTE: We compute h_dementia and h_star (overall), NOT h_death separately
h_dementia_1 <- h_star_1 <- numeric(max_time)
h_dementia_0 <- h_star_0 <- numeric(max_time)

# Compute hazards at each time point
for (t in 1:max_time) {
  # Indicators
  at_risk <- (T_obs >= t)                          # Still at risk at time t
  dementia_t <- (T_obs == t) & (Delta == 1)        # Dementia at time t
  any_event_t <- (T_obs == t) & (Delta %in% c(1,2)) # Any event (dementia OR death) at time t

  # Treated group (a = 1)
  denom_1 <- sum(weights_1 * at_risk)
  if (denom_1 > 0) {
    h_dementia_1[t] <- sum(weights_1 * at_risk * dementia_t) / denom_1
    h_star_1[t] <- sum(weights_1 * at_risk * any_event_t) / denom_1  # Overall hazard directly
  }

  # Control group (a = 0)
  denom_0 <- sum(weights_0 * at_risk)
  if (denom_0 > 0) {
    h_dementia_0[t] <- sum(weights_0 * at_risk * dementia_t) / denom_0
    h_star_0[t] <- sum(weights_0 * at_risk * any_event_t) / denom_0  # Overall hazard directly
  }
}

# Display hazards at selected time points
cat("Dementia hazards at selected times:\n")
```

```
## Dementia hazards at selected times:
```

```
times_show <- c(5, 10, 15, 20, 25)
hazard_table <- data.frame(
  Time = times_show,
  h_dementia_treated = round(h_dementia_1[times_show], 4),
  h_dementia_control = round(h_dementia_0[times_show], 4),
  h_star_treated = round(h_star_1[times_show], 4),
  h_star_control = round(h_star_0[times_show], 4)
)
print(hazard_table)
```

```
##   Time h_dementia_treated h_dementia_control h_star_treated h_star_control
## 1    5             0.0047             0.0109         0.0242         0.0305
## 2   10             0.0092             0.0222         0.0227         0.0427
## 3   15             0.0121             0.0085         0.0309         0.0300
## 4   20             0.0110             0.0158         0.0303         0.0464
## 5   25             0.0140             0.0048         0.0342         0.0280
```

Cumulative Incidence Function

The CIF for dementia is:

$$F^a(t) = \sum_{u=1}^{t} \hat{h}^a(u) \cdot S(u-1)$$

where the overall survival function is:

$$S(u-1) = \exp\left(-\sum_{v=1}^{u-1} h^{*a}(v)\right)$$

```r
# Overall survival: S(t) = exp(-cumulative hazard)
# S[t] represents S(t-1) for indexing convenience
S_1 <- exp(-c(0, cumsum(h_star_1)[-max_time]))
S_0 <- exp(-c(0, cumsum(h_star_0)[-max_time]))

# CIF for dementia: F(t) = sum_{u=1}^t h(u) * S(u-1)
CIF_1 <- cumsum(h_dementia_1 * S_1)
CIF_0 <- cumsum(h_dementia_0 * S_0)

# Risk Ratio
RR <- CIF_1 / CIF_0

# Display results
cat("=== CIF Results at time", t_interest, "===\n")
```

```
## === CIF Results at time 20 ===
```

```r
cat("CIF (Treated):", round(CIF_1[t_interest], 4), "\n")
```

```
## CIF (Treated): 0.1508
```

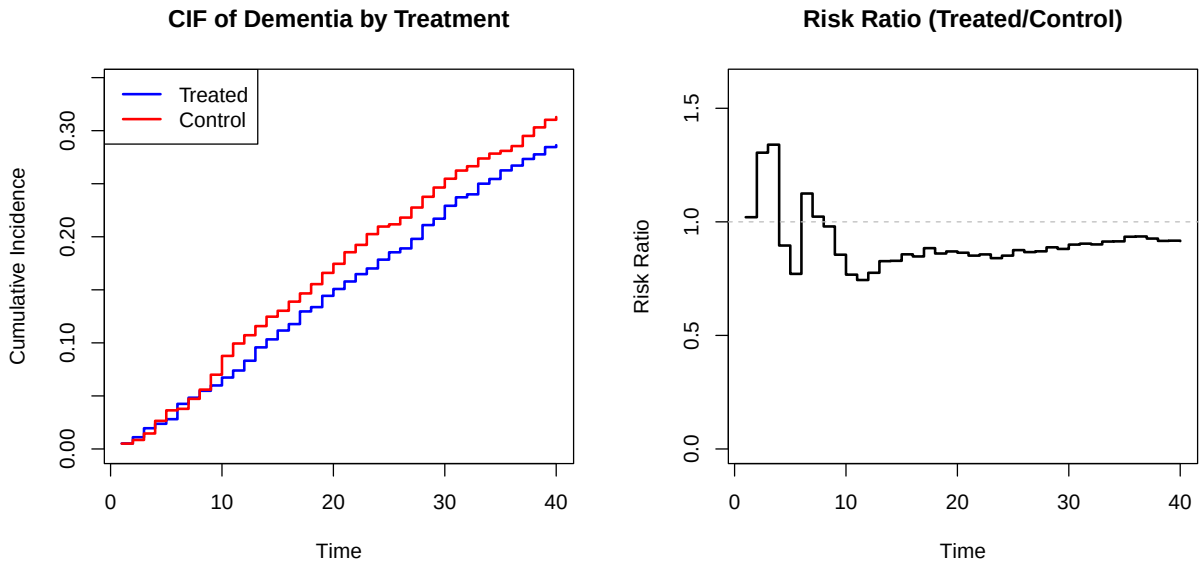
```r
cat("CIF (Control):", round(CIF_0[t_interest], 4), "\n")
```

```
## CIF (Control): 0.1745
```

```r
cat("Risk Ratio:   ", round(RR[t_interest], 4), "\n")
```

```
## Risk Ratio:    0.864
```

```r
# Plot CIF curves (without confidence bands for now)
par(mfrow = c(1, 2))

# CIF plot
plot(1:max_time, CIF_1, type = "s", col = "blue", lwd = 2,
     ylim = c(0, max(CIF_0, CIF_1) * 1.1),
     xlab = "Time", ylab = "Cumulative Incidence",
     main = "CIF of Dementia by Treatment")
lines(1:max_time, CIF_0, type = "s", col = "red", lwd = 2)
legend("topleft", c("Treated", "Control"), col = c("blue", "red"), lwd = 2)

# Risk Ratio plot
plot(1:max_time, RR, type = "s", col = "black", lwd = 2,
     ylim = c(0, max(RR[is.finite(RR)], na.rm = TRUE) * 1.2),
     xlab = "Time", ylab = "Risk Ratio",
     main = "Risk Ratio (Treated/Control)")
abline(h = 1, lty = 2, col = "gray")
```




```r
par(mfrow = c(1, 1))
```

Background: Influence Functions

Key Concept

An estimator $\hat{\theta}$ is **asymptotically linear** if:

$$\sqrt{n}(\hat{\theta} - \theta) = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \psi_i + o_p(1)$$

where ψ_i is the **influence function** for observation i.

Interpretation: The influence function measures how much each observation "pushes" the estimator away from the true value.

Variance from Influence Functions

Since $\sqrt{n}(\hat{\theta} - \theta)$ is approximately a sum of i.i.d. terms, by CLT:

$$\text{Var}(\hat{\theta}) = \frac{1}{n} E[\psi_i^2]$$

We estimate this as:

$$\widehat{\text{Var}}(\hat{\theta}) = \frac{1}{n^2} \sum_{i=1}^{n} \hat{\psi}_i^2 = \frac{1}{n} \cdot \text{mean}(\hat{\psi}_i^2)$$

Influence Function for M-Estimators

For estimators solving $U(\theta) = 0$, the influence function is:

$$\psi_i = -\left[E\left(\frac{\partial U}{\partial \theta'} \right) \right]^{-1} U_i(\theta)$$

This comes from a Taylor expansion of $U(\hat{\theta})$ around the true value.

Influence Function for Two-Step Estimation

The Challenge

Our problem involves two estimating equations:

- $U_2(\gamma) = 0$ for propensity score (nuisance parameter)
- $U_1(\beta, \gamma) = 0$ for hazard (target parameter), which depends on γ

Uncertainty in $\hat{\gamma}$ must propagate to uncertainty in $\hat{\beta}$.

The Solution: Two-Part Influence Function

The influence function for $\hat{\beta}$ is:

$$\psi_i^* = A_i - B_i$$

where:

- **Part A (direct influence):** $A_i = D_\beta^{-1} \cdot U_{1,i}$
- **Part B (PS correction):** $B_i = D_\beta^{-1} \cdot G \cdot D_\gamma \cdot U_{2,i}$

Part A is what we'd get if propensity scores were known. Part B corrects for the fact that they're estimated.

Computing the Components

```
cat("Components needed for influence function:\n\n")
```

```
## Components needed for influence function:
```

```
cat("For Part A:\n")
```

```
## For Part A:
```

```
cat("  - U1_i: score contribution for hazard equation\n")
```

```
##   - U1_i: score contribution for hazard equation
```

```
cat("  - D_beta: derivative of hazard equation w.r.t. h(t)\n\n")
```

```
##   - D_beta: derivative of hazard equation w.r.t. h(t)
```

```
cat("For Part B:\n")
```

```
## For Part B:
```

```
cat("  - G: how U1 depends on gamma (through weights)\n")
```

```
##   - G: how U1 depends on gamma (through weights)
```

```
cat("  - D_gamma: derivative of PS equation (already computed)\n")
```

```
##   - D_gamma: derivative of PS equation (already computed)
```

```
cat("  - U2_i: score contribution for PS equation (already computed)\n")
```

```
##   - U2_i: score contribution for PS equation (already computed)
```

Step-by-Step: Influence Function for Hazards

Estimating Equations

For dementia hazard $h(t)$:

$$U_{1,i}^{dem} = w_i \cdot I(T_i \geq t) \cdot [I(T_i = t, \Delta_i = 1) - h(t)]$$

For overall hazard $h^*(t)$:

$$U_{1,i}^{*} = w_i \cdot I(T_i \geq t) \cdot [I(T_i = t, \Delta_i \in \{1, 2\}) - h^*(t)]$$

Implementation

```r
# Storage for influence functions: n x max_time matrices
psi_h_dementia_1 <- psi_h_star_1 <- matrix(0, n, max_time)
psi_h_dementia_0 <- psi_h_star_0 <- matrix(0, n, max_time)

# Loop over time points
for (t in 1:max_time) {
  # Indicators
  at_risk <- as.numeric(T_obs >= t)
  dementia_t <- as.numeric((T_obs == t) & (Delta == 1))
  any_event_t <- as.numeric((T_obs == t) & (Delta %in% c(1, 2)))  # Dementia OR death

  #============================================================================
  # TREATED GROUP (a = 1)
  #============================================================================

  #--- Part A: Direct influence ---
  # For dementia hazard
  U1_dementia_1 <- weights_1 * at_risk * (dementia_t - h_dementia_1[t])
  # For overall hazard (DIRECT computation)
  U1_star_1 <- weights_1 * at_risk * (any_event_t - h_star_1[t])

  # D_beta = -mean(w * at_risk) [same for both hazards]
  D_beta_1 <- -mean(weights_1 * at_risk)

  if (abs(D_beta_1) > 1e-10) {
    A_dementia_1 <- -U1_dementia_1 / D_beta_1
    A_star_1 <- -U1_star_1 / D_beta_1
  } else {
    A_dementia_1 <- A_star_1 <- rep(0, n)
  }

  #--- Part B: Correction for PS uncertainty ---
  # d(log w)/d(X*gamma) = -(1-p) for treated
  dlogw_dlp_1 <- -(1 - ps_hat)

  # G for dementia hazard
  G_dementia_1 <- colMeans(U1_dementia_1 * dlogw_dlp_1 * X)
  # G for overall hazard
  G_star_1 <- colMeans(U1_star_1 * dlogw_dlp_1 * X)

  # B_i = D_beta^{-1} * G * D_gamma * U2_i
  if (abs(D_beta_1) > 1e-10) {
    GD_dementia_1 <- G_dementia_1 %*% D_gamma  # 1 x q
    GD_star_1 <- G_star_1 %*% D_gamma
    B_dementia_1 <- as.vector((1 / (-D_beta_1)) * (U2 %*% t(GD_dementia_1)))
    B_star_1 <- as.vector((1 / (-D_beta_1)) * (U2 %*% t(GD_star_1)))
  } else {
    B_dementia_1 <- B_star_1 <- rep(0, n)
  }

  # psi*_i = A_i - B_i
  psi_h_dementia_1[, t] <- A_dementia_1 - B_dementia_1
  psi_h_star_1[, t] <- A_star_1 - B_star_1
```

```r
  #============================================================================
  # CONTROL GROUP (a = 0)
  #============================================================================

  #--- Part A ---
  U1_dementia_0 <- weights_0 * at_risk * (dementia_t - h_dementia_0[t])
  U1_star_0 <- weights_0 * at_risk * (any_event_t - h_star_0[t])
  D_beta_0 <- -mean(weights_0 * at_risk)

  if (abs(D_beta_0) > 1e-10) {
    A_dementia_0 <- -U1_dementia_0 / D_beta_0
    A_star_0 <- -U1_star_0 / D_beta_0
  } else {
    A_dementia_0 <- A_star_0 <- rep(0, n)
  }

  #--- Part B ---
  # d(log w)/d(X*gamma) = p for control
  dlogw_dlp_0 <- ps_hat

  G_dementia_0 <- colMeans(U1_dementia_0 * dlogw_dlp_0 * X)
  G_star_0 <- colMeans(U1_star_0 * dlogw_dlp_0 * X)

  if (abs(D_beta_0) > 1e-10) {
    GD_dementia_0 <- G_dementia_0 %*% D_gamma
    GD_star_0 <- G_star_0 %*% D_gamma
    B_dementia_0 <- as.vector((1 / (-D_beta_0)) * (U2 %*% t(GD_dementia_0)))
    B_star_0 <- as.vector((1 / (-D_beta_0)) * (U2 %*% t(GD_star_0)))
  } else {
    B_dementia_0 <- B_star_0 <- rep(0, n)
  }

  psi_h_dementia_0[, t] <- A_dementia_0 - B_dementia_0
  psi_h_star_0[, t] <- A_star_0 - B_star_0
}

cat("Influence functions computed for hazards.\n")
```

```
## Influence functions computed for hazards.
```

```r
cat("Dimensions of psi_h_dementia_1:", dim(psi_h_dementia_1), "\n")
```

```
## Dimensions of psi_h_dementia_1: 2000 40
```

```r
cat("Dimensions of psi_h_star_1:", dim(psi_h_star_1), "\n")
```

```
## Dimensions of psi_h_star_1: 2000 40
```

```r
cat("(Each row is one observation, each column is one time point)\n")
```

```
## (Each row is one observation, each column is one time point)
```

Delta Method: From Hazard to CIF (IMPROVED)

The CIF as a Function of Hazards

$$F(t) = \sum_{u=1}^{t} h(u) \cdot S(u-1) = \sum_{u=1}^{t} h(u) \cdot \exp\left(-\sum_{v=1}^{u-1} h^*(v)\right)$$

Delta Method

Mathematical rule: If $\hat{\theta}$ has influence function ψ_θ, then $g(\hat{\theta})$ has influence function:

$$\psi_g = \frac{\partial g}{\partial \theta} \cdot \psi_\theta$$

Partial Derivatives of CIF

1. **Direct effect of dementia hazard:**

$$\frac{\partial F(t)}{\partial h(u)} = S(u-1) \quad \text{for } u \leq t$$

2. **Indirect effect through survival (via overall hazard):**

$$\frac{\partial F(t)}{\partial h^*(v)} = -\sum_{u=v+1}^{t} h(u) \cdot S(u-1) \quad \text{for } v < t$$

```r
# Influence function for CIF at each time point
psi_CIF_1 <- psi_CIF_0 <- matrix(0, n, max_time)

for (t in 1:max_time) {
  psi_t_1 <- psi_t_0 <- rep(0, n)

  for (u in 1:t) {
    #--- Direct effect: dF/dh(u) = S(u-1) ---
    psi_t_1 <- psi_t_1 + S_1[u] * psi_h_dementia_1[, u]
    psi_t_0 <- psi_t_0 + S_0[u] * psi_h_dementia_0[, u]

    #--- Indirect effect through survival (for u < t) ---
    if (u < t) {
      # dF(t)/dh_star(u) = -sum_{v=u+1}^t h(v) * S(v-1)
      deriv_S_1 <- deriv_S_0 <- 0
      for (v in (u + 1):t) {
        deriv_S_1 <- deriv_S_1 - h_dementia_1[v] * S_1[v]
        deriv_S_0 <- deriv_S_0 - h_dementia_0[v] * S_0[v]
      }
      # SIMPLER: Just multiply by psi_h_star directly
      psi_t_1 <- psi_t_1 + deriv_S_1 * psi_h_star_1[, u]
      psi_t_0 <- psi_t_0 + deriv_S_0 * psi_h_star_0[, u]
    }
  }
```

```
  psi_CIF_1[, t] <- psi_t_1
  psi_CIF_0[, t] <- psi_t_0
}

cat("Influence functions computed for CIF.\n")
```

```
## Influence functions computed for CIF.
```

```
cat("Dimensions of psi_CIF_1:", dim(psi_CIF_1), "\n")
```

```
## Dimensions of psi_CIF_1: 2000 40
```

Variance of CIF and Risk Ratio

Variance of CIF

$$\text{Var}(\hat{F}(t)) = \frac{1}{n} \cdot \text{mean}(\psi_{F,i}^2)$$

```
# Extract influence functions at time of interest
psi_1_t <- psi_CIF_1[, t_interest]
psi_0_t <- psi_CIF_0[, t_interest]

# Variance of CIF
var_CIF_1 <- mean(psi_1_t^2) / n
var_CIF_0 <- mean(psi_0_t^2) / n

se_CIF_1 <- sqrt(var_CIF_1)
se_CIF_0 <- sqrt(var_CIF_0)

cat("=== Variance of CIF at time", t_interest, "===\n")
```

```
## === Variance of CIF at time 20 ===
```

```
cat("CIF (Treated):", round(CIF_1[t_interest], 4),
    " (SE:", round(se_CIF_1, 4), ")\n")
```

```
## CIF (Treated): 0.1508  (SE: 0.0105 )
```

```
cat("CIF (Control):", round(CIF_0[t_interest], 4),
    " (SE:", round(se_CIF_0, 4), ")\n")
```

```
## CIF (Control): 0.1745  (SE: 0.0134 )
```

Variance of Risk Ratio

Delta method for ratio: For $RR = F_1/F_0$:

$$\psi_{RR} = \frac{\partial RR}{\partial F_1}\psi_{F_1} + \frac{\partial RR}{\partial F_0}\psi_{F_0} = \frac{1}{F_0}\psi_{F_1} - \frac{RR}{F_0}\psi_{F_0}$$

For $\log(RR)$ (often preferred for confidence intervals):

$$\psi_{\log RR} = \frac{1}{F_1}\psi_{F_1} - \frac{1}{F_0}\psi_{F_0}$$

```
F1_t <- CIF_1[t_interest]
F0_t <- CIF_0[t_interest]
RR_t <- F1_t / F0_t

# Influence function for RR
psi_RR <- (1 / F0_t) * psi_1_t - (RR_t / F0_t) * psi_0_t
var_RR <- mean(psi_RR^2) / n
se_RR <- sqrt(var_RR)

# Influence function for log(RR)
psi_logRR <- (1 / F1_t) * psi_1_t - (1 / F0_t) * psi_0_t
var_logRR <- mean(psi_logRR^2) / n
se_logRR <- sqrt(var_logRR)

# 95% CI for RR (using log transform for better coverage)
CI_lower <- RR_t * exp(-1.96 * se_logRR)
CI_upper <- RR_t * exp(1.96 * se_logRR)

cat("\n=== Risk Ratio at time", t_interest, "===\n")
```

```
##
## === Risk Ratio at time 20 ===
```

```
cat("RR:         ", round(RR_t, 4), " (SE:", round(se_RR, 4), ")\n")
```

```
## RR:          0.864  (SE: 0.089 )
```

```
cat("log(RR):    ", round(log(RR_t), 4), " (SE:", round(se_logRR, 4), ")\n")
```

```
## log(RR):    -0.1462  (SE: 0.103 )
```

```
cat("95% CI:     (", round(CI_lower, 4), ",", round(CI_upper, 4), ")\n")
```

```
## 95% CI:     ( 0.7061 , 1.0573 )
```

Summary of Results

```r
# Create summary table
results_summary <- data.frame(
  Quantity = c("CIF (Treated)", "CIF (Control)", "Risk Ratio", "log(RR)"),
  Estimate = c(F1_t, F0_t, RR_t, log(RR_t)),
  SE = c(se_CIF_1, se_CIF_0, se_RR, se_logRR),
  CI_Lower = c(F1_t - 1.96*se_CIF_1, F0_t - 1.96*se_CIF_0, CI_lower, log(RR_t) - 1.96*se_logRR),
  CI_Upper = c(F1_t + 1.96*se_CIF_1, F0_t + 1.96*se_CIF_0, CI_upper, log(RR_t) + 1.96*se_logRR)
)
results_summary[, 2:5] <- round(results_summary[, 2:5], 4)

cat("=== Final Results at time", t_interest, "===\n\n")
```

```
## === Final Results at time 20 ===
```

```r
print(results_summary, row.names = FALSE)
```

```
##       Quantity Estimate     SE CI_Lower CI_Upper
##  CIF (Treated)   0.1508 0.0105   0.1303   0.1713
##  CIF (Control)   0.1745 0.0134   0.1483   0.2007
##     Risk Ratio   0.8640 0.0890   0.7061   1.0573
##         log(RR)  -0.1462 0.1030  -0.3480   0.0557
```

Validation: Bootstrap Comparison

We compare our influence function standard errors to bootstrap standard errors.

```r
# Bootstrap validation
B <- 200
RR_boot <- CIF_1_boot <- CIF_0_boot <- numeric(B)

cat("Running", B, "bootstrap replications...\n")
```

```
## Running 200 bootstrap replications...
```

```r
for (b in 1:B) {
  if (b %% 50 == 0) cat("  Bootstrap", b, "/", B, "\n")

  # Resample with replacement
  idx <- sample(1:n, n, replace = TRUE)

  A_b <- A[idx]
  X_b <- X[idx, ]
  T_obs_b <- T_obs[idx]
  Delta_b <- Delta[idx]

  # Re-fit PS model
  ps_fit_b <- glm(A_b ~ X_b[, -1], family = binomial)
```

```r
  ps_hat_b <- fitted(ps_fit_b)

  # Weights
  w1_b <- A_b / ps_hat_b
  w0_b <- (1 - A_b) / (1 - ps_hat_b)

  # Compute hazards
  h_dem_1_b <- h_dem_0_b <- numeric(max_time)
  h_star_1_b <- h_star_0_b <- numeric(max_time)

  for (tt in 1:max_time) {
    at_risk_b <- (T_obs_b >= tt)
    dem_b <- (T_obs_b == tt) & (Delta_b == 1)
    any_event_b <- (T_obs_b == tt) & (Delta_b %in% c(1, 2))

    d1 <- sum(w1_b * at_risk_b)
    d0 <- sum(w0_b * at_risk_b)

    if (d1 > 0) {
      h_dem_1_b[tt] <- sum(w1_b * at_risk_b * dem_b) / d1
      h_star_1_b[tt] <- sum(w1_b * at_risk_b * any_event_b) / d1
    }
    if (d0 > 0) {
      h_dem_0_b[tt] <- sum(w0_b * at_risk_b * dem_b) / d0
      h_star_0_b[tt] <- sum(w0_b * at_risk_b * any_event_b) / d0
    }
  }

  # Compute CIF
  S_1_b <- exp(-c(0, cumsum(h_star_1_b)[-max_time]))
  S_0_b <- exp(-c(0, cumsum(h_star_0_b)[-max_time]))

  CIF_1_boot[b] <- sum(h_dem_1_b[1:t_interest] * S_1_b[1:t_interest])
  CIF_0_boot[b] <- sum(h_dem_0_b[1:t_interest] * S_0_b[1:t_interest])
  RR_boot[b] <- CIF_1_boot[b] / CIF_0_boot[b]
}
```

```
##   Bootstrap 50 / 200
##   Bootstrap 100 / 200
##   Bootstrap 150 / 200
##   Bootstrap 200 / 200
```

```r
# Compare standard errors
comparison_se <- data.frame(
  Quantity = c("CIF (Treated)", "CIF (Control)", "Risk Ratio", "log(RR)"),
  SE_InfluenceFunction = round(c(se_CIF_1, se_CIF_0, se_RR, se_logRR), 4),
  SE_Bootstrap = round(c(sd(CIF_1_boot), sd(CIF_0_boot), sd(RR_boot), sd(log(RR_boot))), 4)
)

cat("\n=== Comparison of Standard Errors ===\n\n")
```

```
##
## === Comparison of Standard Errors ===
```

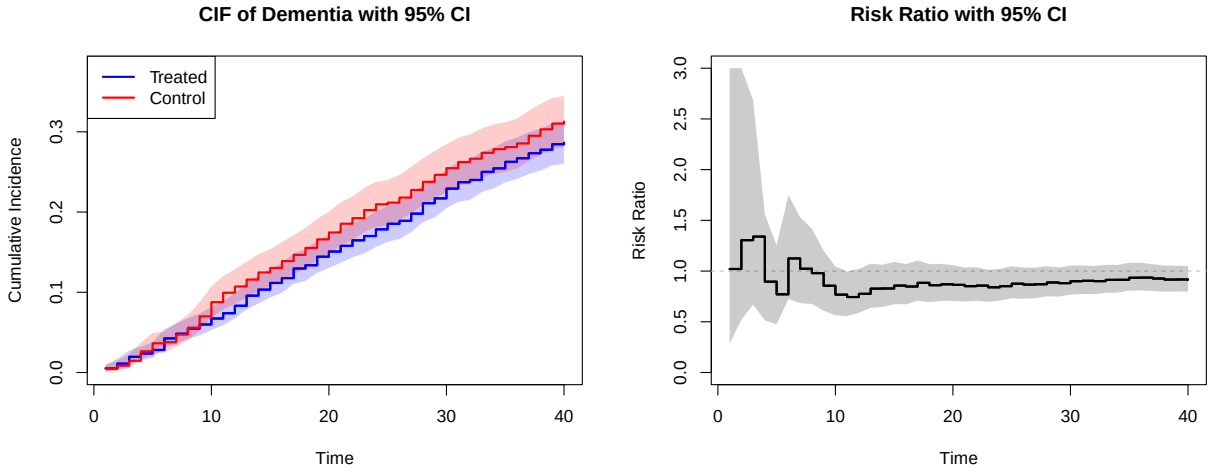
```r
print(comparison_se, row.names = FALSE)
```

```
##       Quantity SE_InfluenceFunction SE_Bootstrap
##  CIF (Treated)                0.0105       0.0098
##  CIF (Control)                0.0134       0.0138
##     Risk Ratio                0.0890       0.0887
##        log(RR)                0.1030       0.1017
```

Final Plot with Confidence Bands

```r
# Compute SE at each time point
se_CIF_1_all <- sqrt(colMeans(psi_CIF_1^2) / n)
se_CIF_0_all <- sqrt(colMeans(psi_CIF_0^2) / n)

# Plot
par(mfrow = c(1, 2))

#--- CIF with confidence bands ---
plot(1:max_time, CIF_1, type = "s", col = "blue", lwd = 2,
     ylim = c(0, max(CIF_0 + 1.96*se_CIF_0_all) * 1.1),
     xlab = "Time", ylab = "Cumulative Incidence",
     main = "CIF of Dementia with 95% CI")

# Confidence bands
polygon(c(1:max_time, max_time:1),
        c(CIF_1 + 1.96*se_CIF_1_all, rev(pmax(0, CIF_1 - 1.96*se_CIF_1_all))),
        col = rgb(0, 0, 1, 0.2), border = NA)
polygon(c(1:max_time, max_time:1),
        c(CIF_0 + 1.96*se_CIF_0_all, rev(pmax(0, CIF_0 - 1.96*se_CIF_0_all))),
        col = rgb(1, 0, 0, 0.2), border = NA)

lines(1:max_time, CIF_1, type = "s", col = "blue", lwd = 2)
lines(1:max_time, CIF_0, type = "s", col = "red", lwd = 2)
legend("topleft", c("Treated", "Control"), col = c("blue", "red"), lwd = 2)

#--- Risk Ratio with confidence bands ---
RR_all <- CIF_1 / CIF_0
se_logRR_all <- sqrt(colMeans(psi_CIF_1^2) / (n * CIF_1^2) +
                     colMeans(psi_CIF_0^2) / (n * CIF_0^2))
CI_lower_all <- RR_all * exp(-1.96 * se_logRR_all)
CI_upper_all <- RR_all * exp(1.96 * se_logRR_all)

plot(1:max_time, RR_all, type = "s", col = "black", lwd = 2,
     ylim = c(0, min(max(CI_upper_all[is.finite(CI_upper_all)], na.rm = TRUE), 3)),
     xlab = "Time", ylab = "Risk Ratio",
     main = "Risk Ratio with 95% CI")
abline(h = 1, lty = 2, col = "gray")
```

```r
# Confidence band
polygon(c(1:max_time, max_time:1),
        c(pmin(CI_upper_all, 3), rev(pmax(0, CI_lower_all))),
        col = rgb(0, 0, 0, 0.2), border = NA)
lines(1:max_time, RR_all, type = "s", col = "black", lwd = 2)
```




```r
par(mfrow = c(1, 1))
```

Conclusion

This tutorial demonstrated:

1. **Two-step IPW estimation:** Propensity scores → Weighted hazards → CIF
2. **Influence function derivation:** Accounts for uncertainty in both steps
3. **Variance estimation:** Uses $\text{Var}(\hat{\theta}) = \frac{1}{n}\text{mean}(\psi_i^2)$
4. **Delta method:** Transforms hazard variance to CIF and RR variance
5. **Validation:** Bootstrap confirms accuracy of influence function approach
6. **Improvement:** Computing ψ_{h^*} directly is simpler and more efficient

Key advantage: The influence function approach is computationally efficient for large datasets, avoiding the need for time-consuming bootstrap.

Funding

This work was supported by Utah Clinical & Translational Science Institute (CTSI) Translational Innovation Pilot (TIP) Program Award (NCATS UM1TR004409).