Algorithm of HOIF estimators for ATE

Contents

1 Input Data **1**

2 Target Formula **2**

3 The Integrated Algorithm (Main Interface) **4**

4 Step-by-Step Technical Notes **5**

4.1 Transformation of Covariates 5

4.2 Compute the Residuals 6

4.3 Compute the Inverse of the Weighted Gram Matrix . . . 6

4.4 Compute the Projection Matrix 7

4.5 Compute the HOIF Estimators 7

1 Input Data

Denote the observed data by $(X_i, A_i, Y_i)_{i=1}^n$, where $A_i \in \{0, 1\}$ is the binary treatment indicator, $Y_i \in \mathbb{R}$ is the observed outcome, and $X_i \in \mathbb{R}^p$ represents the vector of covariates.

We assume the availability of pre-computed nuisance function estimators: the conditional mean outcomes $(\hat{\mu}(1, X_i), \hat{\mu}(0, X_i))$ and the propensity score $\hat{\pi}(X_i)$. All these estimators map to the real line $\mathbb{R}$. In standard applications, $\hat{\mu}(\cdot)$ and $\hat{\pi}(\cdot)$ should be estimated using an independent sample to avoid overfitting bias.

2 Target Formula

The core function `hoif_ate()` in this R package computes the estimators described below.

Let $\hat{\psi}_a^{\text{AIPW}}$ denote the standard first-order doubly robust estimator (also known as double machine learning or AIPW estimator) of $\psi_a = \mathbb{E}[Y(a)]$, and let $\widehat{\text{ATE}}^{\text{AIPW}} = \hat{\psi}_1^{\text{AIPW}} - \hat{\psi}_0^{\text{AIPW}}$. Conditional on the estimated nuisance functions, this estimator is biased.

The quantity $\mathbb{BC}_m^{\text{ATE}}$ is the m-th order higher order influence function (HOIF) estimator of the *estimable bias* of the AIPW estimator of the average treatment effect (ATE) in causal inference. This methodology was developed through a series of works by James M. Robins and his collaborators [4, 3, 2, 1]. It is *not* itself an estimator of the ATE; the sign convention is such that adding it to the AIPW estimate removes the estimable bias:

$$\widehat{\text{ATE}}_m = \widehat{\text{ATE}}^{\text{AIPW}} + \mathbb{BC}_m^{\text{ATE}}.$$

(In the output of `hoif_ate()`, the components `HOIF1`, `HOIF0` and `ATE` correspond to $\mathbb{HOIF}_m^1$, $\mathbb{HOIF}_m^0$ and $\mathbb{BC}_m^{\text{ATE}}$; the component name `ATE` is kept for backward compatibility — it holds the ATE *bias correction*, not the ATE itself.)

The estimators are defined as follows:

$$\mathbb{BC}_m^{\text{ATE}} = \mathbb{HOIF}_m^1(\hat{\Omega}^1) - \mathbb{HOIF}_m^0(\hat{\Omega}^0),$$
$$\mathbb{HOIF}_m^a(\hat{\Omega}^a) = \sum_{j=2}^{m} \mathbb{IF}_j^a(\hat{\Omega}^a) = \sum_{i=2}^{m}\sum_{j=2}^{i} \binom{i-2}{i-j} \mathbb{U}_j^a(\hat{\Omega}^a),$$

where

$$\mathbb{IF}^a_m(\hat{\Omega}^a) = (-1)^m \frac{(n-m)!}{n!} \sum_{\substack{(i_1,\ldots,i_m)\in J_1^{\times m}:\\ i_1\neq i_2\neq\cdots\neq i_m}} r^a_{i_1} Z_{i_1}^\top \hat{\Omega}^a \prod_{s=2}^{m-1} \{(Q^a_{i_s} - (\hat{\Omega}^a)^{-1})\hat{\Omega}^a\} s^a_{i_m} Z_{i_m} R^a_{i_m},$$

$$\begin{aligned}
\mathbb{U}^a_m(\hat{\Omega}^a) &= (-1)^m \frac{(n-m)!}{n!} \sum_{\substack{(i_1,\ldots,i_m)\in J_1^{\times m}:\\ i_1\neq i_2\neq\cdots\neq i_m}} r^a_{i_1} Z_{i_1}^\top \hat{\Omega}^a \prod_{s=2}^{m-1} \{Q^a_{i_s}\hat{\Omega}^a\} s^a_{i_m} Z_{i_m} R^a_{i_m} \\
&= (-1)^m \frac{(n-m)!}{n!} \sum_{\substack{(i_1,\ldots,i_m)\in J_1^{\times m}:\\ i_1\neq i_2\neq\cdots\neq i_m}} r^a_{i_1} \prod_{s=1}^{m-1} \{Z_{i_s}^\top \hat{\Omega}^a s^a_{i_{s+1}} Z_{i_{s+1}}\} R^a_{i_m}.
\end{aligned}$$

The notation above involves the following components:

$$\begin{aligned}
J_1, J_2 &\subseteq \{1, 2, \ldots, n\}, \\
a &\in \{0, 1\}, \\
s^a_i &= A^a_i (1 - A_i)^{1-a}, \\
r^a_i &= 1 - s^a_i / \left((\hat{\pi}(X_i))^a (1 - \hat{\pi}(X_i))^{1-a} \right), \\
R^a_i &= Y_i - \hat{\mu}(a, X_i), \\
Z_i &= \text{transform}(X_i) \in \mathbb{R}^k, \\
Q^a_i &= s^a_i Z_i Z_i^\top, \\
\hat{\Omega}^a &= \left(\frac{1}{|J_2|} \sum_{i\in J_2} s^a_i Z_i Z_i^\top \right)^{-1}.
\end{aligned}$$

Here, we designate the sample in J_1 as the *estimation sample* and the sample in J_2 as the *training sample* for $\hat{\Omega}^a$. The function transform(X_i) denotes a transformation of the covariates, with the transformed variable Z_i taking values in $\mathbb{R}^k$.

Sample Splitting (Cross-Fitting): When employing K-fold sample splitting $(I_1, I_2, \ldots, I_K)$, for each fold $j \in \{1, \ldots, K\}$, the sample in I_j serves as the estimation sample (i.e., $I_j = J_1$), while samples not in I_j form the training sample (i.e., $i \in J_2 \Leftrightarrow i \notin I_j$). The training sample is used to compute $\hat{\Omega}^a$, which is then applied along with the estimation sample to calculate $\mathbb{HOIF}_m^a$. The final estimator is obtained by averaging across all folds, yielding the empirical HOIF (eHOIF) estimator [2].

No Sample Splitting: When sample splitting is not used, we simply set $J_1 = J_2$, using the entire dataset for both training and estimation, which corresponds to the stable HOIF (sHOIF) estimator [1].

3 The Integrated Algorithm (Main Interface)

The whole function combines all the following steps to get the HOIF bias-correction terms for the ATE. It serves as the primary entry point, orchestrating the data transformation, residual calculation, and the choice of cross-fitting strategy.

Function Inputs:

- Full observed data $(X_i, A_i, Y_i)_{i=1}^n$.
- Nuisance function estimators $(\hat{\mu}(1, X_i), \hat{\mu}(0, X_i), \hat{\pi}(X_i))$.
- Transformation method and its tuning parameters.
- Inverse method of weighted Gram matrix.
- Maximum HOIF order m and ustat backend.
- **Switch**: A boolean flag for sample splitting and the number of splits K.

Procedure:

1. **Global Pre-processing**:
    - Transform the covariates X_i on the whole data to obtain basis functions $(Z_i)_{i=1}^n$.
    - Compute the global residuals $((R_i^1, r_i^1), (R_i^0, r_i^0))_{i=1}^n$.
2. **Branching Logic**:

- *If not using sample splitting*:

  Proceed with the entire dataset using the steps described in the following sections. Output $(\mathrm{BC}_l^{\mathrm{ATE}}, \mathrm{HOIF}_l^a, \mathrm{IIFF}_l^a)$ for $l = 2, \ldots, m$ and $a \in \{0, 1\}$ (returned by the package as `ATE`, `HOIF1`/`HOIF0`, `IIFF1`/`IIFF0`).

- *If using sample splitting (Cross-fitting)*:

  (a) Split the indices $\{1, \ldots, n\}$ into K disjoint parts $(I_1, I_2, \ldots, I_K)$.

  (b) **For each fold** $j = 1, \ldots, K$:

  - **Training**: Use data with indices not in I_j $(i \notin I_j)$ to compute the inverse Gram matrices $(\Omega_{1,j}, \Omega_{0,j})$.
  - **Estimation**: Use data with indices in I_j $(i \in I_j)$ and the pre-computed $(\Omega_{1,j}, \Omega_{0,j})$ to compute:
    * Local projection matrices $(B_{1,j}, B_{0,j})$.
    * Local HOIF bias-correction terms $(\mathrm{BC}_{l,j}^{\mathrm{ATE}}, \mathrm{HOIF}_{l,j}^a, \mathrm{IIFF}_{l,j}^a)$ for $l = 2, \ldots, m$.

  (c) **Aggregation**: Average the results across all K folds:

$$\mathrm{BC}_l^{\mathrm{ATE}} = \frac{1}{K}\sum_{j=1}^{K} \mathrm{BC}_{l,j}^{\mathrm{ATE}}, \quad \mathrm{HOIF}_l^a = \frac{1}{K}\sum_{j=1}^{K} \mathrm{HOIF}_{l,j}^a, \quad \mathrm{IIFF}_l^a = \frac{1}{K}\sum_{j=1}^{K} \mathrm{IIFF}_{l,j}^a \tag{1}$$

Output: Final averaged bias-correction terms $(\mathrm{BC}_l^{\mathrm{ATE}}, \mathrm{HOIF}_l^a, \mathrm{IIFF}_l^a)$ for $l = 2, \ldots, m$.

4 Step-by-Step Technical Notes

4.1 Transformation of Covariates

First, we transform the covariates X_i into a set of basis functions $Z_i \in \mathbb{R}^k$. Common choices include B-splines or Fourier basis functions.

Function Requirement:

- *Input*: Covariate matrix X, the transformation method (e.g., Fourier or B-splines), and relevant tuning parameters (including the basis dimension k).

- *Output*: The transformed basis matrix $Z \in \mathbb{R}^{n\times k}$, where each row Z_i corresponds to the i-th observation.

4.2 Compute the Residuals

Next, we compute two pairs of residuals, indexed by the treatment assignment $a \in \{0, 1\}$.

For the treatment group ($a = 1$):

$$\begin{aligned} R_i^1 &= Y_i - \hat{\mu}(1, X_i) \\ r_i^1 &= 1 - A_i/\hat{\pi}(X_i) \end{aligned}$$

For the control group ($a = 0$):

$$\begin{aligned} R_i^0 &= Y_i - \hat{\mu}(0, X_i) \\ r_i^0 &= 1 - (1 - A_i)/(1 - \hat{\pi}(X_i)) \end{aligned}$$

(The treatment indicator s_i^a is *not* folded into R_i^a here; it enters through the projection matrix B^a below, matching the implementation.)

Function Requirement:

- *Input*: Observed data (A, Y) and estimated nuisance functions $(\hat{\mu}(1, X), \hat{\mu}(0, X), \hat{\pi}(X))$.
- *Output*: Two residual pairs $((R_i^1, r_i^1))_{i=1}^n$ and $((R_i^0, r_i^0))_{i=1}^n$.

4.3 Compute the Inverse of the Weighted Gram Matrix

Compute the inverse of the weighted Gram matrix G_a for each $a \in \{0, 1\}$:

$$G_a = \frac{1}{n} \sum_{i=1}^{n} s_i^a Z_i Z_i^T, \tag{2}$$

where $s_i^a = A_i^a(1-A_i)^{1-a}$ serves as the indicator for the a-th group. Let $\Omega_a = G_a^{-1}$ denote the corresponding inverse matrix.

Several estimation methods can be employed for Ω_a, such as direct inversion (e.g., using `chol2inv()` in R for efficiency) or shrinkage estimators (e.g., via the `nlshrink` or `corpcor` R packages) to improve numerical stability in high-dimensional settings.

Function Requirement:

- *Input*: Basis functions Z, treatment indicators A, and the inversion method (`direct`, `nlshrink`, or `corpcor`).
- *Output*: A pair of inverse matrices (Ω_1, Ω_0).

4.4 Compute the Projection Matrix

Compute the projection-like basis matrix B^a for each $a \in \{0, 1\}$:

$$B^a = Z\Omega_a(s^a \odot Z)^T, \quad \text{i.e.} \quad B_{ij}^a = Z_i^\top \Omega_a \, s_j^a Z_j, \tag{3}$$

where $Z \in \mathbb{R}^{n \times k}$ is the matrix of basis functions and $s^a \odot Z$ multiplies row j of Z by s_j^a (in code: `B1 = Z %*% Omega1 %*% t(Z * A)`). Note that B^a is an $n \times n$ matrix carrying the treatment indicator on its second index.

Function Requirement:

- *Input*: Basis matrix Z and the inverse matrices (Ω_1, Ω_0).
- *Output*: Basis matrices (B^1, B^0).

4.5 Compute the HOIF Estimators

Finally, we compute the HOIF bias-correction terms for the ATE.

Function Requirement:

- *Input*:
  - Maximum order m.
  - Residual pairs $((R_i^a, r_i^a))_{a \in \{0,1\}}$.
  - Projection matrices (B^1, B^0).

– Backend for ustat (`numpy` or `torch`).

- *Procedure*:

  For each order j from 2 to m, and for each treatment assignment $a \in \{0, 1\}$, calculate the U-statistics:

  $$\mathrm{U}_j^a = (-1)^j \text{ ustat(tensors} = T_j^a, \text{expression} = E_j^a, \text{backend} = \text{“backend”}, \text{average} = 1) \tag{4}$$

  The tensors T_j^a and index expressions E_j^a are defined as:

  $$T_j^a = (r^a, \underbrace{B^a, \ldots, B^a}_{j-1 \text{ times}}, R^a)$$
  $$E_j^a = (1, (1, 2), \ldots, (j-1, j), j)$$

  so that the first tensor r^a is indexed by i_1 and the last tensor R^a by i_j, matching the target formula.

  Then, for each order $l \in \{2, \ldots, m\}$, compute:

  $$\mathrm{IIFF}_l^a = \sum_{j=2}^{l} \binom{l-2}{l-j} \mathrm{U}_j^a$$
  $$\mathrm{HOIF}_l^a = \sum_{j=2}^{l} \mathrm{IIFF}_j^a$$
  $$\mathrm{BC}_l^{\mathrm{ATE}} = \mathrm{HOIF}_l^1 - \mathrm{HOIF}_l^0$$

- *Output*: $(\mathrm{BC}_l^{\mathrm{ATE}}, \mathrm{HOIF}_l^a, \mathrm{IIFF}_l^a)$ for $l = 2, \ldots, m$, returned by the package as `ATE`, `HOIF1`/`HOIF0` and `IIFF1`/`IIFF0`.

References

[1] Lin Liu and Chang Li. New $\sqrt{n}$-consistent, numerically stable higher-order influence function estimators. *arXiv preprint*, 2023.

[2] Lin Liu, Rajarshi Mukherjee, Whitney K Newey, and James M Robins. Semiparametric efficient empirical higher order influence function estimators. *arXiv preprint arXiv:1705.07577*, 2017.

[3] James M. Robins, Lingling Li, Rajarshi Mukherjee, Eric J. Tchetgen Tchetgen, and Aad van der Vaart. Minimax estimation of a functional on a structured high-dimensional model. *The Annals of Statistics*, 45(5):1951–1987, 2017.

[4] James M. Robins, Lingling Li, Eric Tchetgen Tchetgen, and Aad van der Vaart. Higher order influence functions and minimax estimation of nonlinear functionals. In *Probability and Statistics: Essays in Honor of David A. Freedman*, pages 335–421. Institute of Mathematical Statistics, Beachwood, OH, 2008. IMS Collections.