Adaptive Targeted Machine Learning of ATE using Highly Adaptive Lasso

Lars van der Laan
Joint work with Marco Carone, Alex Luedtke, Mark van der Laan

University of Washington

March 2024

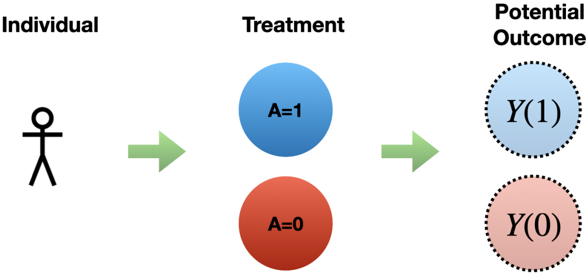
Problem setup

- Consider an **observational study** of *n iid* individuals assigned to either *treatment* or *control*.
- We record **baseline covariates** $W \in \mathbb{R}^d$, treatment indicator $A \in \{0, 1\}$, and an **outcome** Y, where $(W, A, Y) \sim P_0$.

Problem setup

- Consider an **observational study** of *n iid* individuals assigned to either *treatment* or *control*.
- We record **baseline covariates** $W \in \mathbb{R}^d$, treatment indicator $A \in \{0, 1\}$, and an **outcome** Y, where $(W, A, Y) \sim P_0$.
- Is treatment $A = 1$ better than control $A = 0$?
- To answer this question, we want inference on the average treatment effect (ATE).

Potential outcomes framework



What is $\mathbb{E}[Y(1) - Y(0)]$?

Identification of average treatment effect

- Assume:
  - (i) *Consistency:* $Y(A) = Y$.
  - (ii) *Randomization:* $(Y(0), Y(1)) \perp\!\!\!\perp A \mid W$.
  - (iii) *Positivity:* $1 > P_0(A = 1 \mid W) > 0$.

Identification of average treatment effect

- Assume:
  (i) *Consistency:* $Y(A) = Y$.
  (ii) *Randomization:* $(Y(0), Y(1)) \perp\!\!\!\perp A \mid W$.
  (iii) *Positivity:* $1 > P_0(A = 1 \mid W) > 0$.
- Then, the ATE is identified by a standardized difference in mean outcomes:

$$\mathbb{E}[Y(1) - Y(0)] = E_0\left[E_0(Y \mid A = 1, W) - E_0(Y \mid A = 0, W)\right]$$

Challenges in ATE estimation

- **Nonparametric** estimators require **strong positivity:**

$$1 - \delta > P_0(A = 1 \mid W) > \delta \text{ for } \delta > 0.$$

- Positivity **violations** are common and lead highly **variable** and **unstable** estimators.

Challenges in ATE estimation

- However, positivity **assumptions can be relaxed** if we know the functional form of the CATE:

$$\tau_0(W) := E_0[Y \mid A = 1, W] - E_0[Y \mid A = 0, W].$$

- We can extrapolate to areas of limited treatment overlap if the CATE is constant, linear, additive, etc in W.

Adaptive Targeted Machine Learning (ATMLE)

- Assuming a parametric model for the CATE *a priori* risks **misspecification bias**.
- Instead, we can be **data-adaptive** and learn a CATE model from data.
- **ATMLE**[1] is a framework that allows us to do adaptive model-selection, while still providing **valid inference** for the ATE.
- **How to learn model?** A HAL of the CATE intrinsically performs LASSO model selection over a rich spline basis.

[1] L. van der Laan, M. Carone, A. Luedtke, M. van der Laan (2023)

A risk function for the CATE

- Robinsons' transformation:

$$E_0[Y \mid A, W] = m_0(W) + (A - \pi_0(W))\tau_0(W),$$

with $m_0(W) = E_0[Y \mid W]$, $\pi_0(W) = P_0(A = 1 \mid W)$.

- Implies CATE τ_0 minimizes risk function[2]:

$$\tau \mapsto E_0\left[\{Y - m_0(W) - (A - \pi_0(W))\tau(W)\}^2\right].$$

- Rewrite the above as a weighted LS risk:

$$\tau \mapsto E_0\left[\omega_0(A, W)\left\{\frac{Y - m_0(W)}{A - \pi_0(W)} - \tau(W)\right\}^2\right],$$

where $\omega_0(A, W) = \{A - \pi_0(W)\}^2$.

[2] Xie and Wager (2017)

HAL-based R-learner of CATE

Step 1. **Learn nuisance functions:**

1. Regress $\{Y_i\}_{i=1}^n$ on $\{W_i\}_{i=1}^n$ using HAL to obtain estimate $\widehat{m}$ of m_0.
2. Regress $\{A_i\}_{i=1}^n$ on $\{W_i\}_{i=1}^n$ using HAL to obtain estimate $\widehat{\pi}$ of π_0.

Step 2. **Learn CATE:**

1. Get *pseudo-outcomes* $\{\widehat{Z}_i\}_{i=1}^n$ and *pseudo-weights* $\{\widehat{\omega}_i\}_{i=1}^n$:

$$\widehat{Z}_i := \frac{Y_i - \widehat{m}(W_i)}{A_i - \widehat{\pi}(W_i)}; \; \widehat{\omega}_i := \{A_i - \pi(W_i)\}^2 .$$

2. Obtain estimate $\widehat{\tau}$ of τ_0 by regressing $\{\widehat{Z}_i\}_{i=1}^n$ on $\{W_i\}_{i=1}^n$ with weights $\{\widehat{\omega}_i\}_{i=1}^n$ using (relaxed) HAL.

HAL-ATMLE for ATE

Step 1. **Learn nuisance functions:**

1. Regress $\{Y_i\}_{i=1}^n$ on $\{W_i\}_{i=1}^n$ using HAL to obtain estimate $\widehat{m}$ of m_0.
2. Regress $\{A_i\}_{i=1}^n$ on $\{W_i\}_{i=1}^n$ using HAL to obtain estimate $\widehat{\pi}$ of π_0.

Step 2. **Learn CATE:**

1. Get *pseudo-outcomes* $\{\widehat{Z}_i\}_{i=1}^n$ and *pseudo-weights* $\{\widehat{\omega}_i\}_{i=1}^n$:

$$\widehat{Z}_i := \frac{Y_i - \widehat{m}(W_i)}{A_i - \widehat{\pi}(W_i)};\ \widehat{\omega}_i := \{A_i - \pi(W_i)\}^2 .$$

2. Obtain estimate $\widehat{\tau}$ of τ_0 by regressing $\{\widehat{Z}_i\}_{i=1}^n$ on $\{W_i\}_{i=1}^n$ with weights $\{\widehat{\omega}_i\}_{i=1}^n$ using (relaxed) HAL.

Step 3. Plug-in to **learn ATE**: $\psi_n := \frac{1}{n}\sum_{i=1}^n \widehat{\tau}(W_i)$ and bootstrap with selected basis functions for confidence intervals.

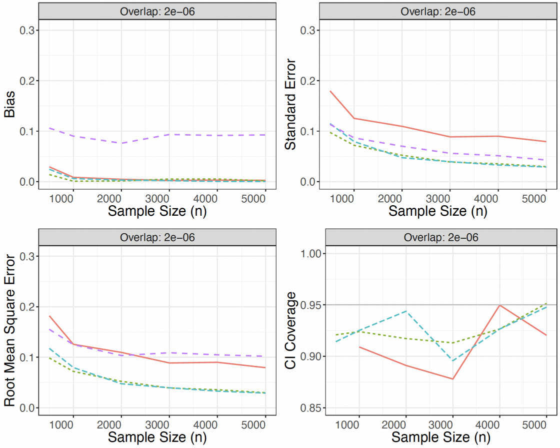
Simulation design: How does it perform?

- **Generating process:**
  - $X \in \mathbb{R}^4$ and varying levels of treatment overlap.
  - Normally distributed outcome with CATE piece-wise linear in some covariates:

$$\tau_0(x) := 1 + x_1 + |x_2| + \cos(4x_3) + x_4$$

- **Model selection:**
  - Specify additive basis for τ_0 using piece-wise linear hinge functions $x \mapsto \max\{x - t, 0\}$ with knot $t \in \mathbb{R}$.
  - CATE model $\mathcal{T}_n$ is learned using lasso-regularized R-learner over basis (total variation denoising/HAL).
- **Compare:** ATML (2 types) vs AIPW and semiparametric (intercept).

Simulation results: superefficiency



(a) Limited overlap ($c_0 \approx 10^{-6}$)

Figure 2: Comparison of empirical bias, standard error and root mean squared error of estimator, and coverage of nominal 95% confidence interval across 5000 MCMC replications for partially linear and plug-in HAL-ADMLEs, prespecified semiparametric estimator (assuming constant CATE), and nonparametric AIPW estimator, under sampling from a fixed distribution *not* satisfying linearity and with varying degrees of treatment overlap.

What is HAL-ATMLE estimating?

- Let $\mathcal{T}_n$ be the linear span of the spline basis functions selected using the HAL estimator $\widehat{\tau}$ of the CATE.
- ψ_n is an efficient estimator of the **data-adaptive parameter**:

$$\Psi_n(P) = E_P[\Pi_n \tau_P(W)]$$
$$\Pi_n \tau_P := \underset{\tau \in \mathcal{T}_n}{\operatorname{argmin}} \, E_P \left[\pi_P(X)\{1 - \pi_P(X)\} \{\tau_P(X) - \tau(X)\}^2 \right].$$

- We can show $\sqrt{n}\,(\psi_n - \Psi_n(P_0)) \to N(0, \sigma_0^2)$.
- What about the ATE $\Psi(P_0) = E_0[\tau_0(W)]$?

Oracle bias due to model approximation

- Under P_0, assume $\mathcal{T}_n$ asymptotically **approaches** some limiting **oracle model** $\mathcal{T}_0$ containing τ_0.
- If $\mathcal{T}_n \subseteq \mathcal{T}_0$, there exists a function $\gamma_0 \in \mathcal{T}_0$ such that

$$|\Psi_n(P_0) - \Psi(P_0)| \leq \|\gamma_0 - \Pi_n \gamma_0\| \|\tau_0 - \Pi_n \tau_0\|.$$

- If γ_0 and τ_0 have bounded sectional variation norm, then

$$\|\gamma_0 - \Pi_n \gamma_0\| \|\tau_0 - \Pi_n \tau_0\| = o_p(n^{-1/2});$$

$$\sqrt{n}\,(\psi_n - \Psi(P_0)) \to N(0, \sigma_0^2).$$

What else is HAL-ATMLE estimating?

- Under P_0, assume $\mathcal{T}_n$ asymptotically **approaches** some limiting **oracle model** $\mathcal{T}_0$ containing τ_0.

- Then, ψ_n is an efficient estimator of the **oracle parameter**:

$$\Psi_0(P) := E_P\left[\Pi_0\tau_P(X)\right]$$

$$\Pi_0\tau_P := \operatorname*{argmin}_{\tau\in\mathcal{T}_0} E_P\left[\pi_P(X)\{1-\pi_P(X)\}\left\{\tau_P(X)-\tau(X)\right\}^2\right].$$

Note:

- **Same estimand**: If $\tau_0 \in \mathcal{T}_0$, then $\Psi(P_0) = \Psi_0(P_0)$.

- **Different efficiency bound:** Efficiency bound of Ψ_0 driven by size of $\mathcal{T}_0$.

Concluding remarks

ATML is a general framework for adaptive and superefficient inference using data-driven model selection.

- ATML shows superefficiency is a continuum — not a dichotomy.
- ATML includes nonparametric regular and efficient estimators as a special case.
- ATML provides a means for nonparametric inference when regular estimators do not exist or behave poorly.
- ATML can beat any prespecified (semi)parametric estimator by learning a working model containing their model.

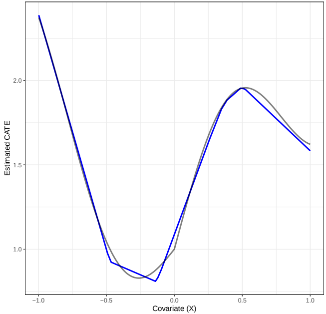
Setup and Data Generation

```
# Install causalHAL branch of hal9001 Github package
devtools::install_github("tlverse/hal9001@causalHAL")
library(hal9001)

# Generate data
n <- 1000
X <- runif(n, -1, 1)
pi.true <- plogis(-1 + abs(X) + X^2 + 0.5*sin(4*X))
A <- rbinom(n, 1, pi.true)
m.true <- 2*X^2 + pi.true * (1 + abs(X) + 0.5*sin(4*X))
cate.true <- (1 + abs(X) + 0.5*sin(4*X))
mu.true <- m.true + (A - pi.true) * cate.true
Y <- rnorm(n, mu.true, 0.2)
```

Estimate CATE using HAL

```
# HAL model fitting
cate_fit <- fit_hal_cate(X, Y, A,
        smoothness_orders = 1,
        num_knots = 100,
        max_degree = 1)
cate.hat <- predict(cate_fit, X)
```



Customize HAL nuisance estimators

```
fit_hal_cate(X, Y, A,
             smoothness_orders = 1,
             num_knots = 100,
              max_degree = 1,
             A_fit_params = list(smoothness_orders = 1,
                                 num_knots = 100,
                                 max_degree = 1),
             Y_fit_params = list(smoothness_orders = 1,
                                 num_knots = 100.
                                 max_degree = 1))
```

Specify custom nuisance estimates

```
# Estimate E[Y|X]
A_fit <- fit_hal(X, A, smoothness_orders = 1,
                 num_knots = 100, max_degree = 1,
                 return_cv_predictions = TRUE)
A.hat <- A_fit$cv_predictions

# Estimate E[Y|X]
Y_fit <- fit_hal(X, Y, smoothness_orders = 1,
                 num_knots = 100, max_degree = 1,
                 return_cv_predictions = TRUE)
Y.hat <- Y_fit$cv_predictions

# Pass in custom nuisance estimates.
cate_fit <- fit_hal_cate(X, Y, A, smoothness_orders = 1,
                    num_knots = 100, max_degree = 1
                    A.hat = A.hat, Y.hat = Y.hat)
```

Bootstrap-assisted Inference for CATE and ATE

```
# Bootstrap the HAL fit
bootstrapped_cate_fit <- bootstrap_hal(cate_fit)

# Pointwise inference on new data
out <- inference_pointwise(bootstrapped_cate_fit,
            new_data = X)
```

prediction <dbl>	CI_lower <dbl>	CI_right <dbl>
1.7722840	1.7165366	1.8261782
1.8839604	1.8154606	1.9537709
1.2509638	1.1762711	1.3141461
0.8408733	0.7752579	0.9203952
1.2511315	1.1764752	1.3143329
2.1626062	2.0774746	2.2412668

Inference for Functionals of CATE

```
# Functional inference for ATE
functional_mean <- function(hal_fit, X, ...) {
  mean(predict(hal_fit, X))
}
out <- inference_delta_method(
                      bootstrapped_cate_fit,
                      functional = functional_mean)
```

estimate <dbl>	CI_lower <dbl>	CI_right <dbl>
1.492985	1.449743	1.529833

Outline of general framework

- **Pathwise differentable parameter** $\Psi : \mathcal{M}_{np} \to \mathbb{R}$ on nonparametric model $\mathcal{M}_{np}$.
- **Learn from data** a working model $\mathcal{M}_n \subset \mathcal{M}_{np}$.
- Let $\mathcal{M}_n$ **stabilize** appropriately to an *oracle model* $\mathcal{M}_0$.
- Define projection-based **working parameter** and **oracle parameter**:

$$\Psi_n := \Psi \circ \Pi_n \text{ for } \Pi_n : \mathcal{M}_{np} \to \mathcal{M}_n;$$
$$\Psi_0 := \Psi \circ \Pi_0 \text{ for } \Pi_0 : \mathcal{M}_{np} \to \mathcal{M}_0$$

- **Oracle bias** is second order:

$$\Psi_n(P_0) - \Psi_0(P_0) = (\Pi_n P_0 - P_0)\{D_{\Psi_0,P_0} - \Pi_n D_{\Psi_0,P_0}\} + Rem_n.$$

- Construct **debiased** estimator ψ_n of $\Psi_n(P_0)$ using DML/TML.
- Under conditions, ψ_n is locally RAL and efficient for Ψ_0.

References

- Robinson, Peter M. "Root-N-consistent semiparametric regression." Econometrica: Journal of the Econometric Society (1988): 931-954.
- van der Laan, Lars, Marco Carone, Alex Luedtke, and Mark van der Laan. "Adaptive debiased machine learning using data-driven model selection techniques." arXiv preprint arXiv:2307.12544 (2023).