Acknowledgements: Rachael Phillips and Lars van der Laan

Highly Adaptive Lasso

In Causal Inference

Mark van der Laan

Professor of Biostatistics and Statistics
University of California, Berkeley

October 25, 2021, BASS-meeting

Traditional Lasso Estimator

- Advantages:
  - L_1-regularization performs both variable selection and penalized regression
  - Interpretable
  - Cross-validated selection of L_1-norm/penalty parameter
- Disadvantages:
  - Reliance on parametric forms places strong assumptions on the functional relationships between variables

HAL Advantages

- First estimator that guarantees asymptotically efficient estimation of any pathwise differentiable estimand[1] (e.g., the average causal effect or treatment-specific survival), without enforcing strong smoothness conditions.
- Assumptions are exceedingly mild, and expected to hold in almost every practical application.
- Can be implemented with standard Lasso software.
- Converges to true function at rate $n^{-1/3}(\log n)^{d/2}$.
- Accommodates a variety of function space specifications.

[1] An estimand that is a weakly differentiable functional of the density of the data, the case for most causal inference estimands under positivity.

Highly Adaptive Lasso (HAL)

A maximum likelihood estimator over all, or subset of, cadlag functions with finite variation norm.

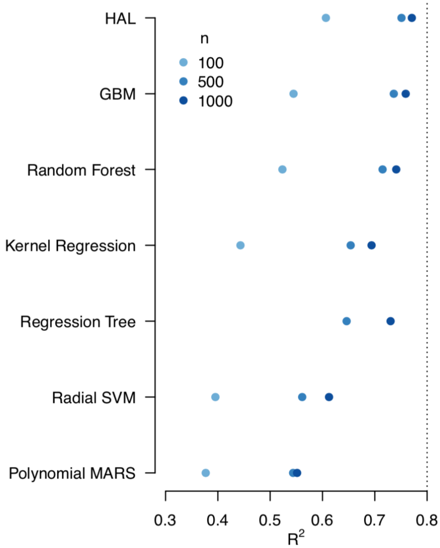
Key Ingredients

- Any stochastic relation/function we aim to learn from data can be approximated by linear combination (i.e., sum) of spline basis functions $X \rightarrow I(X > x_j)$ for knot point x_j.
- The variation norm (i.e., complexity) of a function is the L_1-norm.
- Optimize empirical performance over all such linear models under fixed L_1-norm that is selected with cross-validation.

---

van der Laan, Mark. "A generally efficient targeted minimum loss based estimator based on the highly adaptive lasso." The International Journal of Biostatistics (2017).

Theoretically proven to approximate truth faster than known machine learning algorithms

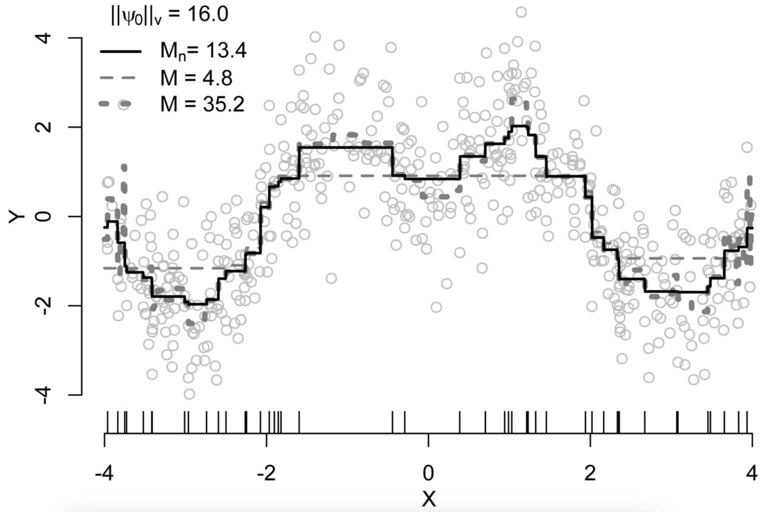


Highly Adaptive Lasso

Mark van der Laan

- Motivation
- Overview
- Implementation
- Applications of HAL
  - Super Learning
  - Propensity Score Estimation in TMLE
  - Efficient Plug-in Estimation
  - Nonparametric Bootstrap
- Concluding Remarks

Illustration in Low Dimensions



Tuning HAL

HAL's computational cost is controlled by the number of basis functions, which can be as large as $n * 2^{d-1}$.

Options for constraining the functional form of the target function:

- Enforce a minimum proportion of 1's in basis function.
- Enforce a maximal order of interaction.
- Specify particular additive model.
- Enforce monotonicity for some of the functions.
- Enforce higher order splines and thereby smoothness.
- Discretize continuous covariates: fewer knot-points.
- Greedy screening: iteratively, selected j-th order basis functions generate $j+1$-th order basis functions.

Basic R hal9001 Functionality

1. Load package and data:
```
library(hal9001)
data(mtcars)
```
2. Create numeric vector for dependent variable:
```
Y <- mtcars[,"mpg"]
```
3. Create dataframe or matrix of predictors:
```
X <- mtcars[,c("cyl", "disp", "hp", "wt")]
```
4. Fit HAL:
```
hal_fit <- fit_hal(X=X, Y=Y)
```

Note: default `max_degree=3` considers no more than 3-way interactions, and default `reduce_basis=0` places no restrictions on the minimum proportion of 1's in basis functions.

Summary table of hal9001 HAL fit


```
summary(hal_fit)$table
```

coef	term
35.4070	Intercept
-4.0801	I(disp >= 440)
-4.0118	I(disp >= 78.7)
-2.7170	I(wt >= 1.513)
-2.4454	I(wt >= 3.215)
-1.8184	I(disp >= 71.1)
-1.7208	I(hp >= 180)
-1.6830	I(disp >= 95.1)
-1.6039	I(hp >= 66)*I(wt >= 2.2)
-1.5623	I(wt >= 2.2)
1.3785	I(disp >= 351)
-1.2444	I(hp >= 175)
-1.1888	I(disp >= 301)
-0.9026	I(hp >= 123)
0.7336	I(hp >= 52)
-0.5810	I(disp >= 120.1)*I(hp >= 97)
-0.4395	I(disp >= 108)*I(hp >= 93)

Specifying `hal9001` model formulas

Example: Observe $O = (W_1, W_2, A, Y) \sim P_0$

R code: `fit_hal(Y, X, formula, data, ...)`

Additive model `formula`:

`Y` $\sim$. or `Y` $\sim$ `h(W1) + h(W2) + h(A)`

Bi-additive model `formula`:

`Y` $\sim$.^2 or

`Y` $\sim$ `h(W1)+h(W2)+h(A)+h(W1,W2)+h(W1,A)+h(W2,A)`

Only interactions with A `formula`:

`Y` $\sim$ `h(.)` $+$ `h(.,A)` or

`Y` $\sim$ `h(W1)+h(W2)+h(A)+h(W1,A)+h(W2,A)`

Monotone $\uparrow$ (i) $\downarrow$ (d) `formula` examples:

`Y` $\sim$ `i(.)` or `Y` $\sim$ `i(.)` $+$ `i(.,.)` or `Y` $\sim$ `i(W1)+d(W2)+i(A)`

Possible HAL fits under various smoothness orders

Example: Observe $(W, A, Y) \sim P_0$

R code: `fit_hal(Y, X, data, formula, s, ...)`

Example fits for 0-order smoothness, `s=0`:

Additive model:

$Y = \mathbb{I}(W > 0.5) + \mathbb{I}(W > 0.3) + \mathbb{I}(A > 0)$

Bi-additive model:

$Y = \mathbb{I}(W > 0.5) + \mathbb{I}(A > 0) + \mathbb{I}(W > 0.5, A > 0)$

Example fits for 1st-order smoothness, `s=1`:

Additive model:

$Y = \mathbb{I}(W > 0.5)[W - 0.5] + \mathbb{I}(W > 0.3)[W - 0.3] + \mathbb{I}(A > 0)[A - 0]$

Bi-additive model:

$Y = \mathbb{I}(W > 0.5)[W - 0.5] + \mathbb{I}(A > 0)[A - 0] + \mathbb{I}(W > 0.5, A > 0)[W - 0.5][A - 0]$

Performance under Various Screening Options

- Binning: Reducing number of spline knot points, separately for 1-way, 2-way, 3-way basis functions.
- Greedy screening: Build interactions sequentially from screened basis functions of lower order interactions.

$$n = 2000;\ d = 12$$

Size of regression matrix ($n \times p$) up to 3-way interactions:
Binning (100, 25, 5): $p = 30,000$
No Binning: $p = 600,000$

Sequential (up to 2-way): first p=20,000 then p=2,000, R^2=0.88, MSE=0.376
No Sequential (up to 2-way): p=150,000, R^2=0.85, MSE=0.394

More basis functions does not imply better performance.

Super Learner incorporating HAL

- By varying tuning parameters in HAL, one can include many HAL estimators in the library.
- The super learner will perform as well as the oracle choice among all these HAL estimators, and thereby achieves at minimal rate of convergence $n^{-1/3}(\log n)^{d/2}$.
- Can include other machine learning algorithms and parametric models as well.

Meta-learning with HAL

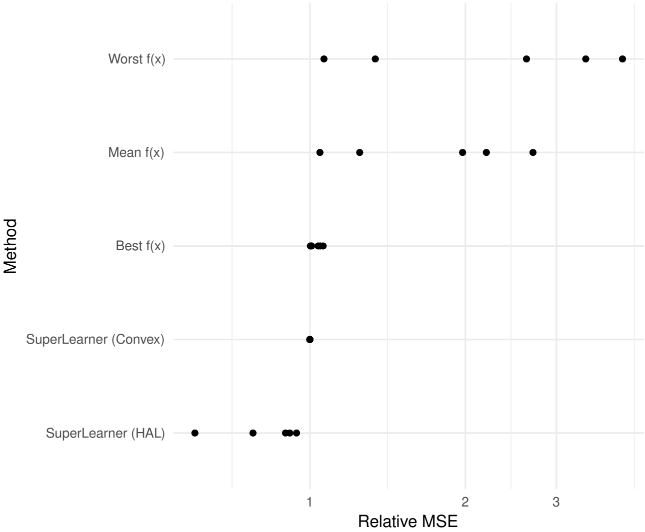
The super learner is defined by the the library of candidate estimators, loss function, and meta-learning algorithm.

Procedure for the Meta-HAL Super Learner

1. Perform meta-learning with HAL under specified L_1-norm.
2. Define a discrete super learner that includes as candidates the L_1-norm specific meta-HAL super learners, in order to optimally select the L_1-norm of the HAL meta-learner (double cross-validation).

This implementation guarantees the final discrete-selected meta-HAL will perform as well as the optimally tuned meta-HAL super learner.

Meta-HAL Super Learner



Outcome-regression weighted LASSO (OAL)

Shortreed & Artefaie (2017) proposed outcome-regression weighted Lasso (OAL) for propensity score (PS) estimation:

- Fit unpenalized linear model for $\mathbb{E}(Y|A, W)$:

$$(\hat{\alpha}, \hat{\eta}) = \arg\min_{\alpha,\eta} l_n(\alpha; Y, A, W).$$

  where η is the coefficient for A, and α is the coefficient for W.

- Denote the coefficient for variable W_j in the outcome regression with $\hat{\alpha}_j$.

- Fit PS with Lasso using regularization term

$$\lambda \sum_j ||\alpha_j||^{-\gamma}||\beta_j|| \text{ instead of usual } \sum_j \mid \beta_j \mid .$$

HAL-based OAL for PS Estimation

The theoretical property of OAL relies on the ***correct parametric formula***, which is often unknown in practice.

We extend OAL to outcome-regression weighted HAL (OHAL):

1. Compute the outcome regression using Lasso, with dependent variable the outcome Y and features the basis functions $\phi_{s,i}$ and the treatment indicator A.
2. Get the outcome regression coefficients $\alpha_{s,i}$ of $\phi_{s,i}$.
3. Compute the propensity score using a Lasso logistic regression, with dependent variable A and features $\phi_{s,i}$. The L_1-constraint for $\beta_{s,i}$, the coefficient for $\phi_{s,i}$, is defined as the weighted L_1-norm above.
4. Tune the L_1-norm with C-TMLE.

OHAL Performance based on Kang & Shafer (2007) Simulation

- Pre-treatment covariates $(Z_{i1}, \cdots, Z_{i4})$ are generated from uncorrelated standard normal distributions.
- Treatment indicator is then generated from a Bernoulli distribution with:

$$P(A_i = 1|Z_i) = \text{expit}(-Z_{i1} + 0.5Z_{i2} - 0.25Z_{i3} - 0.1Z_{i4})$$

- Potential outcomes are generated by:

$$Y_i^{(a)} = 210 + 27.4Z_{i1} + 13.7Z_{i2} + 13.7Z_{i3} + 13.7Z_{i4} + \epsilon$$
$$\epsilon \sim N(0,1)$$

- Thus, the value of the estimand, the ATE, is 0.

Simulation to Assess OHAL Performance (Continued)

- Only transformed covariates W are observed:

$$
\begin{aligned}
W_{i1} &= \exp(Z_{i1}/2) \\
W_{i2} &= z_{i2}/(1 + \exp(Z_{i1}) + 10 \\
W_{i3} &= (Z_{i1}Z_{i3}/25 + 0.6)^3 \\
W_{i4} &= (Z_2 + Z_4 + 20)^2.
\end{aligned}
$$

- In our experiment, we also included an instrumental variable W_{i5}, and the treatment mechanism was modified to:

$$P(A_i = 1 | Z_i, W_i) = \text{expit}(\frac{-Z_{i1} + 0.5Z_{i2} - 0.25Z_{i3} - 0.1Z_{i4}}{2} + W_{i5})$$

OHAL Simulation Results

We use main-term linear regression to create biased initial estimator $\bar{Q}_n^0$ for TMLE/C-TMLE.

	TMLE-HAL	CTMLE-HAL	CTMLE-OHAL	Oracle
N=500	7.06	6.34	**2.94**	3.64
N=1000	4.42	3.55	**1.40**	1.70
N=2000	2.94	1.85	**0.83**	0.87

Table: MSE for each estimator across 200 replications with different sample size.

HAL-TMLE is Asymptotically Efficient

- Due to the HAL-super learner converging to true nuisance parameter at faster rate than $n^{-1/4}$, the HAL-TMLE (using HAL for all nuisance parameters) is efficient in great generality (vdL, 15).
- The only necessary model assumptions are:
  - The true nuisance parameters have finite sectional variation norm
  - The loss functions of the true nuisance parameters are uniformly bounded, so that oracle inequality applies
  - The strong positivity assumption holds

Example: Asymptotic efficiency of HAL-TMLE for treatment-specific mean / ATE

Consider the HAL-TMLE of $EY_1 = EE(Y \mid A = 1, W)$ based on $(W, A, Y) \sim P_0$ in a nonparametric statistical model.

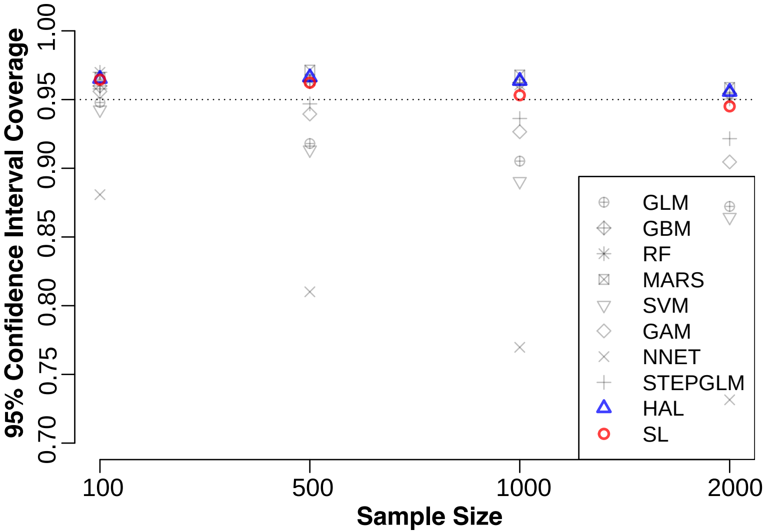
It is asymptotically efficient if

1. $\delta < P_0(A = 1 \mid W)$ for some $\delta > 0$
2. $W \rightarrow E_0(Y \mid A = 1, W)$ and $W \rightarrow P_0(A = 1 \mid W)$ are cadlag
3. $W \rightarrow E_0(Y \mid A = 1, W)$ and $W \rightarrow P_0(A = 1 \mid W)$ have finite sectional variation norm.

Can we break HAL-TMLE?

Highly Adaptive Lasso

Mark van der Laan



Undersmoothed HAL-MLE or Meta-HAL is efficient uniformly over large class of target estimands

- HAL-MLE is efficient for pathwise differentiable target estimands, if L_1-norm is chosen large enough.
- Using HAL in the meta-learning step of the super learner to determine the best functional combination of candidate estimators: Meta-HAL SL.
- Meta HAL-SL is efficient for pathwise differentiable target estimands if L_1-norm in meta-HAL is chosen large enough.
- Due to being an MLE, it solves a large class of score equations, in particular, efficient scores corresponding with target estimands. As a consequence, it can be analyzed as a plug-in TMLE, even though it is not targeted.

Nonparametric Bootstrap of HAL-TMLE

- Fix M at the cross-validation selector M_n or another selector (we propose a plateau selector!).
- Draw 10,000 samples of size n from empirical measure P_n. For each bootstrap sample $P_n^{\#}$, recompute the HAL-TMLE(M), say $\mathbf{P}_{n,M}^{\#*}$.
- The HAL on bootstrap sample can be restricted to only include indicator basis functions that were selected by HAL-MLE(M) on original data.
- Use sampling distribution of $\psi_{n,M}^{\#*} = \Psi(\mathbf{P}_{n,M}^{\#*})$, conditional on P_n, to construct 0.95-confidence interval.
- Increase M till plateau in confidence interval for optimal coverage.

Bootstrap works for HAL-TMLE

- Conditional on the data $(P_n : n \geq 1)$, the bootstrap sampling distribution of ψ_n^* converges to optimal normal limit distribution $N(0, \sigma_0^2)$.
- The approximation error of bootstrap is driven by performance of nonparametric bootstrap for an empirical process indexed by Donsker class (i.e., cadlag functions with sectional variation norm bounded by M).
- This suggests robust finite sample behavior of the nonparametric bootstrap.

Case Study

Compare two confidence intervals for ATE $EY_1 - EY_0$:

1. Wald-type
2. HAL-TMLE bootstrap, using plateau selection of $L1$-norm in HAL.

Simulation

Setting:

- $W \sim N(0, 4^2, -10, 10)$ drawn i.i.d. from a truncated normal distribution, bounded within [-10,10].
- $A \sim Bernoulli(p(W))$ with probability p(W) as a function of W bounded between [0.3, 0.7], given by $p(W) = 0.3 + 0.1Wsin(0.1W) + \varepsilon$, $\varepsilon \sim N(0, 0.05^2)$
- $Y = 3sin(a_1 W) + A + \varepsilon_2$ is a sinusoidal function of W, where $\varepsilon_2 \sim N(0, 1)$.
- a_1 controls the frequency (and true sectional variation norm) of the sinusoidal function.

The value of the parameter of interest, the ATE, is 1.

- The experiment is repeated 500 times, and interval coverages are computed

Simulation for $n = 100$

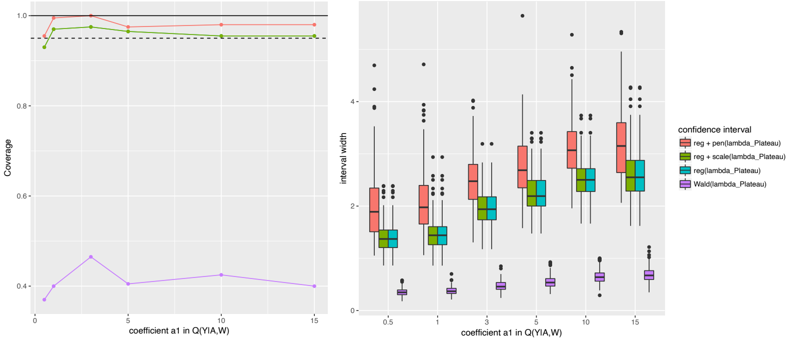


Figure: Coverage (left) and interval width (right) as a function of the a_1 coefficient (i.e., sectional variation norm) of the true data-generating distribution.

Concluding Remarks

- HAL is the first general nonparametric MLE.
- It converges at fast rate.
- HAL-TMLE is guaranteed asymptotically efficient.
- It provides finite sample robust TMLE for causal inference.
- It represents a class of HAL estimators by restricting function space, a priori, or data adaptively (screening).
- Its fit is a sparse representation, at most $n - 1$ coefficients.
- It allows for nonparametric bootstrap.
- It has many applications, such as meta-HAL super-learner and outcome adaptive HAL-TMLE.

Frequently Asked Questions

- Why are global smoothing assumptions better than local smoothing assumptions?
- If I include HAL in my Super Learner library, then why would I include anything else?
- How does screening HAL's basis functions affect it's rate of convergence to the true function?
- How does HAL compare to BART?