Chernozhukov et al. on Double / Debiased Machine Learning

Slides by Chris Felton
Princeton University, Sociology Department

Sociology Statistics Reading Group
November 2018
Papers

Background:

Main paper:
Papers

- **Background:**
Papers

**Background:**

**Main paper:**
Why This Paper?

Provides a general framework for estimating treatment effects using machine learning (ML) methods.

In particular, we can use any (preferably $n^{1/4}$-consistent) ML estimator with this approach.

Enables us to construct valid confidence intervals for our treatment effect estimates.

Introduces a $\sqrt{n}$-consistent estimator.

As $n \to \infty$, the estimation error $\hat{\theta} - \theta$ goes to zero at a rate of $n^{-1/2}$ (or $1/\sqrt{n}$).

We really like our estimators to be at least $\sqrt{n}$-consistent. $1/n^{1/2}$ will approach 0 more quickly than, e.g., $1/n^{1/4}$ as $n$ grows.
Why This Paper?

- Provides a **general** framework for estimating treatment effects using machine learning (ML) methods
Why This Paper?

- Provides a **general** framework for estimating treatment effects using machine learning (ML) methods
  - In particular, we can use any (preferably $n^{\frac{1}{4}}$-consistent) ML estimator with this approach
Why This Paper?

- Provides a **general** framework for estimating treatment effects using machine learning (ML) methods
  - In particular, we can use any (preferably $n^{\frac{1}{4}}$-consistent) ML estimator with this approach
- Enables us to construct valid **confidence intervals** for our treatment effect estimates
Why This Paper?

- Provides a **general** framework for estimating treatment effects using machine learning (ML) methods
  - In particular, we can use any (preferably \( n^{1/4} \)-consistent) ML estimator with this approach
- Enables us to construct valid **confidence intervals** for our treatment effect estimates
- Introduces a \( \sqrt{n} \)-consistent estimator
Why This Paper?

- Provides a **general** framework for estimating treatment effects using machine learning (ML) methods
  - In particular, we can use any (preferably $n^{1/4}$-consistent) ML estimator with this approach
- Enables us to construct valid **confidence intervals** for our treatment effect estimates
- Introduces a $\sqrt{n}$-**consistent** estimator
  - As $n \to \infty$, the estimation error $\hat{\theta} - \theta$ goes to zero at a rate of $n^{-1/2}$ (or $1/\sqrt{n}$)
Why This Paper?

- Provides a **general** framework for estimating treatment effects using machine learning (ML) methods
  - In particular, we can use any (preferably $n^{1/4}$-consistent) ML estimator with this approach
- Enables us to construct valid **confidence intervals** for our treatment effect estimates
- Introduces a $\sqrt{n}$-**consistent** estimator
  - As $n \to \infty$, the estimation error $\hat{\theta} - \theta$ goes to zero at a rate of $n^{-1/2}$ (or $1/\sqrt{n}$)
  - We really like our estimators to be at least $\sqrt{n}$-consistent
Why This Paper?

- Provides a **general** framework for estimating treatment effects using machine learning (ML) methods
  - In particular, we can use any (preferably $n^{1/4}$-consistent) ML estimator with this approach
- Enables us to construct valid **confidence intervals** for our treatment effect estimates
- Introduces a $\sqrt{n}$-**consistent** estimator
  - As $n \to \infty$, the estimation error $\hat{\theta} - \theta$ goes to zero at a rate of $n^{-1/2}$ (or $1/\sqrt{n}$)
  - We really like our estimators to be at least $\sqrt{n}$-consistent
  - $1/n^{1/2}$ will approach 0 more quickly than, e.g., $1/n^{1/4}$ as $n$ grows
Why Use ML for Causal Inference, Anyway?

In observational studies, we often estimate causal effects by conditioning on confounders. We typically condition on confounders by making strong assumptions about the functional form of our model. E.g., a standard OLS model assumes a linear and additive conditional expectation function. If we misspecify the functional form, we will end up with biased estimates of treatment effects even in the absence of unmeasured confounding. Our parametric specifications often lack strong substantive justification. ML provides a systematic framework for learning the form of the conditional expectation function from the data.
In observational studies, we often estimate causal effects by conditioning on confounders.
Why Use ML for Causal Inference, Anyway?

- In observational studies, we often estimate causal effects by conditioning on confounders.
- We typically condition on confounders by making **strong assumptions** about the functional form of our model.
Why Use ML for Causal Inference, Anyway?

- In observational studies, we often estimate causal effects by conditioning on confounders.
- We typically condition on confounders by making strong assumptions about the functional form of our model.
  - E.g., a standard OLS model assumes a linear and additive conditional expectation function.
Why Use ML for Causal Inference, Anyway?

- In observational studies, we often estimate causal effects by conditioning on confounders.
- We typically condition on confounders by making strong assumptions about the functional form of our model.
  - E.g., a standard OLS model assumes a linear and additive conditional expectation function.
- If we misspecify the functional form, we will end up with biased estimates of treatment effects even in the absence of unmeasured confounding.
Why Use ML for Causal Inference, Anyway?

- In observational studies, we often estimate causal effects by conditioning on confounders.
- We typically condition on confounders by making **strong assumptions** about the functional form of our model.
  - E.g., a standard OLS model assumes a linear and additive conditional expectation function.
- If we misspecify the functional form, we will end up with **biased estimates** of treatment effects even in the absence of unmeasured confounding.
- Our parametric specifications often **lack strong substantive justification**.
Why Use ML for Causal Inference, Anyway?

- In observational studies, we often estimate causal effects by conditioning on confounders.
- We typically condition on confounders by making strong assumptions about the functional form of our model.
  - E.g., a standard OLS model assumes a linear and additive conditional expectation function.
- If we misspecify the functional form, we will end up with biased estimates of treatment effects even in the absence of unmeasured confounding.
- Our parametric specifications often lack strong substantive justification.
- ML provides a systematic framework for learning the form of the conditional expectation function from the data.
Why Use ML for Causal Inference, Anyway?

We sometimes find ourselves working with high-dimensional data. I.e., we have many covariates $p$ relative to the number of observations $n$.

Two types of high-dimensionality:

1. We simply have many measured covariates (e.g., text data, genetic data).
2. We have few measured covariates but wish to generate many non-linear transformations of and interactions between these covariates.

ML models perform much better in high dimensions than traditional statistical models do.
Why Use ML for Causal Inference, Anyway?

- We sometimes find ourselves working with high-dimensional data.
Why Use ML for Causal Inference, Anyway?

- We sometimes find ourselves working with high-dimensional data
  - i.e., we have many covariates $p$ relative to the number of observations $n$
Why Use ML for Causal Inference, Anyway?

- We sometimes find ourselves working with high-dimensional data
  - I.e., we have many covariates \( p \) relative to the number of observations \( n \)
- Two types of high-dimensionality:
Why Use ML for Causal Inference, Anyway?

- We sometimes find ourselves working with high-dimensional data
  - I.e., we have many covariates $p$ relative to the number of observations $n$
- Two types of high-dimensionality:
  1. We simply have many measured covariates (e.g., text data, genetic data)
Why Use ML for Causal Inference, Anyway?

- We sometimes find ourselves working with **high-dimensional** data
  - I.e., we have many covariates $p$ relative to the number of observations $n$

- Two types of high-dimensionality:
  1. We simply have many measured covariates (e.g., text data, genetic data)
  2. We have few measured covariates but wish to generate many non-linear transformations of and interactions between these covariates
Why Use ML for Causal Inference, Anyway?

- We sometimes find ourselves working with high-dimensional data
  - i.e., we have many covariates $p$ relative to the number of observations $n$

Two types of high-dimensionality:

1. We simply have many measured covariates (e.g., text data, genetic data)
2. We have few measured covariates but wish to generate many non-linear transformations of and interactions between these covariates

- ML models perform much better in high dimensions than traditional statistical models do
Why Use ML for Causal Inference, Anyway?

1. ML allows us to do causal inference with minimal assumptions about the functional form of our model.

2. Warning: ML does not help us relax identification assumptions (e.g., no unmeasured confounding, parallel trends, exclusion restriction, etc.).

3. ML allows us to do causal inference with high-dimensional data.
Why Use ML for Causal Inference, Anyway?

ML allows us to do causal inference with **minimal assumptions about the functional form** of our model.
Why Use ML for Causal Inference, Anyway?

ML allows us to do causal inference with **minimal assumptions about the functional form** of our model.

- **Warning:** ML does not help us relax identification assumptions (e.g., no unmeasured confounding, parallel trends, exclusion restriction, etc.)
Why Use ML for Causal Inference, Anyway?

1. ML allows us to do causal inference with **minimal assumptions about the functional form** of our model
   - **Warning:** ML does not help us relax identification assumptions (e.g., no unmeasured confounding, parallel trends, exclusion restriction, etc.)

2. ML allows us to do causal inference with **high-dimensional data**
Why Using ML for Causal Inference is Tricky

ML methods were designed for prediction. But off-the-shelf ML methods are biased estimators for treatment effects. To minimize $\text{MSE} = \text{bias}^2 + \text{variance}$, we trade off variance for bias. Consistent ML methods converge more slowly than $\frac{1}{\sqrt{n}}$. Off-the-shelf methods also fail to provide confidence intervals for our treatment effect estimates.
Why Using ML for Causal Inference is Tricky

- ML methods were designed for prediction
Why Using ML for Causal Inference is Tricky

- ML methods were designed for prediction
- But off-the-shelf ML methods are biased estimators for treatment effects
Why Using ML for Causal Inference is Tricky

- ML methods were designed for prediction
- But off-the-shelf ML methods are biased estimators for treatment effects
  - To minimize MSE = bias² + variance, we trade off variance for bias
Why Using ML for Causal Inference is Tricky

- ML methods were designed for prediction
- But off-the-shelf ML methods are biased estimators for treatment effects
  - To minimize $\text{MSE} = \text{bias}^2 + \text{variance}$, we trade off variance for bias
- Consistent ML methods converge more slowly than $\frac{1}{\sqrt{n}}$
Why Using ML for Causal Inference is Tricky

- ML methods were designed for prediction
- But off-the-shelf ML methods are **biased** estimators for treatment effects
  - To minimize $\text{MSE} = \text{bias}^2 + \text{variance}$, we trade off variance for bias
- Consistent ML methods converge more slowly than $\frac{1}{\sqrt{n}}$
- Off-the-shelf methods also fail to provide confidence intervals for our treatment effect estimates
Key Aims of Double / Debiased Machine Learning (DML)
Key Aims of Double / Debiased Machine Learning (DML)

1. Eliminate the bias
Key Aims of Double / Debiased Machine Learning (DML)

1. Eliminate the bias

2. Achieve $\sqrt{n}$-consistency
Key Aims of Double / Debiased Machine Learning (DML)

1. Eliminate the bias
2. Achieve $\sqrt{n}$-consistency
3. Construct valid confidence intervals
Outline of Presentation

1. Introduce partially linear model set-up
2. Explain two sources of estimation bias from ML and how we overcome them
   - Correct bias from regularization with Neyman orthogonality
     We will see that we can achieve Neyman orthogonality using a residuals-on-residuals approach reminiscent of Robinson (1988) and the Frisch–Waugh–Lovell theorem
   - Correct bias from overfitting using sample-splitting
     Employ cross-fitting to avoid the loss of efficiency that normally comes with sample-splitting
3. Outline a procedure for conducting inference with DML
4. Examine estimators for the ATE and variance that go beyond the partially linear model set-up
Outline of Presentation

1. Introduce partially linear model set-up

2. Explain two sources of estimation bias from ML and how we overcome them
   - Correct bias from regularization with Neyman orthogonality
     We will see that we can achieve Neyman orthogonality using a residuals-on-residuals approach reminiscent of Robinson (1988) and the Frisch–Waugh–Lovell theorem
   - Correct bias from overfitting using sample-splitting
     Employ cross-fitting to avoid the loss of efficiency that normally comes with sample-splitting

3. Outline a procedure for conducting inference with DML

4. Examine estimators for the ATE and variance that go beyond the partially linear model set-up
Outline of Presentation

1. Introduce partially linear model set-up
2. Explain two sources of estimation bias from ML and how we overcome them
Outline of Presentation

1. Introduce partially linear model set-up
2. Explain two sources of estimation bias from ML and how we overcome them
   - Correct bias from regularization with Neyman orthogonality
3. Outline a procedure for conducting inference with DML
4. Examine estimators for the ATE and variance that go beyond the partially linear model set-up
Outline of Presentation

1. Introduce partially linear model set-up
2. Explain two sources of estimation bias from ML and how we overcome them
   - Correct bias from *regularization* with *Neyman orthogonality*
     - We will see that we can achieve Neyman orthogonality using a residuals-on-residuals approach reminiscent of Robinson (1988) and the Frisch–Waugh–Lovell theorem
Outline of Presentation

1. Introduce partially linear model set-up
2. Explain two sources of estimation bias from ML and how we overcome them
   - Correct bias from regularization with Neyman orthogonality
     - We will see that we can achieve Neyman orthogonality using a residuals-on-residuals approach reminiscent of Robinson (1988) and the Frisch–Waugh–Lovell theorem
   - Correct bias from overfitting using sample-splitting
Outline of Presentation

1. Introduce partially linear model set-up
2. Explain two sources of estimation bias from ML and how we overcome them
   - Correct bias from **regularization** with **Neyman orthogonality**
     - We will see that we can achieve Neyman orthogonality using a residuals-on-residuals approach reminiscent of Robinson (1988) and the Frisch–Waugh–Lovell theorem
   - Correct bias from **overfitting** using **sample-splitting**
     - Employ **cross-fitting** to avoid the loss of efficiency that normally comes with sample-splitting
Outline of Presentation

1. Introduce partially linear model set-up
2. Explain two sources of estimation bias from ML and how we overcome them
   - Correct bias from regularization with Neyman orthogonality
     - We will see that we can achieve Neyman orthogonality using a residuals-on-residuals approach reminiscent of Robinson (1988) and the Frisch–Waugh–Lovell theorem
   - Correct bias from overfitting using sample-splitting
     - Employ cross-fitting to avoid the loss of efficiency that normally comes with sample-splitting
3. Outline a procedure for conducting inference with DML
Outline of Presentation

1. Introduce partially linear model set-up

2. Explain two sources of estimation bias from ML and how we overcome them
   - Correct bias from **regularization** with **Neyman orthogonality**
     - We will see that we can achieve Neyman orthogonality using a residuals-on-residuals approach reminiscent of Robinson (1988) and the Frisch–Waugh–Lovell theorem
   - Correct bias from **overfitting** using **sample-splitting**
     - Employ **cross-fitting** to avoid the loss of efficiency that normally comes with sample-splitting

3. Outline a procedure for conducting inference with DML

4. Examine estimators for the ATE and variance that go beyond the partially linear model set-up
Don’t worry if none of that makes sense yet!
Don’t worry if none of that makes sense yet!

It’s not as complicated as it sounds, and we will work through it slowly.
The Partially Linear Model Set-up
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]

\[ D = m_0(X) + V \]
The Partially Linear Model Set-up

\[ Y = D \theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- \( Y \): Outcome
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- **Y**: Outcome
- **D**: Treatment
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- \( Y \): Outcome
- \( D \): Treatment
- \( X \): Measured confounders
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- **Y**: Outcome
- **D**: Treatment
- **X**: Measured confounders
- **U** and **V** are our error terms
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- **Y**: Outcome
- **D**: Treatment
- **X**: Measured confounders
- **U** and **V** are our error terms
- We assume zero conditional mean:
  \[ \mathbb{E}[U \mid X, D] = 0 \quad \mathbb{E}[V \mid X] = 0 \]
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]

\[ D = m_0(X) + V \]

\( \theta_0 \): The true treatment effect

Warning: not necessarily the Average Treatment Effect

Regression gives us a weighted average of individual treatment effects where weights are determined by the conditional variance of treatment (see Aronow and Samii 2016)
The Partially Linear Model Set-up

\[ Y = D \theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- \( \theta_0 \): The true treatment effect
  - “theta-naught”
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- \( \theta_0 \): The true treatment effect
  - “theta-naught”
  - **Warning:** not necessarily the Average Treatment Effect
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- \( \theta_0 \): The true treatment effect
  - “theta-naught”
  - **Warning**: not necessarily the Average Treatment Effect

  Regression gives us a weighted average of individual treatment effects where weights are determined by the conditional variance of treatment (see Aronow and Samii 2016)
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- \( \theta_0 \): The true treatment effect
  - “theta-naught”
  - **Warning**: not necessarily the Average Treatment Effect
    - Regression gives us a weighted average of individual treatment effects where weights are determined by the conditional variance of treatment (see Aronow and Samii 2016)

- \( g_0(\cdot) \): some function mapping \( X \) to \( Y \), conditional on \( D \)
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- \( \theta_0 \): The true treatment effect
  - “theta-naught”
  - **Warning:** not necessarily the Average Treatment Effect
    - Regression gives us a weighted average of individual treatment effects where weights are determined by the conditional variance of treatment (see Aronow and Samii 2016)

- \( g_0(\cdot) \): some function mapping \( X \) to \( Y \), conditional on \( D \)

- \( m_0(\cdot) \): some function mapping \( X \) to \( D \)
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U\]
\[ D = m_0(X) + V\]

- This set-up allows both \( Y \) and \( D \) to be non-linear and interactive functions of \( X \) in contrast to standard OLS, which assumes a linear and additive model.
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- This set-up allows both \( Y \) and \( D \) to be non-linear and interactive functions of \( X \) in contrast to standard OLS, which assumes a linear and additive model.
- However, note that this partially linear model assumes that the effect of \( D \) on \( Y \) is **additive and linear**.
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- This set-up allows both \( Y \) and \( D \) to be non-linear and interactive functions of \( X \) in contrast to standard OLS, which assumes a linear and additive model.
- However, note that this partially linear model assumes that the effect of \( D \) on \( Y \) is additive and linear.
- Our confounders can interact with one another, but not with our treatment!
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- This set-up allows both \( Y \) and \( D \) to be non-linear and interactive functions of \( X \) in contrast to standard OLS, which assumes a linear and additive model.
- However, note that this partially linear model assumes that the effect of \( D \) on \( Y \) is **additive** and **linear**.
- Our confounders can interact with one another, but not with our treatment!
- And remember we’re still making the standard identification assumptions (unconfoundedness conditional on \( X \), positivity, and consistency).
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

If ML is useful because it allows us to relax linearity and additivity, why would we assume linearity and additivity in \( D \)?
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- If ML is useful because it allows us to relax linearity and additivity, why would we assume linearity and additivity in \( D \)?
- We’re just using this model for illustration!
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]
\[ D = m_0(X) + V \]

- If ML is useful because it allows us to relax linearity and additivity, why would we assume linearity and additivity in \( D \)?
- We’re just using this model for illustration!
- We can assume a fully interactive and non-linear model when we actually use DML.
The Partially Linear Model Set-up

\[ Y = D\theta_0 + g_0(X) + U \]

\[ D = m_0(X) + V \]

- If ML is useful because it allows us to relax linearity and additivity, why would we assume linearity and additivity in \( D \)?
- We’re just using this model for illustration!
- We can assume a fully interactive and non-linear model when we actually use DML
- But our partially linear model set-up will allow us to better explain how DML works
Where We Are and Where We’re Going

We've introduced the partially linear model set-up. Next, we will introduce an intuitive procedure—"the naive approach"—for estimating $\theta_0$ with ML assuming a partially linear model. We will show that this estimation procedure is biased and not $\sqrt{n}$-consistent. Then we're going to illustrate two sources of this bias and show how DML avoids these two types of bias.
Where We Are and Where We’re Going

- We’ve introduced the partially linear model set-up
Where We Are and Where We’re Going

- We’ve introduced the partially linear model set-up
- Next, we will introduce an intuitive procedure—“the naive approach”—for estimating $\theta_0$ with ML assuming a partially linear model
Where We Are and Where We’re Going

- We’ve introduced the partially linear model set-up
- Next, we will introduce an intuitive procedure—“the naive approach”—for estimating $\theta_0$ with ML assuming a partially linear model
- We will show that this estimation procedure is biased and not $\sqrt{n}$-consistent
Where We Are and Where We’re Going

- We’ve introduced the partially linear model set-up
- Next, we will introduce an intuitive procedure—“the naive approach”—for estimating \( \theta_0 \) with ML assuming a partially linear model
- We will show that this estimation procedure is biased and not \( \sqrt{n} \)-consistent
- Then we’re going to illustrate two sources of this bias and show how DML avoids these two types of bias
Causal Inference with ML: The Naive Approach

The naive approach: estimate \( Y = D \hat{\theta}_0 + \hat{g}_0(X) + \hat{U} \) using ML. \( \hat{\theta}_0 \) : our naive estimate of \( \theta_0 \), "theta-naught-hat"

How might we estimate \( \hat{\theta}_0 \) and \( \hat{g}_0(X) \)?

Remember, \( m_0(X) = E[D|X] \), but \( g_0(X) \neq E[Y|X] \)

That's because \( D \theta_0 \) is also included in this model.

In the paper, the authors use \( \ell_0(X) \) for \( E[Y|X] \)

To estimate both \( \hat{\theta}_0 \) and \( \hat{g}_0(X) \), we could use an iterative method that alternates between using random forest for estimating \( \hat{g}_0(X) \) and OLS for estimating \( \hat{\theta}_0 \).

Alternatively, we could generate many non-linear transformations of the covariates in \( X \) as well as interactions between these covariates and use LASSO to estimate the model.
Causal Inference with ML: The Naive Approach

- The naive approach: estimate $Y = D\hat{\theta}_0 + \hat{g}_0(X) + \hat{U}$ using ML
Causal Inference with ML: The Naive Approach

- The naive approach: estimate \( Y = D\hat{\theta}_0 + \hat{g}_0(X) + \hat{U} \) using ML
  - \( \hat{\theta}_0 \): our naive estimate of \( \theta_0 \), “theta-naught-hat”
Causal Inference with ML: The Naive Approach

- The naive approach: estimate $Y = D\hat{\theta}_0 + \hat{g}_0(X) + \hat{U}$ using ML
  - $\hat{\theta}_0$: our naive estimate of $\theta_0$, “theta-naught-hat”
- How might we estimate $\hat{\theta}_0$ and $\hat{g}_0(X)$?
Causal Inference with ML: The Naive Approach

- The naive approach: estimate $Y = D\hat{\theta}_0 + \hat{g}_0(X) + \hat{U}$ using ML
  - $\hat{\theta}_0$: our naive estimate of $\theta_0$, “theta-naught-hat”
- How might we estimate $\hat{\theta}_0$ and $\hat{g}_0(X)$?
  - Remember, $m_0(X) = E[D \mid X]$, but $g_0(X) \neq E[Y \mid X]$
Causal Inference with ML: The Naive Approach

- The naive approach: estimate \( Y = D\hat{\theta}_0 + \hat{g}_0(X) + \hat{U} \) using ML
  - \( \hat{\theta}_0 \): our naive estimate of \( \theta_0 \), “theta-naught-hat”

- How might we estimate \( \hat{\theta}_0 \) and \( \hat{g}_0(X) \)?
  - Remember, \( m_0(X) = E[D \mid X] \), but \( g_0(X) \neq E[Y \mid X] \)
  - That’s because \( D\theta_0 \) is also included in this model
Causal Inference with ML: The Naive Approach

- The naive approach: estimate \( Y = D\hat{\theta}_0 + \hat{g}_0(X) + \hat{U} \) using ML
  - \( \hat{\theta}_0 \): our naive estimate of \( \theta_0 \), “theta-naught-hat”

- How might we estimate \( \hat{\theta}_0 \) and \( \hat{g}_0(X) \)?
  - Remember, \( m_0(X) = E[D \mid X] \), but \( g_0(X) \neq E[Y \mid X] \)
  - That’s because \( D\theta_0 \) is also included in this model
  - In the paper, the authors use \( \ell_0(X) \) for \( E[Y \mid X] \)
Causal Inference with ML: The Naive Approach

- The naive approach: estimate \( Y = D\hat{\theta}_0 + \hat{g}_0(X) + \hat{U} \) using ML
  - \( \hat{\theta}_0 \): our naive estimate of \( \theta_0 \), “theta-naught-hat”
- How might we estimate \( \hat{\theta}_0 \) and \( \hat{g}_0(X) \)?
  - Remember, \( m_0(X) = E[D \mid X] \), but \( g_0(X) \neq E[Y \mid X] \)
  - That’s because \( D\theta_0 \) is also included in this model
  - In the paper, the authors use \( \ell_0(X) \) for \( E[Y \mid X] \)
- To estimate both \( \hat{\theta}_0 \) and \( \hat{g}_0(X) \), we could use an iterative method that alternates between using random forest for estimating \( \hat{g}_0(X) \) and OLS for estimating \( \hat{\theta}_0 \)
Causal Inference with ML: The Naive Approach

- The naive approach: estimate $Y = D\hat{\theta}_0 + \hat{g}_0(X) + \hat{U}$ using ML
  - $\hat{\theta}_0$: our naive estimate of $\theta_0$, “theta-naught-hat”
- How might we estimate $\hat{\theta}_0$ and $\hat{g}_0(X)$?
  - Remember, $m_0(X) = \text{E}[D \mid X]$, but $g_0(X) \neq \text{E}[Y \mid X]$,
    That’s because $D\theta_0$ is also included in this model
  - In the paper, the authors use $\ell_0(X)$ for $\text{E}[Y \mid X]$.
- To estimate both $\hat{\theta}_0$ and $\hat{g}_0(X)$, we could use an iterative method that alternates between using random forest for estimating $\hat{g}_0(X)$ and OLS for estimating $\hat{\theta}_0$
- Alternatively, we could generate many non-linear transformations of the covariates in $X$ as well as interactions between these covariates and use LASSO to estimate the model.
Two Sources of Bias in Our Naive Estimator

1. Bias from regularization

To avoid overfitting the data with complex functional forms, ML algorithms use regularization. This decreases the variance of the estimator and reduces overfitting... but introduces bias and prevents $\sqrt{n}$-consistency.

2. Bias from overfitting

Sometimes our efforts to regularize fail to prevent overfitting. Overfitting: mistaking noise for signal. More carefully, we overfit when we model the idiosyncrasies of our particular sample too closely, which may lead to poor out-of-sample performance.

Overfitting $\rightarrow$ bias and slow convergence.
Two Sources of Bias in Our Naive Estimator

1. Bias from regularization

Bias from overfitting

Sometimes our efforts to regularize fail to prevent overfitting. Overfitting occurs when we mistook noise for signal. More carefully, we overfit when we model the idiosyncrasies of our particular sample too closely, which may lead to poor out-of-sample performance. Overfitting leads to bias and slow convergence.
Two Sources of Bias in Our Naive Estimator

1. Bias from *regularization*
   - To avoid overfitting the data with complex functional forms, ML algorithms use regularization
Two Sources of Bias in Our Naive Estimator

1. Bias from **regularization**
   - To avoid overfitting the data with complex functional forms, ML algorithms use regularization
   - This decreases the variance of the estimator and reduces overfitting...
Two Sources of Bias in Our Naive Estimator

1. Bias from **regularization**
   - To avoid overfitting the data with complex functional forms, ML algorithms use regularization.
   - This decreases the variance of the estimator and reduces overfitting...
   - ...but introduces bias and prevents $\sqrt{n}$-consistency.
Two Sources of Bias in Our Naive Estimator

1. Bias from **regularization**
   - To avoid overfitting the data with complex functional forms, ML algorithms use regularization.
   - This decreases the variance of the estimator and reduces overfitting...
   - ...but introduces bias and prevents $\sqrt{n}$-consistency.

2. Bias from **overfitting**
Two Sources of Bias in Our Naive Estimator

1. Bias from regularization
   - To avoid overfitting the data with complex functional forms, ML algorithms use regularization
   - This decreases the variance of the estimator and reduces overfitting...
   - ...but introduces bias and prevents $\sqrt{n}$-consistency

2. Bias from overfitting
   - Sometimes our efforts to regularize fail to prevent overfitting
Two Sources of Bias in Our Naive Estimator

1. Bias from **regularization**
   - To avoid overfitting the data with complex functional forms, ML algorithms use regularization
   - This decreases the variance of the estimator and reduces overfitting...
   - ...but introduces bias and prevents \(\sqrt{n}\)-consistency

2. Bias from **overfitting**
   - Sometimes our efforts to regularize fail to prevent overfitting
   - Overfitting: mistaking noise for signal
Two Sources of Bias in Our Naive Estimator

1. Bias from **regularization**
   - To avoid overfitting the data with complex functional forms, ML algorithms use regularization
   - This decreases the variance of the estimator and reduces overfitting...
   - ...but introduces bias and prevents $\sqrt{n}$-consistency

2. Bias from **overfitting**
   - Sometimes our efforts to regularize fail to prevent overfitting
   - Overfitting: mistaking noise for signal
   - More carefully, we overfit when we model the idiosyncrasies of our particular sample too closely, which may lead to poor out-of-sample performance
Two Sources of Bias in Our Naive Estimator

1. Bias from **regularization**
   - To avoid overfitting the data with complex functional forms, ML algorithms use regularization
   - This decreases the variance of the estimator and reduces overfitting...
   - ...but introduces bias and prevents $\sqrt{n}$-consistency

2. Bias from **overfitting**
   - Sometimes our efforts to regularize fail to prevent overfitting
   - Overfitting: mistaking noise for signal
   - More carefully, we overfit when we model the idiosyncrasies of our particular sample too closely, which may lead to poor out-of-sample performance
   - Overfitting $\rightarrow$ bias and slow convergence
Two Sources of Bias in Our Naive Estimator
Two Sources of Bias in Our Naive Estimator

- For clarity, we will isolate each type of bias
Two Sources of Bias in Our Naive Estimator

- For clarity, we will isolate each type of bias.

- When we look at regularization bias, we will assume we have used sample-splitting to avoid bias from overfitting.
Two Sources of Bias in Our Naive Estimator

- For clarity, we will isolate each type of bias.

- When we look at regularization bias, we will assume we have used sample-splitting to avoid bias from overfitting.

- When we look at bias from overfitting, we will assume we have used orthogonalization to prevent regularization bias.
Two Sources of Bias in Our Naive Estimator

- For clarity, we will isolate each type of bias

- When we look at regularization bias, we will assume we have used sample-splitting to avoid bias from overfitting

- When we look at bias from overfitting, we will assume we have used orthogonalization to prevent regularization bias

- We’ll explain how sample-splitting and orthogonalization work soon
Regularization Bias

Let’s start by looking at the scaled estimation error in $\hat{\theta}_0$ when we use sample-splitting without orthogonalization
Regularization Bias

\[ \sqrt{n}(\hat{\theta}_0 - \theta_0) = \left( \frac{1}{n} \sum_{i \in L} D_i^2 \right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in L} D_i U_i \]
\[ := a \]
\[ + \left( \frac{1}{n} \sum_{i \in L} D_i^2 \right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in L} D_i (g_0(X_i) - \hat{g}_0(X_i)) \]
\[ := b \]
Regularization Bias

$$\sqrt{n}(\hat{\theta}_0 - \theta_0) = \left(\frac{1}{n} \sum_{i \in I} D_i^2 \right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} D_i U_i$$

$$:= a$$

$$+ \left(\frac{1}{n} \sum_{i \in I} D_i^2 \right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} D_i (g_0(X_i) - \hat{g}_0(X_i))$$

$$:= b$$

- This looks scary, so let’s take it one term at a time
Regularization Bias

\[
\sqrt{n}(\hat{\theta}_0 - \theta_0) = \left(\frac{1}{n} \sum_{i \in I} D_i^2\right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} D_i U_i
\]

\[\begin{align*}
\therefore a \\
+ \left(\frac{1}{n} \sum_{i \in I} D_i^2\right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} D_i (g_0(X_i) - \hat{g}_0(X_i))
\end{align*}\]

\[\begin{align*}
\therefore b
\end{align*}\]

- This looks scary, so let’s take it one term at a time
- \(\sqrt{n}(\hat{\theta}_0 - \theta_0)\) represents our scaled estimation error
Regularization Bias

\[
\sqrt{n}(\hat{\theta}_0 - \theta_0) = \left( \frac{1}{n} \sum_{i \in I} D_i^2 \right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} D_i U_i
\]

\[
\equiv a
\]

\[
+ \left( \frac{1}{n} \sum_{i \in I} D_i^2 \right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} D_i (g_0(X_i) - \hat{g}_0(X_i))
\]

\[
\equiv b
\]

- This looks scary, so let’s take it one term at a time
- \(\sqrt{n}(\hat{\theta}_0 - \theta_0)\) represents our scaled estimation error
- If we want consistency, we want our estimation error to go to zero
Regularization Bias

$$\sqrt{n}(\hat{\theta}_0 - \theta_0) = \left( \frac{1}{n} \sum_{i \in I} D_i^2 \right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} D_i U_i$$

$$:= a$$

$$+ \left( \frac{1}{n} \sum_{i \in I} D_i^2 \right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} D_i (g_0(X_i) - \hat{g}_0(X_i))$$

$$:= b$$

- This looks scary, so let’s take it one term at a time
- $\sqrt{n}(\hat{\theta}_0 - \theta_0)$ represents our scaled estimation error
- If we want consistency, we want this term to go to zero
- $a \rightsquigarrow N(0, \bar{\Sigma})$. Great!
Regularization Bias

\[
\sqrt{n}(\hat{\theta}_0 - \theta_0) = \left(\frac{1}{n} \sum_{i \in I} D_i^2\right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} D_i U_i := a \\
+ \left(\frac{1}{n} \sum_{i \in I} D_i^2\right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} D_i (g_0(X_i) - \hat{g}_0(X_i)) := b
\]

- \( b \) is the sum of terms that do not have mean zero divided by \( \sqrt{n} \)
Regularization Bias

\[
\sqrt{n} (\hat{\theta}_0 - \theta_0) = \left( \frac{1}{n} \sum_{i \in I} D_i^2 \right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} D_i U_i
\]

\[
\equiv a
\]

\[
+ \left( \frac{1}{n} \sum_{i \in I} D_i^2 \right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} D_i (g_0(X_i) - \hat{g}_0(X_i))
\]

\[
\equiv b
\]

- \(b\) is the sum of terms that do not have mean zero divided by \(\sqrt{n}\)
- Specifically, \(g_0(X_i) - \hat{g}_0(X_i)\) will not have mean zero because \(\hat{g}_0\) is biased
Regularization Bias

\[
\sqrt{n}(\hat{\theta}_0 - \theta_0) = \left( \frac{1}{n} \sum_{i \in I} D_i^2 \right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} D_i U_i
\]

\[::=a\]

\[+ \left( \frac{1}{n} \sum_{i \in I} D_i^2 \right)^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} D_i (g_0(X_i) - \hat{g}_0(X_i))
\]

\[::=b\]

- \(b\) is the sum of terms that do not have mean zero divided by \(\sqrt{n}\)
- Specifically, \(g_0(X_i) - \hat{g}_0(X_i)\) will not have mean zero because \(\hat{g}_0\) is biased
- \(b\) will approach 0, but too slowly for our estimator to be \(\sqrt{n}\)-consistent!
Causal Inference with ML using Orthogonalization

To overcome this regularization bias, let’s use orthogonalization. Instead of fitting one ML model, we fit two:

1. Estimate
   \[ D = \hat{m}_0(X) + \hat{V}, \]
   our treatment model

2. Estimate
   \[ Y = D \hat{\theta}_0 + \hat{g}_0(X) + \hat{U}, \]
   as we do in the naive approach, our outcome model

3. Regress
   \[ Y - \hat{g}_0(X) \]
   on \[ \hat{V} \]
   The resulting \[ \hat{\theta}_0 \] ("theta-naught-check") is free of regularization bias!

We can call this a “partialling-out” approach because we have partialled out the associations between \[ X \] and \[ D \] and between \[ Y \] and \[ X \] (conditional on \[ D \]).
Causal Inference with ML using Orthogonalization

To overcome this regularization bias, let’s use **orthogonalization**
To overcome this regularization bias, let’s use **orthogonalization**

Instead of fitting one ML model, we fit two:

1. Estimate $D = \hat{m}_0(X) + \hat{V}$, our treatment model
2. Estimate $Y = D \hat{\theta}_0 + \hat{g}_0(X) + \hat{U}$ as we do in the naive approach, our outcome model
3. Regress $Y - \hat{g}_0(X)$ on $\hat{V}$

The resulting $\hat{\theta}_0$ ("theta-naught-check") is free of regularization bias!

We can call this a "partialling-out" approach because we have partialled out the associations between $X$ and $D$ and between $Y$ and $X$ (conditional on $D$)
To overcome this regularization bias, let’s use **orthogonalization**.

Instead of fitting one ML model, we fit two:

1. Estimate $D = \hat{m}_0(X) + \hat{V}$, our treatment model.
2. Estimate $Y = D \hat{\theta}_0 + \hat{g}_0(X) + \hat{U}$ as we do in the naive approach, our outcome model.
3. Regress $Y - \hat{g}_0(X)$ on $\hat{V}$.

The resulting $\hat{\theta}_0$ ("theta-naught-check") is free of regularization bias!

We can call this a “partialling-out” approach because we have partialled out the associations between $X$ and $D$, and between $Y$ and $X$ (conditional on $D$).
Causal Inference with ML using Orthogonalization

To overcome this regularization bias, let’s use **orthogonalization**

Instead of fitting one ML model, we fit two:

1. Estimate $D = \hat{m}_0(X) + \hat{V}$, our treatment model
2. Estimate $Y = D\hat{\theta}_0 + \hat{g}_0(X) + \hat{U}$ as we do in the naive approach, our outcome model
Causal Inference with ML using Orthogonalization

To overcome this regularization bias, let’s use **orthogonalization**

Instead of fitting one ML model, we fit two:

1. Estimate $D = \hat{m}_0(X) + \hat{V}$, our treatment model
2. Estimate $Y = D\hat{\theta}_0 + \hat{g}_0(X) + \hat{U}$ as we do in the naive approach, our outcome model
3. Regress $Y - \hat{g}_0(X)$ on $\hat{V}$
Causal Inference with ML using Orthogonalization

- To overcome this regularization bias, let’s use **orthogonalization**
- Instead of fitting one ML model, we fit two:
  1. Estimate $D = \hat{m}_0(X) + \hat{V}$, our treatment model
  2. Estimate $Y = D\hat{\theta}_0 + \hat{g}_0(X) + \hat{U}$ as we do in the naive approach, our outcome model
  3. Regress $Y - \hat{g}_0(X)$ on $\hat{V}$
- The resulting $\check{\theta}_0$ ("theta-naught-check") is free of regularization bias!
Causal Inference with ML using Orthogonalization

To overcome this regularization bias, let’s use **orthogonalization**

Instead of fitting one ML model, we fit two:

1. Estimate \( D = \hat{m}_0(X) + \hat{V} \), our treatment model
2. Estimate \( Y = D\hat{\theta}_0 + \hat{g}_0(X) + \hat{U} \) as we do in the naive approach, our outcome model
3. Regress \( Y - \hat{g}_0(X) \) on \( \hat{V} \)

The resulting \( \hat{\theta}_0 \) ("theta-naught-check") is free of regularization bias!

We can call this a “partialling-out” approach because we have partialled out the associations between \( X \) and \( D \) and between \( Y \) and \( X \) (conditional on \( D \))
Causal Inference with ML using Orthogonalization

In notation:

\[
\hat{\theta}_0 = \left( \frac{1}{n} \sum_{i \in l} \hat{V}_i D_i \right)^{-1} \frac{1}{n} \sum_{i \in l} \hat{V}(Y_i - \hat{g}_0(X_i))
\]
Causal Inference with ML using Orthogonalization

- In notation:

\[
\theta_0 = \left( \frac{1}{n} \sum_{i \in I} \hat{V}_i D_i \right)^{-1} \frac{1}{n} \sum_{i \in I} \hat{V}(Y_i - \hat{g}_0(X_i))
\]

- Look familiar?
Causal Inference with ML using Orthogonalization

- In notation:

$$\hat{\theta}_0 = \left( \frac{1}{n} \sum_{i \in I} \hat{V}_i D_i \right)^{-1} \frac{1}{n} \sum_{i \in I} \hat{V} (Y_i - \hat{g}_0(X_i))$$

- Look familiar?

- What about now?

$$\hat{\beta}_{IV} = (Z' D)^{-1} Z' y$$
Causal Inference with ML using Orthogonalization

- In notation:
  \[
  \hat{\theta}_0 = \left( \frac{1}{n} \sum_{i \in I} \hat{\mathbf{V}}_i \mathbf{D}_i \right)^{-1} \frac{1}{n} \sum_{i \in I} \hat{\mathbf{V}} (Y_i - \hat{g}_0(X_i))
  \]

- Look familiar?

- What about now?

  \[
  \hat{\beta}_{IV} = (\mathbf{Z}' \mathbf{D})^{-1} \mathbf{Z}' \mathbf{y}
  \]

- It’s very similar to our standard linear instrumental variable estimator, two-stage least squares!
How Orthogonalization De-biases

Remember $b$ from the scaled estimation error equation? Now we have

$$b^* = \left( E \left[ V^2 \right] \right) - \sum_{i \in I} \left( \hat{m}_0(X_i) - m_0(X_i) \right)$$

Because this term is based on the product of two estimation errors, it vanishes more quickly.

If $\hat{g}_0$ and $\hat{m}_0$ are each $n^{1/4}$-consistent, $\hat{\theta}_0$ will be $\sqrt{n}$-consistent.

To see why, just note that $n^{1/4} \times n^{1/4} = n^{1/2}$.
How Orthogonalization De-biases

Remember \( b \) from the scaled estimation error equation? Now we have \( b^* \):
Remember $b$ from the scaled estimation error equation? Now we have $b^*$:

$$b^* = (E[V^2])^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} (\hat{m}_0(X_i) - m_0(X_i))(\hat{g}_0(X_i) - g_0(X_i))$$

- $\hat{m}_0$ estimation error
- $\hat{g}_0$ estimation error
How Orthogonalization De-biases

- Remember $b$ from the scaled estimation error equation? Now we have $b^*$:

$$b^* = (E[V^2])^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} (\hat{m}_0(X_i) - m_0(X_i)) (\hat{g}_0(X_i) - g_0(X_i))$$

- Because this term is based on the product of two estimation errors, it vanishes more quickly.
How Orthogonalization De-biases

- Remember $b$ from the scaled estimation error equation? Now we have $b^*$:

$$b^* = (E[V^2])^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} (\hat{m}_0(X_i) - m_0(X_i)) (\hat{g}_0(X_i) - g_0(X_i))$$

- Because this term is based on the product of two estimation errors, it vanishes more quickly.

- If $\hat{g}_0$ and $\hat{m}_0$ are each $n^{\frac{1}{4}}$-consistent, $\hat{\theta}_0$ will be $\sqrt{n}$-consistent.
Remember $b$ from the scaled estimation error equation? Now we have $b^*$:

$$b^* = (E[V^2])^{-1} \frac{1}{\sqrt{n}} \sum_{i \in I} (\hat{m}_0(X_i) - m_0(X_i)) (\hat{g}_0(X_i) - g_0(X_i))$$

Because this term is based on the product of two estimation errors, it vanishes more quickly.

If $\hat{g}_0$ and $\hat{m}_0$ are each $n^{1/4}$-consistent, $\hat{\theta}_0$ will be $\sqrt{n}$-consistent.

To see why, just note that $n^{1/4} \times n^{1/4} = n^{1/2}$.
A Quick Detour

Chernozhukov et al. use a second partialling-out estimator for partially linear models as well. This estimator is very similar to Robinson's partialling-out estimator, which is in turn very similar the Frisch–Waugh–Lovell partialling-out estimator. If you've taken an introductory statistics course, you've probably learned about the Frisch–Waugh–Lovell theorem. But even if you haven't (or don't remember!), reviewing Frisch–Waugh–Lovell theorem and Robinson can help us build intuition for how DML works.
A Quick Detour

- Chernozhukov et al. use a second partialling-out estimator for partially linear models as well
Chernozhukov et al. use a second partialling-out estimator for partially linear models as well.

This estimator is very similar to Robinson’s partialling-out estimator, which is in turn very similar to the Frisch–Waugh–Lovell partialling-out estimator.
A Quick Detour

- Chernozhukov et al. use a second partialling-out estimator for partially linear models as well.
- This estimator is very similar to Robinson’s partialling-out estimator, which is in turn very similar to the Frisch–Waugh–Lovell partialling-out estimator.
- If you’ve taken an introductory statistics course, you’ve probably learned about the Frisch–Waugh–Lovell theorem.
Chernozhukov et al. use a second partialling-out estimator for partially linear models as well. This estimator is very similar to Robinson’s partialling-out estimator, which is in turn very similar the Frisch–Waugh–Lovell partialling-out estimator. If you’ve taken an introductory statistics course, you’ve probably learned about the Frisch–Waugh–Lovell theorem. But even if you haven’t (or don’t remember!), reviewing Frisch–Waugh–Lovell theorem and Robinson can help us build intuition for how DML works.
The Frisch–Waugh–Lovell Theorem

Let's say we want to estimate the following model using OLS:

\[ Y = \beta_0 + \beta_1 D + \beta_2 X + U \]

The Frisch–Waugh–Lovell Theorem shows us that we can recover the OLS estimate of \( \beta_1 \) using a residuals-on-residuals OLS regression:

1. Regress \( D \) on \( X \) using OLS
   - Let \( \hat{D} \) be the predicted values of \( D \) and let the residuals \( \hat{V} = D - \hat{D} \)
2. Regress \( Y \) on \( X \) using OLS
   - Let \( \hat{Y} \) be the predicted values of \( Y \) and let the residuals \( \hat{W} = Y - \hat{Y} \)
3. Regress \( \hat{W} \) on \( \hat{V} \) using OLS
   - The estimated coefficient on \( \hat{V} \) will be the same as the estimated coefficient \( \hat{\beta}_1 \) from regressing \( Y \) on \( D \) and \( X \) using OLS!
The Frisch–Waugh–Lovell Theorem

Let’s say we want to estimate the following model using OLS:

\[ Y = \beta_0 + \beta_1 D + \beta_2 X + U \]

The Frisch–Waugh–Lovell Theorem shows us that we can recover the OLS estimate of \( \beta_1 \) using a residuals-on-residuals OLS regression:

1. Regress \( D \) on \( X \) using OLS
   
   Let \( \hat{D} \) be the predicted values of \( D \) and let the residuals \( \hat{V} = D - \hat{D} \)

2. Regress \( Y \) on \( X \) using OLS
   
   Let \( \hat{Y} \) be the predicted values of \( Y \) and let the residuals \( \hat{W} = Y - \hat{Y} \)

3. Regress \( \hat{W} \) on \( \hat{V} \) using OLS
   
   The estimated coefficient on \( \hat{V} \) will be the same as the estimated coefficient \( \hat{\beta}_1 \) from regressing \( Y \) on \( D \) and \( X \) using OLS!
The Frisch–Waugh–Lovell Theorem

- Let’s say we want to estimate the following model using OLS:
  \[ Y = \beta_0 + \beta_1 D + \beta_2 X + U \]
The Frisch–Waugh–Lovell Theorem

- Let’s say we want to estimate the following model using OLS:
  
  \[ Y = \beta_0 + \beta_1 D + \beta_2 X + U \]

- The Frisch–Waugh–Lovell Theorem shows us that we can recover the OLS estimate of \( \beta_1 \) using a residuals-on-residuals OLS regression:
  
  1. Regress \( D \) on \( X \) using OLS
  2. Let \( \hat{D} \) be the predicted values of \( D \) and let the residuals \( \hat{V} = D - \hat{D} \)
  3. Regress \( Y \) on \( X \) using OLS
  4. Let \( \hat{Y} \) be the predicted values of \( Y \) and let the residuals \( \hat{W} = Y - \hat{Y} \)
  5. Regress \( \hat{W} \) on \( \hat{V} \) using OLS

  The estimated coefficient on \( \hat{V} \) will be the same as the estimated coefficient \( \hat{\beta}_1 \) from regressing \( Y \) on \( D \) and \( X \) using OLS!
The Frisch–Waugh–Lovell Theorem

Let’s say we want to estimate the following model using OLS:

\[ Y = \beta_0 + \beta_1 D + \beta_2 X + U \]

The Frisch–Waugh–Lovell Theorem shows us that we can recover the OLS estimate of \( \beta_1 \) using a residuals-on-residuals OLS regression:

1. Regress \( D \) on \( X \) using OLS
   - Let \( \hat{D} \) be the predicted values of \( D \) and let the residuals \( \hat{V} = D - \hat{D} \)
The Frisch–Waugh–Lovell Theorem

Let’s say we want to estimate the following model using OLS:

\[ Y = \beta_0 + \beta_1 D + \beta_2 X + U \]

The Frisch–Waugh–Lovell Theorem shows us that we can recover the OLS estimate of \( \beta_1 \) using a residuals-on-residuals OLS regression:

1. Regress \( D \) on \( X \) using OLS
   - Let \( \hat{D} \) be the predicted values of \( D \) and let the residuals
     \[ \hat{V} = D - \hat{D} \]

2. Regress \( Y \) on \( X \) using OLS
   - Let \( \hat{Y} \) be the predicted values of \( Y \) and let the residuals
     \[ \hat{W} = Y - \hat{Y} \]
The Frisch–Waugh–Lovell Theorem

Let’s say we want to estimate the following model using OLS:

\[ Y = \beta_0 + \beta_1 D + \beta_2 X + U \]

The Frisch–Waugh–Lovell Theorem shows us that we can recover the OLS estimate of \( \beta_1 \) using a residuals-on-residuals OLS regression:

1. Regress \( D \) on \( X \) using OLS
   - Let \( \hat{D} \) be the predicted values of \( D \) and let the residuals \( \hat{V} = D - \hat{D} \)
2. Regress \( Y \) on \( \hat{D} \) using OLS
   - Let \( \hat{Y} \) be the predicted values of \( Y \) and let the residuals \( \hat{W} = Y - \hat{Y} \)
3. Regress \( \hat{W} \) on \( \hat{V} \) using OLS

The estimated coefficient on \( \hat{V} \) will be the same as the estimated coefficient \( \hat{\beta}_1 \) from regressing \( Y \) on \( D \) and \( X \) using OLS!
The Frisch–Waugh–Lovell Theorem

- Let’s say we want to estimate the following model using OLS:
  \[ Y = \beta_0 + \beta_1 D + \beta_2 X + U \]

- The Frisch–Waugh–Lovell Theorem shows us that we can recover the OLS estimate of \( \beta_1 \) using a residuals-on-residuals OLS regression:
  1. Regress \( D \) on \( X \) using OLS
     - Let \( \hat{D} \) be the predicted values of \( D \) and let the residuals \( \hat{V} = D - \hat{D} \)
  2. Regress \( Y \) on \( X \) using OLS
     - Let \( \hat{Y} \) be the predicted values of \( Y \) and let the residuals \( \hat{W} = Y - \hat{Y} \)
  3. Regress \( \hat{W} \) on \( \hat{V} \) using OLS

- The estimated coefficient on \( \hat{V} \) will be the same as the estimated coefficient \( \hat{\beta}_1 \) from regressing \( Y \) on \( D \) and \( X \) using OLS!
Robinson

The Frisch–Waugh–Lovell procedure:

1. Linear regression of $D$ on $X$
2. Linear regression of $Y$ on $X$
3. Linear regression of the residuals from 2 on the residuals from 1

Robinson's innovation: let's replace the linear regressions from 1 and 2 with some non-parametric regression

Robinson's procedure:

1. Kernel regression of $D$ on $X$
2. Kernel regression of $Y$ on $X$
3. Linear regression of the residuals from 2 on the residuals from 1
The Frisch–Waugh–Lovell procedure:

1. Linear regression of $D$ on $X$
2. Linear regression of $Y$ on $X$
3. Linear regression of the residuals from 2 on the residuals from 1

Robinson's innovation: let's replace the linear regressions from 1 and 2 with some non-parametric regression.
Robinson

The Frisch–Waugh–Lovell procedure:

1. **Linear** regression of $D$ on $X$

2. Linear regression of $Y$ on $X$

3. Linear regression of the residuals from 2 on the residuals from 1

Robinson's innovation: let's replace the linear regressions from 1 and 2 with some non-parametric regression.
The Frisch–Waugh–Lovell procedure:

1. Linear regression of $D$ on $X$
2. Linear regression of $Y$ on $X$
Robinson

The Frisch–Waugh–Lovell procedure:

1. **Linear** regression of $D$ on $X$
2. **Linear** regression of $Y$ on $X$
3. **Linear** regression of the residuals from 2 on the residuals from 1
The Frisch–Waugh–Lovell procedure:

1. **Linear** regression of \( D \) on \( X \)
2. **Linear** regression of \( Y \) on \( X \)
3. **Linear** regression of the residuals from 2 on the residuals from 1

Robinson's innovation: let's replace the **linear** regressions from 1 and 2 with some **non-parametric** regression
The Frisch–Waugh–Lovell procedure:

1. Linear regression of $D$ on $X$
2. Linear regression of $Y$ on $X$
3. Linear regression of the residuals from 2 on the residuals from 1

Robinson's innovation: let's replace the linear regressions from 1 and 2 with some non-parametric regression

Robinson's procedure:
The Frisch–Waugh–Lovell procedure:

1. **Linear** regression of $D$ on $X$
2. **Linear** regression of $Y$ on $X$
3. **Linear** regression of the residuals from 2 on the residuals from 1

Robinson’s innovation: let’s replace the linear regressions from 1 and 2 with some non-parametric regression

Robinson’s procedure:

1. **Kernel** regression of $D$ on $X$
The *Frisch–Waugh–Lovell* procedure:

1. **Linear** regression of $D$ on $X$
2. **Linear** regression of $Y$ on $X$
3. **Linear** regression of the residuals from 2 on the residuals from 1

Robinson’s innovation: let’s replace the linear regressions from 1 and 2 with some *non-parametric* regression

**Robinson’s** procedure:

1. **Kernel** regression of $D$ on $X$
2. **Kernel** regression of $Y$ on $X
The Frisch–Waugh–Lovell procedure:

1. Linear regression of $D$ on $X$
2. Linear regression of $Y$ on $X$
3. Linear regression of the residuals from 2 on the residuals from 1

Robinson’s innovation: let’s replace the linear regressions from 1 and 2 with some non-parametric regression

Robinson’s procedure:

1. Kernel regression of $D$ on $X$
2. Kernel regression of $Y$ on $X$
3. Linear regression of the residuals from 2 on the residuals from 1
Another Way to Orthogonalize

Estimate $D = \hat{m}(X) + \hat{V}$

Estimate $Y = \ell_0(X) + \hat{U}$

Note the absence of $D$ and the switch from $g_0(\cdot)$ to $\ell_0(\cdot)$, which is essentially $E[Y|X]$

Regress $\hat{U}$ on $\hat{V}$ using OLS for an estimate $\hat{\theta}_0$
Another Way to Orthogonalize

- DML using residuals-on-residuals regression:

\[ \hat{D} = \hat{\beta}_0 (X) + \hat{\beta}_1 (X) \]

\[ \hat{Y} = \hat{\beta}_0 (X) + \hat{\beta}_1 (X) \]

Note the absence of \( D \) and the switch from \( g_0 (\cdot) \) to \( \ell_0 (\cdot) \), which is essentially \( E[Y | X] \).

Regress \( \hat{U} \) on \( \hat{V} \) using OLS for an estimate \( \hat{\theta}_0 \).
Another Way to Orthogonalize

- DML using residuals-on-residuals regression:
  1. Estimate $D = \hat{m}_0(X) + \hat{V}$
Another Way to Orthogonalize

DML using residuals-on-residuals regression:

1. Estimate $D = \hat{m}_0(X) + \hat{V}$
2. Estimate $Y = \hat{\ell}_0(X) + \hat{U}$

- Note the absence of $D$ and the switch from $g_0(\cdot)$ to $\ell_0(\cdot)$, which is essentially $E[Y \mid X]$
Another Way to Orthogonalize

DML using residuals-on-residuals regression:

1. Estimate $D = \hat{m}_0(X) + \hat{V}$
2. Estimate $Y = \hat{\ell}_0(X) + \hat{U}$
   - Note the absence of $D$ and the switch from $g_0(\cdot)$ to $\ell_0(\cdot)$, which is essentially $\mathbb{E}[Y \mid X]$
3. Regress $\hat{U}$ on $\hat{V}$ using OLS for an estimate $\hat{\theta}_0$
Another Way to Orthogonalize

1. Predict $D$ with $X$ using kernel regression
2. Predict $Y$ with $X$ using kernel regression
3. Linear regression of the residuals from 2 on the residuals from 1

DML residuals-on-residuals

1. Predict $D$ with $X$ using any $n_1$-consistent ML model
2. Predict $Y$ with $X$ using any $n_1$-consistent ML model
3. Linear regression of the residuals from 2 on the residuals from 1
Another Way to Orthogonalize

- **Robinson’s** procedure:
  1. Predict $D$ with $X$ using kernel regression
  2. Predict $Y$ with $X$ using kernel regression
  3. Linear regression of the residuals from 2 on the residuals from 1

- **DML residuals-on-residuals** procedure:
  1. Predict $D$ with $X$ using any $n_1^4$-consistent ML model
  2. Predict $Y$ with $X$ using any $n_1^4$-consistent ML model
  3. Linear regression of the residuals from 2 on the residuals from 1
Another Way to Orthogonalize

- **Robinson’s** procedure:
  1. Predict $D$ with $X$ using **kernel regression**
Another Way to Orthogonalize

- **Robinson’s procedure:**
  1. Predict $D$ with $X$ using *kernel regression*
  2. Predict $Y$ with $X$ using *kernel regression*
Another Way to Orthogonalize

- **Robinson’s** procedure:
  1. Predict $D$ with $X$ using **kernel regression**
  2. Predict $Y$ with $X$ using **kernel regression**
  3. **Linear** regression of the residuals from 2 on the residuals from 1

- **DML residuals-on-residuals** procedure:
Another Way to Orthogonalize

- **Robinson’s** procedure:
  1. Predict $D$ with $X$ using **kernel regression**
  2. Predict $Y$ with $X$ using **kernel regression**
  3. **Linear** regression of the residuals from 2 on the residuals from 1

- **DML residuals-on-residuals** procedure:
  1. Predict $D$ with $X$ using **any $n^{1/4}$-consistent ML model**
Another Way to Orthogonalize

- **Robinson’s procedure:**
  1. Predict $D$ with $X$ using *kernel regression*
  2. Predict $Y$ with $X$ using *kernel regression*
  3. *Linear* regression of the residuals from 2 on the residuals from 1

- **DML residuals-on-residuals procedure:**
  1. Predict $D$ with $X$ using *any $n^{1/4}$-consistent ML model*
  2. Predict $Y$ with $X$ using *any $n^{1/4}$-consistent ML model*
Another Way to Orthogonalize

- **Robinson’s procedure:**
  1. Predict $D$ with $X$ using **kernel regression**
  2. Predict $Y$ with $X$ using **kernel regression**
  3. **Linear** regression of the residuals from 2 on the residuals from 1

- **DML residuals-on-residuals procedure:**
  1. Predict $D$ with $X$ using **any $n^{1/4}$-consistent ML model**
  2. Predict $Y$ with $X$ using **any $n^{1/4}$-consistent ML model**
  3. **Linear** regression of the residuals from 2 on the residuals from 1
Where We Are and Where We’re Going

We saw that can eliminate regularization bias using orthogonalization.

Now we're going to show how we can eliminate bias from overfitting using sample-splitting and cross-fitting.
Where We Are and Where We’re Going

- We saw that can eliminate regularization bias using orthogonalization
Where We Are and Where We’re Going

- We saw that can eliminate regularization bias using orthogonalization
- Now we’re going to show how we can eliminate bias from overfitting using sample-splitting and cross-fitting
Bias from Overfitting
Bias from Overfitting

ML algorithms sometimes overfit our data due to their flexibility, and this can lead to bias and slower convergence.
Bias from Overfitting

- ML algorithms sometimes overfit our data due to their flexibility, and this can lead to bias and slower convergence.
- Let’s say we estimate $\hat{\theta}_0$ using orthogonalization but without sample-splitting.
Bias from Overfitting

- ML algorithms sometimes overfit our data due to their flexibility, and this can lead to bias and slower convergence.
- Let’s say we estimate $\hat{\theta}_0$ using orthogonalization but without sample-splitting.
- That is, we fit our machine learning models and estimate our target parameter $\hat{\theta}_0$ on the same set of observations.
Bias from Overfitting

- ML algorithms sometimes overfit our data due to their flexibility, and this can lead to bias and slower convergence.
- Let’s say we estimate $\hat{\theta}_0$ using orthogonalization but without sample-splitting.
- That is, we fit our machine learning models and estimate our target parameter $\hat{\theta}_0$ on the same set of observations.
- Our scaled estimation error $\sqrt{n}(\hat{\theta}_0 - \theta_0) = a^* + b^* + c^*$.
Bias from Overfitting

- ML algorithms sometimes overfit our data due to their flexibility, and this can lead to bias and slower convergence.
- Let’s say we estimate \( \hat{\theta}_0 \) using orthogonalization but without sample-splitting.
- That is, we fit our machine learning models and estimate our target parameter \( \hat{\theta}_0 \) on the same set of observations.
- Our scaled estimation error \( \sqrt{n}(\hat{\theta}_0 - \theta_0) = a^* + b^* + c^* \).
- We looked at \( b^* \) before, and we don’t have to worry about \( a^* \).
Bias from Overfitting

But $c^*$ contains terms like this:

$$\frac{1}{\sqrt{n}} \sum_{i \in I} V_i(\hat{g}_0(X_i) - g_0(X_i))$$
Bias from Overfitting

- But $c^*$ contains terms like this:

$$
\frac{1}{\sqrt{n}} \sum_{i \in I} V_i (\hat{g}_0(X_i) - g_0(X_i))
$$

- $V$: the error term \textbf{not} the estimated residuals) from $D = m_0(X) + V$
Bias from Overfitting

- But $c^\ast$ contains terms like this:

$$\frac{1}{\sqrt{n}} \sum_{i \in I} V_i(\hat{g}_0(X_i) - g_0(X_i))$$

- $V$: the error term (not the estimated residuals) from

$$D = m_0(X) + V$$

- $\hat{g}_0(X_i) - g_0(X_i)$: estimation error in $\hat{g}_0$ from

$$Y = D\theta_0 + \hat{g}_0(X) + \hat{U}$$
Bias from Overfitting

1. But $c^*$ contains terms like this:

$$
\frac{1}{\sqrt{n}} \sum_{i \in I} V_i(\hat{g}_0(X_i) - g_0(X_i))
$$

2. $V$: the error term (not the estimated residuals) from $D = m_0(X) + V$

3. $\hat{g}_0(X_i) - g_0(X_i)$: estimation error in $\hat{g}_0$ from $Y = D\hat{\theta}_0 + \hat{g}_0(X) + \hat{U}$

4. What happens if we estimate $\hat{\theta}_0$ with the same set of observations we used to fit $\hat{m}_0(X)$ and $\hat{g}_0(X)$?
Bias from Overfitting

But $c^*$ contains terms like this:

$$\frac{1}{\sqrt{n}} \sum_{i \in I} V_i(\hat{g}_0(X_i) - g_0(X_i))$$
Bias from Overfitting

- But $c^*$ contains terms like this:
  \[
  \frac{1}{\sqrt{n}} \sum_{i \in I} V_i (\hat{g}_0(X_i) - g_0(X_i))
  \]

- Overfitting: modeling the noise too closely
Bias from Overfitting

- But $c^*$ contains terms like this:
  \[ \frac{1}{\sqrt{n}} \sum_{i \in I} V_i (\hat{g}_0(X_i) - g_0(X_i)) \]

- Overfitting: modeling the noise too closely
- When $\hat{g}_0$ is overfit, it will pick up on some of the noise $U$ from the outcome model
Bias from Overfitting

- But $c^*$ contains terms like this:
  \[
  \frac{1}{\sqrt{n}} \sum_{i \in I} V_i (\hat{g}_0(X_i) - g_0(X_i))
  \]

- Overfitting: modeling the noise too closely
- When $\hat{g}_0$ is overfit, it will pick up on some of the noise $U$ from the outcome model
- If our noise terms $V$ and $U$ are associated, estimation error in $\hat{g}_0$ from overfitting might be associated with $V$
Bias from Overfitting

- But $c^*$ contains terms like this:
  \[
  \frac{1}{\sqrt{n}} \sum_{i \in I} V_i(\hat{g}_0(X_i) - g_0(X_i))
  \]

- Overfitting: modeling the noise too closely
- When $\hat{g}_0$ is overfit, it will pick up on some of the noise $U$ from the outcome model
- If our noise terms $V$ and $U$ are associated, estimation error in $\hat{g}_0$ from overfitting might be associated with $V$
- **Note:** $\hat{g}_0$ and $V$ might also be associated if we have any unmeasured confounding, but we’re assuming that away
How to Avoid Bias from Overfitting

To break the association between $V$ and $\hat{g}_0$ and avoid bias from overfitting, we employ sample-splitting:

1. Randomly partition our data into two subsets.
2. Fit your ML models $\hat{g}_0$ and $\hat{m}_0$ on the first subset.
3. Estimate $\hat{\theta}_0$ in the second subset using the $\hat{g}_0$ and $\hat{m}_0$ functions we fit in the first subset.

Key:
We never estimate $\hat{\theta}_0$ using the same observations that we used to fit $\hat{g}_0$ and $\hat{m}_0$.
To break the association between $V$ and $\hat{g}_0$ and avoid bias from overfitting, we employ sample-splitting:
How to Avoid Bias from Overfitting

To break the association between \( V \) and \( \hat{g}_0 \) and avoid bias from overfitting, we employ \textbf{sample-splitting}:

1. Randomly partition our data into two subsets
How to Avoid Bias from Overfitting

To break the association between $V$ and $\hat{g}_0$ and avoid bias from overfitting, we employ **sample-splitting:**

1. Randomly partition our data into two subsets
2. Fit your ML models $\hat{g}_0$ and $\hat{m}_0$ on the **first** subset
How to Avoid Bias from Overfitting

To break the association between $V$ and $\hat{g}_0$ and avoid bias from overfitting, we employ **sample-splitting:**

1. Randomly partition our data into two subsets
2. Fit your ML models $\hat{g}_0$ and $\hat{m}_0$ on the **first** subset
3. Estimate $\hat{\theta}_0$ in the **second** subset using the $\hat{g}_0$ and $\hat{m}_0$ functions we fit in the first subset
How to Avoid Bias from Overfitting

- To break the association between $V$ and $\hat{g}_0$ and avoid bias from overfitting, we employ **sample-splitting:**
  1. Randomly partition our data into two subsets
  2. Fit your ML models $\hat{g}_0$ and $\hat{m}_0$ on the first subset
  3. Estimate $\hat{\theta}_0$ in the second subset using the $\hat{g}_0$ and $\hat{m}_0$ functions we fit in the first subset

**Key:** We never estimate $\hat{\theta}_0$ using the same observations that we used to fit $\hat{g}_0$ and $\hat{m}_0$
Cross-Fitting

The standard approach to sample-splitting will reduce efficiency and statistical power. We can avoid the loss of efficiency and power using cross-fitting:

1. Randomly partition your data into two subsets.
2. Fit two ML models $\hat{g}_0, \hat{m}_0$ and $\hat{g}_1, \hat{m}_1$ in the first subset.
3. Estimate $\hat{\theta}_0, \hat{\theta}_1$ in the second subset using the $\hat{g}_0, \hat{g}_1$ and $\hat{m}_0, \hat{m}_1$ functions we fit in the first subset.
4. Fit two ML models $\hat{g}_0, \hat{m}_0$ and $\hat{g}_2, \hat{m}_2$ in the second subset.
5. Estimate $\hat{\theta}_0, \hat{\theta}_2$ in the first subset using the $\hat{g}_0, \hat{g}_2$ and $\hat{m}_0, \hat{m}_2$ functions we fit in the second subset.
6. Average our two estimates $\hat{\theta}_0, \hat{\theta}_1$ and $\hat{\theta}_0, \hat{\theta}_2$ for our final estimate $\hat{\theta}_0$. 
Cross-Fitting

- The standard approach to sample-splitting will reduce efficiency and statistical power

1. Randomly partition your data into two subsets
2. Fit two ML models $\hat{g}_0, \hat{m}_0$ and $\hat{g}_1, \hat{m}_1$ in the first subset
3. Estimate $\hat{\theta}_0, \hat{\theta}_1$ in the second subset using the $\hat{g}_0, \hat{m}_0$ and $\hat{g}_1, \hat{m}_1$ functions we fit in the first subset
4. Fit two ML models $\hat{g}_0, \hat{m}_0$ and $\hat{g}_2, \hat{m}_2$ in the second subset
5. Estimate $\hat{\theta}_0, \hat{\theta}_2$ in the first subset using the $\hat{g}_0, \hat{m}_0$ and $\hat{g}_2, \hat{m}_2$ functions we fit in the second subset
6. Average our two estimates $\hat{\theta}_0, \hat{\theta}_1$ and $\hat{\theta}_0, \hat{\theta}_2$ for our final estimate $\hat{\theta}_0$
Cross-Fitting

- The standard approach to sample-splitting will reduce efficiency and statistical power.
- We can avoid the loss of efficiency and power using cross-fitting.
Cross-Fitting

- The standard approach to sample-splitting will reduce efficiency and statistical power
- We can avoid the loss of efficiency and power using **cross-fitting:**
  1. Randomly partition your data into two subsets
Cross-Fitting

- The standard approach to sample-splitting will reduce efficiency and statistical power
- We can avoid the loss of efficiency and power using **cross-fitting**:  
  1. Randomly partition your data into two subsets
  2. Fit two ML models \( \hat{g}_{0,1} \) and \( \hat{m}_{0,1} \) in the first subset
  3. Estimate \( \hat{\theta}_{0,1} \) in the second subset using the \( \hat{g}_{0,1} \) and \( \hat{m}_{0,1} \) functions we fit in the first subset
  4. Fit two ML models \( \hat{g}_{0,2} \) and \( \hat{m}_{0,2} \) in the second subset
  5. Estimate \( \hat{\theta}_{0,2} \) in the first subset using the \( \hat{g}_{0,2} \) and \( \hat{m}_{0,2} \) functions we fit in the second subset
  6. Average our two estimates \( \hat{\theta}_{0,1} \) and \( \hat{\theta}_{0,2} \) for our final estimate \( \hat{\theta}_0 \)
Cross-Fitting

- The standard approach to sample-splitting will reduce efficiency and statistical power
- We can avoid the loss of efficiency and power using cross-fitting:
  1. Randomly partition your data into two subsets
  2. Fit two ML models $\hat{g}_{0,1}$ and $\hat{m}_{0,1}$ in the first subset
  3. Estimate $\hat{\theta}_{0,1}$ in the second subset using the $\hat{g}_{0,1}$ and $\hat{m}_{0,1}$ functions we fit in the first subset
  4. Fit two ML models $\hat{g}_{0,2}$ and $\hat{m}_{0,2}$ in the second subset
  5. Estimate $\hat{\theta}_{0,2}$ in the first subset using the $\hat{g}_{0,2}$ and $\hat{m}_{0,2}$ functions we fit in the second subset
  6. Average our two estimates $\hat{\theta}_{0,1}$ and $\hat{\theta}_{0,2}$ for our final estimate $\hat{\theta}_{0}$
Cross-Fitting

- The standard approach to sample-splitting will reduce efficiency and statistical power
- We can avoid the loss of efficiency and power using cross-fitting:
  1. Randomly partition your data into two subsets
  2. Fit two ML models \( \hat{g}_{0,1} \) and \( \hat{m}_{0,1} \) in the first subset
  3. Estimate \( \hat{\theta}_{0,1} \) in the second subset using the \( \hat{g}_{0,1} \) and \( \hat{m}_{0,1} \) functions we fit in the first subset
  4. Fit two ML models \( \hat{g}_{0,2} \) and \( \hat{m}_{0,2} \) in the second subset
Cross-Fitting

- The standard approach to sample-splitting will reduce efficiency and statistical power.
- We can avoid the loss of efficiency and power using **cross-fitting**:
  1. Randomly partition your data into two subsets.
  2. Fit two ML models $\hat{g}_{0,1}$ and $\hat{m}_{0,1}$ in the **first** subset.
  3. Estimate $\hat{\theta}_{0,1}$ in the **second** subset using the $\hat{g}_{0,1}$ and $\hat{m}_{0,1}$ functions we fit in the first subset.
  4. Fit two ML models $\hat{g}_{0,2}$ and $\hat{m}_{0,2}$ in the **second** subset.
  5. Estimate $\hat{\theta}_{0,2}$ in the **first** subset using the $\hat{g}_{0,2}$ and $\hat{m}_{0,2}$ functions we fit in the second subset.
Cross-Fitting

- The standard approach to sample-splitting will reduce efficiency and statistical power.
- We can avoid the loss of efficiency and power using **cross-fitting**:

1. Randomly partition your data into two subsets.
2. Fit two ML models $\hat{g}_{0,1}$ and $\hat{m}_{0,1}$ in the **first** subset.
3. Estimate $\hat{\theta}_{0,1}$ in the **second** subset using the $\hat{g}_{0,1}$ and $\hat{m}_{0,1}$ functions we fit in the first subset.
4. Fit two ML models $\hat{g}_{0,2}$ and $\hat{m}_{0,2}$ in the **second** subset.
5. Estimate $\hat{\theta}_{0,2}$ in the **first** subset using the $\hat{g}_{0,2}$ and $\hat{m}_{0,2}$ functions we fit in the second subset.
6. Average our two estimates $\hat{\theta}_{0,1}$ and $\hat{\theta}_{0,2}$ for our final estimate $\hat{\theta}_0$. 

Where We Are and Where We’re Going

We now (hopefully) have a good intuition for how fitting two ML models allows us to remove bias and achieve faster convergence. Next, we’re going to look at how the authors formally define Neyman orthogonality and DML.
Where We Are and Where We’re Going

- We now (hopefully) have a good intuition for how fitting two ML models allows us to remove bias and achieve faster convergence.
Where We Are and Where We’re Going

- We now (hopefully) have a good intuition for how fitting two ML models allows us to remove bias and achieve faster convergence.

- Next we’re going to look at how the authors formally define Neyman orthogonality and DML.
Defining Neyman Orthogonality

- Remember: orthogonalize $D$ with respect to $X \rightarrow$ eliminate regularization bias
Defining Neyman Orthogonality

- Remember: orthogonalize $D$ with respect to $X \rightarrow$ eliminate regularization bias
- Now we formalize this “Neyman orthogonality” condition

Let $\eta_0 = (g_0, m_0), \eta = (g, m)$

$\eta_0$ is our nuisance parameter. We don’t really care what $g_0$ and $m_0$ are. We just want to use them to get good estimates of $\theta_0$. The exact form of our nuisance parameter isn’t a scientifically substantive quantity of interest.
Defining Neyman Orthogonality

- Remember: orthogonalize $D$ with respect to $X \rightarrow$ eliminate regularization bias
- Now we formalize this “Neyman orthogonality” condition
- Let

$$\eta_0 = (g_0, m_0), \eta = (g, m)$$
Defining Neyman Orthogonality

- Remember: orthogonalize $D$ with respect to $X \rightarrow$ eliminate regularization bias
- Now we formalize this “Neyman orthogonality” condition
- Let
  \[ \eta_0 = (g_0, m_0), \eta = (g, m) \]
- We have a new friend! Her name is $\eta_0$ (“eta-naught”)
Defining Neyman Orthogonality

- Remember: orthogonalize $D$ with respect to $X \rightarrow$ eliminate regularization bias
- Now we formalize this “Neyman orthogonality” condition
- Let
  \[ \eta_0 = (g_0, m_0), \eta = (g, m) \]

- We have a new friend! Her name is $\eta_0$ (“eta-naught”)
- $\eta_0$ is our nuisance parameter
Defining Neyman Orthogonality

- Remember: orthogonalize $D$ with respect to $X \rightarrow$ eliminate regularization bias
- Now we formalize this “Neyman orthogonality” condition
- Let
  \[ \eta_0 = (g_0, m_0), \eta = (g, m) \]

- We have a new friend! Her name is $\eta_0$ (“eta-naught”)
- $\eta_0$ is our *nuisance parameter*
  - We don’t really care what $g_0$ and $m_0$ are
Defining Neyman Orthogonality

1. Remember: orthogonalize $D$ with respect to $X \rightarrow$ eliminate regularization bias
2. Now we formalize this “Neyman orthogonality” condition
3. Let
   \[ \eta_0 = (g_0, m_0), \eta = (g, m) \]
4. We have a new friend! Her name is $\eta_0$ (“eta-naught”)
5. $\eta_0$ is our nuisance parameter
   - We don’t really care what $g_0$ and $m_0$ are
   - We just want to use them to get good estimates of $\theta_0$
Defining Neyman Orthogonality

- Remember: orthogonalize $D$ with respect to $X \rightarrow$ eliminate regularization bias
- Now we formalize this “Neyman orthogonality” condition
- Let
  $$\eta_0 = (g_0, m_0), \eta = (g, m)$$

- We have a new friend! Her name is $\eta_0$ (“eta-naught”)
- $\eta_0$ is our nuisance parameter
  - We don’t really care what $g_0$ and $m_0$ are
  - We just want to use them to get good estimates of $\theta_0$
  - The exact form of our nuisance parameter isn’t a scientifically substantive quantity of interest
Defining Neyman Orthogonality

Introduce the following score function $\psi(W;\theta,\eta_0) = (D - m_0(X) ) \ast V \times (Y - g_0(X) - (D - m_0(X)) \theta)$. 

Looks scary! Let's break it down:

- $\psi$: our score function "psi"
- $W$: our data
- Each underbraced term represents a noise term from our partially linear model.
Defining Neyman Orthogonality

- Introduce the following score function:

\[
\psi(W; \theta, \eta_0) = (D - m_0(X)) \times (Y - g_0(X) - (D - m_0(X))\theta)
\]
Defining Neyman Orthogonality

- Introduce the following score function:

\[ \psi(W; \theta, \eta_0) = (D - m_0(X)) \times (Y - g_0(X) - (D - m_0(X))\theta) \]

- Looks scary! Let's break it down
Defining Neyman Orthogonality

- Introduce the following score function:

\[ \psi(W; \theta, \eta_0) = (D - m_0(X)) \times (Y - g_0(X) - (D - m_0(X))\theta) \]

- Looks scary! Let's break it down

- \( \psi \): our score function “psi”
Defining Neyman Orthogonality

- Introduce the following score function:

\[ \psi(W; \theta, \eta_0) = \left( D - m_0(X) \right) \times \left( Y - g_0(X) - (D - m_0(X))\theta \right) \]

- Looks scary! Let's break it down
- \( \psi \): our score function “psi”
- \( W \): our data
Defining Neyman Orthogonality

- Introduce the following score function:

\[ \psi(W; \theta, \eta_0) = (D - m_0(X)) \times (Y - g_0(X) - (D - m_0(X))\theta) \]

- Looks scary! Let's break it down
  - \( \psi \): our score function “psi”
  - \( W \): our data
  - Each underbraced term represents a noise term from our partially linear model
Defining Neyman Orthogonality

Introduce the following moment condition:

$$\psi(W; \theta, \eta_0) = (D - m_0(X)) \times (Y - g_0(X) - (D - m_0(X))\theta) = 0$$

So we want our score function to = 0. Why?

$$Y = V_\theta + g_0(X) + U$$

$$V = (D - m_0(X))\theta_0 + g_0(X) + U$$

is our regressor

$$U = (Y - g_0(X) - (D - m_0(X))\theta)$$

is our error term
Defining Neyman Orthogonality

- Introduce the following *moment condition*:

\[
\psi(W; \theta, \eta_0) = (D - m_0(X)) \times (Y - g_0(X) - (D - m_0(X))\theta) = 0
\]
Defining Neyman Orthogonality

- Introduce the following *moment condition*:

  \[ \psi(W; \theta, \eta_0) = (D - m_0(X)) \times (Y - g_0(X) - (D - m_0(X))\theta) = 0 \]

- So we want our score function to = 0. Why?
Defining Neyman Orthogonality

- Introduce the following *moment condition*:
  \[
  \psi(W; \theta, \eta_0) = (D - m_0(X)) \times (Y - g_0(X) - (D - m_0(X))\theta) = 0
  \]
- So we want our score function to = 0. Why?
  \[
  Y = V\theta_0 + g_0(X) + U
  \]
  \[
  Y = (D - m_0(X))\theta_0 + g_0(X) + U
  \]
Defining Neyman Orthogonality

- Introduce the following *moment condition*:

\[ \psi(W; \theta, \eta_0) = (D - m_0(X)) \times (Y - g_0(X) - (D - m_0(X))\theta) = 0 \]

- So we want our score function to = 0. Why?

\[ Y = V\theta_0 + g_0(X) + U \]

\[ Y = (D - m_0(X))\theta_0 + g_0(X) + U \]

- \( V = (D - m_0(X)) \) is our regressor
Defining Neyman Orthogonality

- Introduce the following *moment condition*:
  \[
  \psi(W; \theta, \eta_0) = (D - m_0(X)) \times (Y - g_0(X) - (D - m_0(X))\theta) = 0
  \]

- So we want our score function to be 0. Why?

  \[
  Y = V\theta_0 + g_0(X) + U
  \]
  \[
  Y = (D - m_0(X))\theta_0 + g_0(X) + U
  \]

- \( V = (D - m_0(X)) \) is our regressor

- \( U = (Y - g_0(X) - (D - m_0(X))\theta) \) is our error term
Defining Neyman Orthogonality

- We’re saying we want our regressor $V$ and our error term $U$ to be orthogonal to one another\(^1\)

---

\(^1\)Moment conditions like this will look more familiar to people who know Generalized Method of Moments (GMM). My understanding is that while GMM is popular in economics, it’s less common in political science and sociology, which is why I explain what’s going on in a little more depth here.
Defining Neyman Orthogonality

- We’re saying we want our regressor $V$ and our error term $U$ to be orthogonal to one another\(^1\)
- Two vectors are orthogonal to one another when their dot product equals 0

---

\(^1\)Moment conditions like this will look more familiar to people who know Generalized Method of Moments (GMM). My understanding is that while GMM is popular in economics, it’s less common in political science and sociology, which is why I explain what’s going on in a little more depth here.
Defining Neyman Orthogonality

- We’re saying we want our regressor $V$ and our error term $U$ to be orthogonal to one another\(^1\)
- Two vectors are orthogonal to one another when their dot product equals 0
- This moment condition is very similar to saying we want our error to be uncorrelated with our regressor

---
\(^1\)Moment conditions like this will look more familiar to people who know Generalized Method of Moments (GMM). My understanding is that while GMM is popular in economics, it’s less common in political science and sociology, which is why I explain what’s going on in a little more depth here.
Defining Neyman Orthogonality

- We’re saying we want our regressor $V$ and our error term $U$ to be orthogonal to one another\(^1\)
- Two vectors are orthogonal to one another when their dot product equals 0
- This moment condition is very similar to saying we want our error to be uncorrelated with our regressor
- It is also very similar to (but slightly weaker than) the standard zero conditional mean assumption for OLS

\(^1\)Moment conditions like this will look more familiar to people who know Generalized Method of Moments (GMM). My understanding is that while GMM is popular in economics, it’s less common in political science and sociology, which is why I explain what’s going on in a little more depth here.
Now we can define Neyman orthogonality!

\[ \partial \eta \mathbb{E}[\psi(W; \theta_0, \eta_0)] = 0 \]

In words: the (Gateaux) derivative of our score function with respect to our nuisance parameter is 0. Recall that a derivative represents our instantaneous rate of change. Thus, when the derivative = 0, our score function is robust to small perturbations in \( \eta_0 \). It doesn't change much when \( \eta_0 \) moves around a little.
Now we can define Neyman orthogonality!

\[ \partial \eta E[\psi(W; \theta_0, \eta_0)] = 0 \]

In words: the (Gateaux) derivative of our score function with respect to our nuisance parameter is 0. Recall that a derivative represents our instantaneous rate of change. Thus, when the derivative = 0, our score function is robust to small perturbations in \( \eta_0 \). It doesn't change much when \( \eta_0 \) moves around a little.
Defining Neyman Orthogonality

Now we can define Neyman orthogonality!

\[ \partial_{\eta} E[\psi(W; \theta_0, \eta_0)][\eta - \eta_0] = 0 \]
Defining Neyman Orthogonality

- Now we can define Neyman orthogonality!

\[ \partial_\eta E[\psi(W; \theta_0, \eta_0)](\eta - \eta_0) = 0 \]

- In words: the (Gateaux) derivative of our score function with respect to our nuisance parameter is 0.
Defining Neyman Orthogonality

- Now we can define Neyman orthogonality!

\[ \partial_\eta E[\psi(W; \theta_0, \eta_0)][\eta - \eta_0] = 0 \]

- In words: the (Gateaux) derivative of our score function with respect to our nuisance parameter is 0.

- Recall that a derivative represents our instantaneous rate of change.
Defining Neyman Orthogonality

Now we can define Neyman orthogonality!

\[ \partial_{\eta} E[\psi(W; \theta_0, \eta_0)][\eta - \eta_0] = 0 \]

In words: the (Gateaux) derivative of our score function with respect to our nuisance parameter is 0.

Recall that a derivative represents our instantaneous rate of change.

Thus, when the derivative = 0, our score function is robust to small perturbations in \( \eta_0 \).
Defining Neyman Orthogonality

Now we can define Neyman orthogonality!

$$\partial_\eta E[\psi(W; \theta_0, \eta_0)][\eta - \eta_0] = 0$$

In words: the (Gateaux) derivative of our score function with respect to our nuisance parameter is 0.

Recall that a derivative represents our instantaneous rate of change.

Thus, when the derivative = 0, **our score function is robust to small perturbations in \( \eta_0 \)**.

It doesn’t change much when \( \eta_0 \) moves around a little.
Now we’re ready to formally define DML!
Take a $K$-fold random partition $(I_k)_{k=1}^K$ of observation indices $[N] = 1, \ldots, N$ such that the size of each fold $I_k$ is $n = N/K$. Also, for each $k \in [K] = 1, \ldots, K$, define $I_k^c := 1, \ldots, N \setminus I_k$.

- Create $K$ equally sized partitions
- $I_k^c$ is the **complement** of $I_k$: if we have 100 observations and $I_k$ is the set of observations 1–20, then $I_k^c$ is the set of observations 21–100
For each \( k \in [K] \), construct an ML estimator

\[
\hat{\eta}_{0,k} = \hat{\eta}_0((W_i)_{i \in I^c_k})
\]

- **Important:** The estimator we use for fold \( k \) was fit in \( I^c_k \)!
- This is the **sample splitting** we talked about earlier that removes bias from overfitting.
Defining DML

Construct the estimator $\tilde{\theta}_0$ ("theta-naught-tilde") as the solution to

$$\frac{1}{K} \sum_{k=1}^{K} E_{n,k}[\psi(W; \tilde{\theta}_0, \hat{\eta}_{0,k})] = 0$$

- Note that $\tilde{\theta}_0$ is not indexed by $k$, but the nuisance parameter $\hat{\eta}_{0,k}$ is.
- We’re finding the $\tilde{\theta}_0$ that minimizes the average of the scores across all folds, where the scores vary by fold due to $\hat{\eta}_{0,k}$.
- This is a slightly different\(^2\) version of the cross-fitting approach we talked about earlier that enables us to do sample splitting without loss of efficiency.

\(^2\)In particular, we are no longer taking the average of $k$ different estimates of $\tilde{\theta}_0$ but instead finding one estimate that minimizes the average of the $k$ different score functions. Chernozhukov et al. recommend this latter approach because it behaves better in smaller samples.
The ATE

Even under unconfoundedness, OLS does not (necessarily) give us the ATE. We set up the following moment condition for estimation of the ATE:

\[ \psi(W; \theta, \eta) := (g(1, X) - g(0, X)) + D(Y - g(1, X))m(X) - (1 - D)(Y - g(0, X)) \frac{1}{1 - m(X)} - \theta \]

Recall the score function has to = 0. So we're saying we want the three big terms in the middle = \( \theta \). Let's look at them more closely.
The ATE

- Even under unconfoundedness, OLS does not (necessarily) give us the ATE
The ATE

- Even under unconfoundedness, OLS does not (necessarily) give us the ATE
- We set up the following moment condition for estimation of the ATE:

\[ \psi(W; \theta, \eta) := (g(1, X) - g(0, X)) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)} - \theta \]

Recall the score function has to = 0
So we're saying we want the three big terms in the middle to = \( \theta \)
Let's look at them more closely
The ATE

- Even under unconfoundedness, OLS does not (necessarily) give us the ATE.
- We set up the following moment condition for estimation of the ATE:

\[
\psi(W; \theta, \eta) := (g(1, X) - g(0, X)) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)} - \theta
\]

- Recall the score function has to = 0.
The ATE

- Even under unconfoundedness, OLS does not (necessarily) give us the ATE
- We set up the following moment condition for estimation of the ATE:

\[
\psi(W; \theta, \eta) := (g(1, X) - g(0, X)) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)} - \theta
\]

- Recall the score function has to $= 0$
- So we’re saying we want the three big terms in the middle to $= \theta$
The ATE

- Even under unconfoundedness, OLS does not (necessarily) give us the ATE.
- We set up the following moment condition for estimation of the ATE:

\[
\psi(W; \theta, \eta) :=
\]

\[
(g(1, X) - g(0, X)) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)} - \theta
\]

- Recall the score function has to = 0.
- So we’re saying we want the three big terms in the middle to = \theta.
- Let’s look at them more closely.
The ATE

\[
\underbrace{(g(1, X) - g(0, X))}_\text{Biased treatment effect estimate from ML models} + \underbrace{\frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}}_\text{Debiasing terms}
\]
The ATE

\[
\frac{(g(1, X) - g(0, X))}{m(X)} + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}
\]

Biased treatment effect estimate from ML models

Debiasing terms

- **\(g(1, X)\):** predicted outcome given \(X\) when \(D = 1\), i.e., predicted outcome for \textit{treated} units
The ATE

\[
(g(1, X) - g(0, X)) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}
\]

- \(g(1, X)\): predicted outcome given \(X\) when \(D = 1\), i.e., predicted outcome for \textbf{treated} units
- \(g(0, X)\): predicted outcome given \(X\) when \(D = 0\), i.e., predicted outcome for \textbf{control} units
The ATE

\[
\left( g(1, X) - g(0, X) \right) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}
\]

- Biased treatment effect estimate from ML models
- Residuals for treated divided by probability of receiving treatment
- Residuals for control divided by probability of receiving control

Recall that we can define the ATE as \( E[Y(1) - Y(0)] \) in potential outcomes notation, where \( Y(1) \) and \( Y(0) \) represent potential outcomes under treatment and control, respectively. When \( \hat{Y}(1) \) is downwardly biased, our \( \hat{ATE} \) will be biased downward. When \( \hat{Y}(0) \) is downwardly biased, our \( \hat{ATE} \) will be biased upward.
The ATE

\[
(g(1, X) - g(0, X)) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}
\]

- Biased treatment effect estimate from ML models
- Residuals for treated divided by probability of receiving treatment
- Residuals for control divided by probability of receiving control

Recall that we can define the ATE as \( E[Y(1) - Y(0)] \) in potential outcomes notation, where \( Y(1) \) and \( Y(0) \) represent potential outcomes under treatment and control, respectively.
The ATE

\[
\begin{align*}
(g(1, X) - g(0, X)) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}
\end{align*}
\]

- Biased treatment effect estimate from ML models
- Residuals for treated divided by probability of receiving treatment
- Residuals for control divided by probability of receiving control

- Recall that we can define the ATE as \( E[Y(1) - Y(0)] \) in potential outcomes notation, where \( Y(1) \) and \( Y(0) \) represent potential outcomes under treatment and control, respectively.
- When \( \hat{Y}(1) \) is downwardly biased, our \( \hat{\text{ATE}} \) will be biased downward.
The ATE

\[
\underbrace{(g(1, X) - g(0, X))}_{\text{Biased treatment effect estimate from ML models}} + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}
\]

- Residuals for treated divided by probability of receiving treatment
- Residuals for control divided by probability of receiving control

Recall that we can define the ATE as \( E[Y(1) - Y(0)] \) in potential outcomes notation, where \( Y(1) \) and \( Y(0) \) represent potential outcomes under treatment and control, respectively.

- When \( \hat{Y}(1) \) is downwardly biased, our \( \hat{\text{ATE}} \) will be biased downward.
- When \( \hat{Y}(0) \) is downwardly biased, our \( \hat{\text{ATE}} \) will be biased upward.
The ATE

\[ \left( \frac{g(1, X) - g(0, X)}{m(X)} \right) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)} \]

- Biased treatment effect estimate from ML models
- Residuals for treated divided by probability of receiving treatment
- Residuals for control divided by probability of receiving control

That's why we want to add the residuals for the treated units and subtract the residuals for the control units. To see this, note that, e.g., \( D(Y - g(1, X)) \) represents the observed potential outcome under treatment minus the predicted potential outcome under treatment.
The ATE

\[
\begin{align*}
(g(1, X) - g(0, X)) & + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)} \\
\end{align*}
\]

- Biased treatment effect estimate from ML models
- Residuals for treated divided by probability of receiving treatment
- Residuals for control divided by probability of receiving control

That’s why we want to add the residuals for the treated units and subtract the residuals for the control units.
The ATE

\[
\left( g(1, X) - g(0, X) \right) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}
\]

- Biased treatment effect estimate from ML models
- Residuals for treated divided by probability of receiving treatment
- Residuals for control divided by probability of receiving control

That’s why we want to **add** the residuals for the treated units and **subtract** the residuals for the control units.

To see this, note that, e.g., \( D(Y - g(1, X)) \) represents the **observed** potential outcome under treatment minus the **predicted** potential outcome under treatment.
The ATE

\[
\left( g(1, X) - g(0, X) \right) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}
\]

- Biased treatment effect estimate from ML models
- Residuals for treated divided by probability of receiving treatment
- Residuals for control divided by probability of receiving control
The ATE

\[
\left( g(1, X) - g(0, X) \right) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}
\]

- Biased treatment effect estimate from ML models
- Residuals for treated divided by probability of receiving treatment
- Residuals for control divided by probability of receiving control

Finally, we weight by the inverse probability of treatment \( \frac{1}{m(X)} \) for the treated units because units with high probability of treatment will be overrepresented among the treated units.
The ATE

\[
\begin{align*}
(g(1, X) - g(0, X)) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}
\end{align*}
\]

- Biased treatment effect estimate from ML models
- Residuals for treated divided by probability of receiving treatment
- Residuals for control divided by probability of receiving control

Finally, we weight by the inverse probability of treatment \( \frac{1}{m(X)} \) for the treated units because units with high probability of treatment will be overrepresented among the treated units.

And we weight by the inverse probability of control \( \frac{1}{1 - m(X)} \) for the control units because units with high probability of control will be overrepresented among the control units.
The ATE

\[
\left( g(1, X) - g(0, X) \right) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}
\]

- Biased treatment effect estimate from ML models
- Residuals for treated divided by probability of receiving treatment
- Residuals for control divided by probability of receiving control

Important: be sure to assess common support! If the probability of treatment or control is very low in some strata of \( X \), the debiasing terms will blow up. Lack of common support → unstable estimates of treatment effects.
The ATE

\[
(g(1, X) - g(0, X)) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}
\]

- **Important**: be sure to assess common support!
The ATE

\[
\frac{(g(1, X) - g(0, X))}{m(X)} + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}
\]

- **Important**: be sure to assess common support!
- If the probability of treatment or control is very low in some strata of \(X\), the debiasing terms will blow up
The ATE

\[
\left( g(1, X) - g(0, X) \right) + \frac{D(Y - g(1, X))}{m(X)} - \frac{(1 - D)(Y - g(0, X))}{1 - m(X)}
\]

Biased treatment effect estimate from ML models

Residuals for treated divided by probability of receiving treatment

Residuals for control divided by probability of receiving control

- **Important:** be sure to assess common support!
- If the probability of treatment or control is very low in some strata of \( X \), the debiasing terms will blow up
- Lack of common support \( \rightarrow \) unstable estimates of treatment effects
Variance and Confidence Intervals

To get valid confidence intervals, we assume that score are linear in the following sense:

\[ \psi(w; \theta, \eta) = \psi_a(w; \eta) \theta + \psi_b(w; \eta) \]

\( \forall w \in W, \theta \in \Theta, \eta \in T \)

We use this new term \( \psi_a(w; \eta) \) to estimate the asymptotic variance of our estimator.
Variance and Confidence Intervals

To get valid confidence intervals, we assume that scores are linear in the following sense:

\[ \psi(w; \theta, \eta) = \psi^a(w; \eta)\theta + \psi^b(w; \eta) \quad \forall \quad w \in \mathcal{W}, \theta \in \Theta, \eta \in T \]
Variance and Confidence Intervals

To get valid confidence intervals, we assume that score are linear in the following sense:

$$\psi(w; \theta, \eta) = \psi^a(w; \eta)\theta + \psi^b(w; \eta) \quad \forall \quad w \in \mathcal{W}, \theta \in \Theta, \eta \in \mathcal{T}$$

We use this new term $$\psi^a(w; \eta)$$ to estimate the asymptotic variance of our estimator.
Variance and Confidence Intervals

We use the following estimator for the asymptotic variance of DML:

\[ \hat{\sigma}^2 = \hat{J}_0^{-1} \sum_{k=1}^{K} E_{n,k} \left[ \psi(W; \hat{\theta}_0, \hat{\eta}_0, k) \right] \psi(W; \hat{\theta}_0, \hat{\eta}_0, k)' \]

where

\[ \hat{J}_0 = \sum_{k=1}^{K} E_{n,k} \left[ \psi(W; \hat{\eta}_0, k) \right] \]

is the score function.
Variance and Confidence Intervals

- We use the following estimator for the asymptotic variance of DML:

\[ \hat{\sigma}^2 = \hat{J}_0^{-1} \left( \sum_{k=1}^{K} E_{n,k} \left[ \psi(W; \tilde{\theta}_0, \hat{\eta}_0, k) \right] \right) \]
Variance and Confidence Intervals

We use the following estimator for the asymptotic variance of DML:

\[
\hat{\sigma}^2 = \hat{J}_0^{-1} \frac{1}{K} \sum_{k=1}^{K} E_{n,k} \left[ \psi(W; \tilde{\theta}_0, \tilde{\eta}_0, k) \psi(W; \tilde{\theta}_0, \tilde{\eta}_0, k)' \right] \left( \hat{J}_0^{-1} \right)'
\]

where

\[
\hat{J}_0 = \frac{1}{K} \sum_{k=1}^{K} E_{n,k} \left[ \psi^a(W; \tilde{\eta}_0, k) \right]
\]
Wrapping Up
Wrapping Up

- Useful for selection-on-observables with high-dimensional confounding
Wrapping Up

- Useful for selection-on-observables with high-dimensional confounding
- Avoids strong functional form assumptions
Wrapping Up

- Useful for selection-on-observables with high-dimensional confounding
- Avoids strong functional form assumptions
- Two sources of bias: regularization and overfitting

\[ \sqrt{n} \text{ consistency if both nuisance parameter estimators are \( n^{-1/4} \)-consistent} \]

Asymptotically valid confidence intervals
Wrapping Up

- Useful for selection-on-observables with high-dimensional confounding
- Avoids strong functional form assumptions
- Two sources of bias: regularization and overfitting
- Two tools for eliminating bias: orthogonalization and cross-fitting
Wrapping Up

- Useful for selection-on-observables with high-dimensional confounding
- Avoids strong functional form assumptions
- Two sources of bias: regularization and overfitting
- Two tools for eliminating bias: orthogonalization and cross-fitting
- $\sqrt{n}$ consistency if both nuisance parameter estimators are $n^{\frac{1}{4}}$-consistent
Wrapping Up

- Useful for selection-on-observables with high-dimensional confounding
- Avoids strong functional form assumptions
- Two sources of bias: regularization and overfitting
- Two tools for eliminating bias: orthogonalization and cross-fitting
- $\sqrt{n}$ consistency if both nuisance parameter estimators are $n^{1/4}$-consistent
- Asymptotically valid confidence intervals