What can we machine learn (too much of) in 2SLS? A cautionary tale from nonlinear intuition

Connor Lennon Edward Rubin Glen R. Waddell*

May 13, 2021

Abstract

Machine learning (ML) evolved primarily to solve "prediction problems." The first stage of two-stage least squares (2SLS) is a prediction problem—suggesting gains from injecting ML into 2SLS's first stage. However, little guidance exists on when ML helps 2SLS—or when it hurts. We investigate the implications of inserting ML into 2SLS, decomposing the bias into three informative components. Mechanically, ML-in-2SLS procedures face issues common to prediction *and* causal-inference settings—and their interaction. Through simulation, we show linear ML methods (*e.g.*, post-Lasso) work "well," while nonlinear methods (*e.g.*, random forests, neural nets) generate substantial bias in second-stage estimates—some *exceeding* the bias in endogenous OLS.

JEL Codes: C26, C53, C18, C51, C52, C45
Keywords: machine learning, instrumental variables, two-stage least squares, simulation

*We thank Jonathan Davis, Benjamin Hansen, and Eric Zou. All remaining errors are the authors'. Lennon: Doctoral student in the Department of Economics at the University of Oregon. (clennon@uoregon.edu). Rubin: Assistant Professor of Economics, University of Oregon (edwardr@uoregon.edu). Waddell: Professor of Economics, University of Oregon, and Research Fellow, IZA Bonn (waddell@uoregon.edu).

1 Introduction

Today machine learning is everywhere—from exciting applications in image processing, linguistics, and forecasting, to obligatory sections in job-market papers, to the increasingly common seminar question: *Have you tried machine learning?* With this recent popularity, machine learning (ML) methods are appearing in an increasingly wide range of empirical econometric applications. Despite this excitement and frequent recommendations,[1] the literature has little to say regarding the appropriateness of enhancing two-stage least squares (2SLS) with ML methods (*e.g.*, what are the benefits and costs of ML-augmented 2SLS with regards to unconfoundedness, exogeneity, and strength of instruments).[2]

In this paper, we focus on a potentially promising application of ML methods: curating and generating the first-stage predictions of two-stage least squares.

The motivation behind integrating machine learning in two-stage least squares is clear: "better" first-stage predictions create a stronger second stage and, to the extent that researchers can incorporate "better" first-stage predictions, inference will strengthen. Because most ML methods are built explicitly for prediction—they typically outperform ordinary-least squares (OLS) at this task—integrating ML methods in the first stage of 2SLS seems quite natural.[3] The risks of adopting out-of-the box ML methods for 2SLS-type applications are less clear.

In this paper, we discuss several phenomena that can bias ML-based 2SLS away from its target parameters. Some of these phenomena are implications of the *forbidden regression*, which a naïve implementation of ML in 2SLS is likely to lead to: injecting estimations from a nonlinear estimator into the first stage of 2SLS (*Angrist and Krueger*, 2001; *Angrist and Pischke*, 2009; *Wooldridge*, 2010).[4,5] If a linear parametric form for the first stage is a 'well-performing' approximation for

[1] For example, *Mullainathan and Spiess* write "Machine learning... revolves around prediction" and "belongs in the part of the toolbox marked $\hat{y}$ rather than in the more familiar $\hat{\beta}$ compartment." The authors then immediately recognize that "the first stage of a linear instrumental variables regression is effectively prediction."

[2] Here, we've closely paraphrased Jeffrey Wooldridge's Twitter post of 26 April 2021, where he continues with "Even worse would be if ML becomes a *de facto* requirement for empirical work in cases where its benefits are questionable—or even when ML might be harmful."

[3] This point has been recognized in the literature (e.g., *Belloni, Chernozhukov, and Hansen* (2011); *Belloni, Chen, Chernozhukov, and Hansen* (2012); *Belloni, Chernozhukov, and Hansen* (2013); *Chernozhukov, Hansen, and Spindler* (2015); *Chernozhukov, Chetverikov, Demirer, Duflo, Hansen, Newey, and Robins* (2018); *Singh, Sahani, and Gretton* (2019); *Angrist and Frandsen* (2020); *Singh, Hosanagar, and Gandhi* (2020); *Chen, Chen, and Lewis* (2020)) inclusive of new artificial intelligence (AI) methods (e.g., *Hartford, Lewis, Leyton-Brown, and Taddy* (2017), *Bennett, Kallus, and Schnabel* (2020), *Liu, Shang, and Cheng* (2020)). Further, *ad hoc* integrations of 2SLS and ML are already appearing in applied work across a wide range of fields—estimating labor market impacts of imprisonment (*Mueller-Smith*, 2015), the effects of racial-composition shocks during the Great Migration (*Derenoncourt*, 2019), the effect of expropriation on growth (*Chen and Yeh*, 2020), the "true" size of China's GDP growth (*Chen, Chen, Hsieh, and Song*, 2019), the inter-generational transmission of health (*Bevis and Villa*, 2020), and the heterogeneous impacts of family size and parental labor supply (*Biewen and Kugler*, 2020). In a recent working paper, *Chen et al.* (2020) also recognizes this motivation, suggesting that the traditional OLS-based implementation of 2SLS "leaves on the table some variation provided by the instruments that may improve precision of estimates." *If* one is willing to accept the fairly strong assumption that any function (nonlinear or linear) of valid instruments is itself a valid instrument, *Chen et al.* (2020) provides an interesting solution to some of the challenges involved with including ML methods in 2SLS. We do not make this assumption.

[4] Because expectations and linear projections do not transmit through nonlinear functions, only an OLS regression in the first stage produce fitted values that are uncorrelated with the residuals (assuming valid instruments).

[5] The *forbidden regression* is sometimes also defined as changing the set or functional form of *controls* between the first

the underlying data-generating process, then one can get away with a simple assumption that the instruments $\mathbf{z}$ is uncorrelated with the endogenous u. If non-linearities are introduced, then the assumption of "no correlation" must be strengthened to conditional mean-independence of the instruments $\mathbf{z}$ from disturbances u—requiring more-careful consideration of the structural relationships of $\mathbf{z}, x, y$.

Other issues are less common to "traditional" econometrics but become key to understanding ML-based results. These include:

- **Recovering endogeneity, part 1:** If the prediction algorithm is *too* good, then the first-stage predictor may entirely recover the endogenous regressor (including both *good* and *bad* variation). With (*i*) a small set of valid instruments and (*ii*) a linear estimator (*e.g.*, OLS), this scenario is of less concern. But as the number of potential instruments increases and the estimator becomes more nonlinear and flexible (a hallmark of many ML methods), this concern becomes real.
- **Recovering endogeneity, part 2:** If our instruments are otherwise valid and we apply OLS in the first stage of 2SLS, the resulting $\hat{x}$ predictions are a linear combination of the exogenous instruments. Thus, $\hat{x}$ is itself exogenous. However, nonlinear combinations of the instruments $f_{nl}(\mathbf{z})$ do not guarantee exogeneity, leaving ML-based 2SLS methods susceptible to substantial bias and potential lack of consistency.[6]
- **Exclusion restrictions:** As *Angrist and Frandsen* (2020) point out, ML methods are not designed to choose exclusion restrictions. If a researcher relies on ML methods to determine a nonlinear functional form, choose instruments, and select first-stage controls in a 2SLS framework, then she ultimately must assume that the algorithm is capable of settling on a valid exclusion restriction—placing a lot of trust in ML to do something it is not typically designed to do. Specifically, nonlinear estimators generate nonlinear combinations of the original instruments and thereby require additional exclusion restrictions *beyond* the original exclusion restriction implied by the linear combination of the instruments. With highly flexible ML methods, the set of exclusion restrictions is nearly infinite—the researcher must either assume that (*i*) the ML algorithm will choose the appropriate exclusion restrictions or (*ii*) *all* possible exclusion restrictions are valid (as the algorithm's choice set is infinite).
- **Amplified bias:** As we show below, the bias of second-stage estimates in 2SLS is inversely related to the variance of the first-stage predictions ($\hat{x}$). Most ML methods reduce variance in the predictions (to reduce out-of-sample prediction performance in the canonical *bias-*

and second stages of 2SLS. Nonlinear implementations of 2SLS must also deal with this challenge.

[6] Another flavor of the *forbidden regression* involves applying different specifications of controls in the first and second stages. Most out-of-the-box ML methods do not offer a method to ensure that second-stage controls are used for prediction in the ML-based first stage (and *in the correct functional form*). There are *ad hoc* solutions to this problem—writing custom functions that implement the ML algorithm *plus* a linear specification of the controls/fixed effects, or residualizing (i.e., Frisch-Waugh-Lovell). For an example, see the `fixest` package in `R` and its `feNmlm()` function, which is written to efficiently estimate maximum likelihood models with multiple fixed-effect (*i.e.*, large factor variables). This issue is particularly important for situations where conditioning on controls/fixed effects is integral to the instruments' exogeneity. Again, ML methods will, in this way, expose researchers to potential pitfalls.

variance tradeoff). This variance-reduction strategy leads to inflated bias in second-stage applications—a consideration not typical to OLS-based 2SLS applications.

We show that most ML-rooted solutions that use common ML procedures in the first stage of 2SLS fail to improve upon standard 2SLS (*i.e.*, using OLS in the first stage)—and generate more bias. Two linear estimators—post-Lasso selection and principal component analysis (PCA)—are the exceptions. Post-Lasso and PCA perform as well, or better, than standard OLS-based 2SLS. Perhaps more importantly, we show that highly nonlinear tree-based methods (*e.g.*, random forests and boosted trees) can amplify bias—providing parameter estimates farther from *truth* than naïve OLS regressions that ignore endogeneity. Given sufficient training time, naïve implementations of neural networks in 2SLS can reproduce the original OLS bias—providing little to no advantage over traditional approaches to recovering exogenous identifying variation through 2SLS.

Ultimately we conclude that while ML methods offer many promises for a range of applications, most out-of-the-box ML methods are not well suited for two-stage least squares. Moreover, choosing the *wrong* ML method in the first stage can make the situation worse.

In Section 2 we formalize the theoretical settings and define the estimators. In Section 3 we introduce two data-generating processes—respecting that use cases will likely differ, we detail one that is rather simple in its construction and another that is more complex. In Section 4 we present the empirical results for the discussed estimators and DGPs and, and in Section 5 we discuss solutions. In Section 6 we conclude.

2 Models

2.1 The problem

Applied researchers commonly apply 2SLS to estimate the causal effect of some x on some y in a setting where the exogeneity of x cannot reasonably be assumed. In other words, where

$$y = \beta_0 + \beta_1 x + u \,, \tag{1}$$

there is concern over the potential for non-zero covariance between the variable of interest x and the disturbance u when estimating the parameter β_1.

If $\mathbf{z}$ denotes a vector of *instrumental variables*, then the first stage of a 2SLS estimates x as a function of these instruments, which can be expressed as

$$x = f(\mathbf{z}) + \varepsilon \,. \tag{2}$$

In its traditional OLS-based implementation, $f(\mathbf{z})$ is linear in $\mathbf{z}$.

Defining the predictions from (2) as $\hat{x} = f(\mathbf{z})$, the second stage of the 2SLS procedure then regresses the outcome variable y on $\hat{x}$,

$$y = \gamma_0 + \gamma_1 \hat{x} + w \,, \tag{3}$$

to achieve an estimate for β_1 in (1)—we let $\hat{\gamma}_1$ be this estimate of β_1. If the instruments are valid (i.e., predictive of x and uncorrelated with u) and $\hat{x}$ results from an OLS regression, then $\hat{x}$ will also be exogenous.[7] The second stage of OLS-implemented 2SLS then generates consistent estimates of β_1, interpreted as the causal effect of x on y.

So why adopt ML at all? Applications of 2SLS identify the effect of x on y by extracting only a fraction of the "good" variation in x. The hope for ML-infused 2SLS methods is that researchers can extract more of the good variation in x—specifically nonlinear combinations of the instruments—while still omitting the bad variation. This desire has likely increased following *Lee, McCrary, Moreira, and Porter* (2020), which argues that many traditional evaluations of instrumental variables considerably overestimate their significance.

2.2 Estimators

In the analysis below we examine three classes of 2SLS-motivated estimators:

Class 1: 'Traditional' two-stage regression methods: This set of estimators covers the standard two-stage regression estimators in an econometrician's toolbox: two-stage least squares, (unbiased) split-sample IV (*Angrist and Krueger*, 1995), the Fuller implementation of limited-information maximum likelihood (LIML) (*Anderson and Rubin*, 1949; *Fuller*, 1977), and jackknife IV (JIVE) (*Angrist, Imbens, and Krueger*, 1999). These methods overlap in three important ways: they (*i*) employ a two-stage approach (*ii*) whose first stage creates a linear combination of the instruments (*iii*) with no formal variable selection.

Class 2: Machine-curated variable selections in standard 2SLS: This second class augments the standard OLS-based version of 2SLS with variable selection/synthesis. Specifically, these methods feature an additional procedure, *prior to the first stage*, that downselects or combines $\mathbf{z}$ into a more parsimonious set of variables—it is the elements of this more parsimonious expression of $\mathbf{z}$ that then appear in the first stage. The rest of the 2SLS process proceeds as usual (*i.e.*, OLS). Importantly, while these models feature variable selection/synthesis, they also preserve linearity in both stages—without any regularization, or penalization. Because these estimates center on linear combinations of $\mathbf{z}$, the original exclusion restriction of $\mathbf{z}$ passes through to the selected/synthesized instruments.

The first of machine-curated methods is the post-Lasso procedure of *Belloni et al.* (2012) that first estimates the linear relationship between x and $\mathbf{z}$ (a linearized version of 2) using penalized

[7] We are assuming homogeneous treatment effects, which removes the requirement of monotonicity.

regression. This penalized regression minimizes the sum of squared error (SSE) *plus* a penalty proportional to the sum of the coefficients' magnitudes. That is, $\lambda \times \|\gamma\|$, where γ is the vector of coefficients on the (standardized) instruments and λ is a the shrinkage parameter chosen by the researcher (typically via cross validation). Because each instrument's coefficient-based penalty changes discontinuously when moving from $\gamma_i = 0$, Lasso can be used to select a set of *stronger* instruments (whose coefficients are non-zero). Post-Lasso selects the instruments whose coefficients are non-zero and then estimates standard, OLS-based 2SLS using those selected instruments.[8]

Principal-component analysis (PCA) offers an alternative route to simplifying $\mathbf{z}$ by selecting $\mathbf{z}$'s first k principal components (*Pearson*, 1901). Thus, as the second machine-curated method we consider, Principal-component analysis (PCA) applied to 2SLS (as in *Ng and Bai* (2009); *Winkelried and Smith* (2011)) passes this set of principal components into the first stage of standard OLS-based 2SLS. While PCA may reduce the first stage's interpretability, this approach can drastically reduce the number of first-stage instruments while retaining considerable explanatory power.

Class 3: ML-based first stages in 2SLS: Our final class of estimators retains the general two-step framework of 2SLS but replaces the first stage with a variety of cross-validated ML algorithms. We evaluate a meaningful subset of machine-learning methods suitable for regression, including random forest (*Ho*, 1995; *Breiman*, 2001), boosted trees (*Breiman*, 1997; *Mason, Baxter, Bartlett, and Frean*, 1999; *Friedman*, 2001, 2002), neural networks (*Turing*, 1948; *McCulloch and Pitts*, 1943; *Farley and Clark*, 1954),[9] and Lasso (*Tibshirani*, 1996; *Santosa and Symes*, 1986).[10] Notably, most of these algorithms offer considerable flexibility (*e.g.*, nonlinearity in $\mathbf{z}$) and variable selection (to varying degrees). This class offers considerable insights into the merits of off-the-shelf ML methods' for machine-assisted 2SLS.

3 Data-generating processes

In order to examine the performance of ML in the predictive stage of 2SLS—in absolute terms and relative to *traditional* options—we employ two general data-generating processes (DGPs). For reasons that will become clear as we describe each, we refer to them as the *low-complexity* case and the *high-complexity* case. While subjective, our intention is to provide bookends of a

[8] *Angrist and Frandsen* (2020) notes that this methodology may suffer from potentially unseen pre-test bias. Because our model comes from relatively strong instruments, as with the intuition of *Zhao, Witten, and Shojaie* (2020), we do not estimate de-biased Lasso models. We therefore allow post-Lasso to serve as a representation of both.

[9] For our purposes, the contributions of *Srivastava, Hinton, Krizhevsky, Sutskever, and Salakhutdinov* (2014), *Ioffe and Szegedy* (2015), and *Kingma and Ba* (2017) are particularly relevant (for dropout, batch normalization, and stochastic optimization, respectively).

[10] For a nice review of ML methods in applied economics—including Lasso, tree-based methods, and neural networks—please see *Storm, Baylis, and Heckelei* (2019). For broader and more in-depth coverage (from the authors of many of the methods), see *James, Witten, Hastie, and Tibshirani* (2013) and *Hastie, Tibshirani, and Friedman* (2009).

sort, as the applied researcher rarely knows the extent to which her case is *complex*—particularly in terms of extent of nonlinearity or the efficient number of instruments.

3.1 A *low-complexity* case

The motivation for this case is to depict the estimators' performances when the DGP is simple and closely matches the ideal scenario for OLS-based 2SLS: an endogenous regressor that is a linear combination of a relatively small set of strong, valid instruments. For example, we imagine this case appealing to researchers seeking to estimate the causal effect of a variable of interest x_1 on outcome y, *i.e.*,

$$y = \beta_0 + \beta_1 x_1 + \varepsilon_y \,, \tag{4}$$

but facing the challenge—omitted variables, simultaneity, *etc.*—that x_1 is endogenous and $\mathrm{E}\left[\varepsilon_y \middle| x_1\right] \neq 0$ prevents OLS from cleanly identifying β_1 in (4). (Note that the causal effect β_1 is common across all individuals—this ensures that differences across estimators are not due to the estimators recovering different local average treatment effects (LATEs).)

In this low-complexity scenario ML-based 2SLS methods are overkill: neither variable selection, nor nonlinearity are necessary.[11]

Specifically, capturing a scenario where there is a single endogenous variable of interest x_1 with a small set of valid (and *individually* strong) instruments, we model the DGP as

$$\begin{aligned} \varepsilon_y &= \beta_2 x_2 + \eta \,, \\ x_2 &= 1 + \varepsilon_c \,, \\ x_1 &= g_x(\mathbf{z}) + \varepsilon_c \,, \end{aligned}$$

drawing special attention to the inclusion of ε_c as the disturbance common to both x_1 (the variable of interest) and x_2 (the *omitted* variable). This common error follows a standard normal distribution; η is distributed uniformly between -1 and 1.

We assume that a set of valid instruments $\mathbf{z}$ exists such that $\mathrm{E}\left[\varepsilon_y \,\middle|\, \mathbf{z}\right] = 0$ and $\mathrm{E}[x_1 \mid \mathbf{z}] \neq 0$ (we focus on the case where $|\mathbf{z}| = 7$). We also imagine that the researcher has no beliefs/insights about the functional form of $g_x(\cdot)$, as is likely the case in practice. In the true DGP for this case, $g_x(\mathbf{z}) = \sum_{i=1}^{7} z_i$. (That is, $g_x(\cdot)$ is linear).

In particular, we draw the instruments $\mathbf{z}$ from a multivariate normal distribution centered at zero (*i.e.*, $\mathrm{E}[\mathbf{z}] = \mathbf{0}$) with variance-covariance matrix $\Sigma_{\mathbf{z}}$ where $\mathrm{Cov}(z_i, z_j) = 0.6^{|h-k|}$ (and thus $\mathrm{Var}(z_i) = 1$ for each i). By implication, $x_1 \sim N(0, \text{ Grand Sum}(\Sigma_{\mathbf{z}}) + 1)$.

[11] Interestingly, as our results demonstrate, ML methods can increase bias relative to 2SLS and even endogenous OLS.

In full, then, the data represents the following system of equations:

$$
\begin{aligned}
y &= \beta_0(=1) + \beta_1(=1)x_1 + \beta_2(=1)x_2 + \eta \ , \\
x_2 &= 1 + \varepsilon_s \ , \\
x_1 &= g_x(\mathbf{z}, \varepsilon_s) = \sum_{i=1}^{7} z_i + \varepsilon_s \ .
\end{aligned}
$$

Importantly, the specification of the instruments in this DGP produces a very strong first-stage with a relatively large *concentration parameter* (*Belloni et al.*, 2012). Put simply, the concentration parameter μ^2 describes the extent to which the weak-instrument problem may arise within a given DGP. A higher value implies that 2SLS, without variable selection, will converge to the true β_1 at relatively small sample sizes.[12] (We discuss this further in the *high-complexity case* section below.) Consequently, the low-complexity case allows us to test how machine-curated first stages perform when there is little to be gained from variable selection/synthesis.

3.2 A *high-complexity* case

As our high-complexity case, we follow *Belloni et al.* (2012)'s DPG with two extensions. This DGP allows the researcher to customize instruments' strength, and with *many* instruments. Following *Belloni et al.* (2012), the DGP in our high-complexity case results from

$$
\begin{aligned}
y &= \beta_0 + \beta_1 x_1 + \varepsilon_y \ , \\
x_1 &= \boldsymbol{\pi}\mathbf{z} + \varepsilon_v \ ,
\end{aligned}
$$

where

$$
\begin{aligned}
(\varepsilon_y, \varepsilon_v) &\sim N\left(0, \begin{bmatrix} \sigma_y^2 & \sigma_y\sigma_v \\ \sigma_v\sigma_y & \sigma_v^2 \end{bmatrix}\right) \ , \\
\mathbf{z} &= \begin{bmatrix} z_1 & z_2 & \cdots & z_{100} \end{bmatrix} \sim N(\mathbf{0}, \Sigma_z) \ , \\
\Sigma_z[j,j] &= \text{Var}(z_j) = \sigma_j^2 = 1, \ \ \forall j \in \{1, \ldots, 100\} \ , \textit{and} \\
\Sigma_z[j,k] &= \text{Cov}(z_j, z_k) = \text{Cor}(z_j, z_k) = 0.6^{|j-k|}, \ \ \forall (j,k) \in \{1, \ldots, 100\} \ .
\end{aligned}
$$

As before, the researcher's interest is in identifying β_1. However, unlike the earlier DGP, the high-complexity case produces sets of relevant and exogenous instruments that vary in their correlation and individual strength (*i.e.*, π_i).

In defining the "exponential" design of the *first-stage* coefficient vector $\boldsymbol{\pi}$, we follow *Belloni et al.* (2012): $\boldsymbol{\pi}$ captures a "beta pattern" $\widetilde{\pi} = (0.7^0, 0.7^1, 0.7^2, \ \ldots, 0.7^{99})$ that is then multiplied by

[12] In this "low-complexity" case, $\mu^2 \approx n \times 20.71$, which exceeds the values in *Belloni et al.* (2012).

a constant C, i.e., $\boldsymbol{\pi} = C \times \widetilde{\boldsymbol{\pi}}$. The constant C implies a value for the concentration parameter, $\mu^2 = \frac{n\boldsymbol{\pi}'\Sigma_z\boldsymbol{\pi}}{\sigma_v^2}$. Panels 1i–1iii (Figure 1) illustrates the three beta patterns that we adopt in the 'high-complexity' DGP—generating three subcases of this DGP. As described above, and in greater depth in *Hansen, Hausman, and Newey* (2008), the concentration parameter is useful for determining the behavior of IV estimators. Because we are less interested in the case of weak instruments, we use $\mu^2 = 180$, which creates a strong set of instruments as outlined in *Belloni et al.* (2012).[13]

Belloni et al. (2012) arrange the coefficients $\boldsymbol{\pi}$ in descending order (*i.e.*, $\pi_1 > \pi_2 > \cdots > \pi_{100}$). However, the definition of Σ_z implies that 'proximate' instruments are more correlated than 'distant' instruments—*i.e.*, $\text{Cor}(z_i, z_{i+1}) > \text{Cor}(z_i, z_{i+k})$ for $k > 1$. Thus, the DGP of *Belloni et al.* (2012) ensures that it is the strongest instruments that are correlated with each other. While this feature may be desirable in many contexts, we will remain agnostic with regard to whether the strongest instruments are most correlated with each other or with other instruments. However, this does require that we consider three sub-cases that each arise from different orderings of the coefficients in $\boldsymbol{\pi}$:

- **Randomly shuffled:** After generating the coefficients, we randomly re-order them to break the relationship between instruments' strengths and their covariance (Σ_z).
- **Descending from** z_1: In this subcase, as in *Belloni et al.* (2012), $\pi_1 > \pi_2 > \cdots > \pi_{100}$.
- **Descending from** z_{50}: Here we modify *Belloni et al.* (2012) by defining π_{50} as the largest coefficient: $\pi_{50} > \pi_{51} > \cdots > \pi_{100} > \pi_1 > \pi_2 > \cdots > \pi_{49}$. Because "proximate" instruments are correlated in Σ_z, this subcase implies that the strongest instrument (z_{50}) is very correlated both with the second-strongest instrument (z_{51}) and with the weakest instrument (z_{49}).

Finally, we define $\sigma_v^2 = \boldsymbol{\pi}'\Sigma_z\boldsymbol{\pi}$ (which forces that $\text{Var}(x_1) = 1$) and $\sigma_y = 1$. In panels 1i-1iii of Figure 1 we illustrate the cross-instrument correlations implied by Σ_z: in Panel 1iv we show a correlation matrix among the 100 instruments, and in Panel 1v we highlight the correlation of z_1 and z_{50} to each of the other 100 instruments. Instruments are strongly correlated with their neighbors and weakly correlated with non-neighbors—limiting the information accessible from any single instrument.

4 Results

Now we turn to discussing the results of our simulations. Among the simulations, we will include an "oracle model" that extracts the exogeneous component of x_1 in its entirety (perfectly removing endogeneity) and a simple OLS model (where we entirely ignore endogeneity). While one might expect the oracle and plain OLS models to bookend 2SLS models, our simulations demonstrate

[13] It is important to select this value thoughtfully. Choosing a μ^2 that is too small will simulate a weak-instruments problem. Choosing a μ^2 that is too large will yield a scenario in which all instruments are "overpoweringly" valid, which reduces the effectiveness of selection or dimension-reduction techniques.

that they *do not*. That is, machine learning can lead to outcomes that are even worse than ignoring endogeneity.

In each case, we are interested in the performances of the estimators in terms of their potential biases and the precision of estimates. Recall that these estimators include three broad classes: (*i*) traditional methods (OLS-based 2SLS, split-sample IV, LIML, and jackknife IV), (*ii*) machine-curated 2SLS (variable-selection or -curation via post-Lasso and PCA), and (*iii*) 2SLS applications with ML-powered predictions in their first stages (*i.e.*, replacing first-stage OLS with either Lasso, boosted trees, random forests, or neural networks).

4.1 Which hammer?

In Figure 2 we depict the distributions of point estimates ($\hat{\beta}_1$) for a given method in given DGPs—in Panel 2i we illustrate the low-complexity case, and in panels 2ii–2iv we represent the high-complexity cases.[14] We summarize simulations by their means and standard deviations in Table 1.

To those with use cases that resemble our "low-complexity case," the simulation results have a clear takeaway: PCA-based 2SLS and post-Lasso perform well and offer very safe choices.[15] Important for the practitioner: All four nonlinear ML-in-the-first-stage methods (*i.e.*, Lasso, boosted trees, neural networks, and random forests) perform poorly in terms of both bias and variance. In fact, "random-forest infused 2SLS" generates *more bias* in $\hat{\beta}_1$ than the OLS estimator that entirely ignores endogeneity—it is possible for an ML-based 2SLS estimator to *amplify* bias relative to plain OLS.[16]

In the three high-complexity cases in Table 1 (columns *B*–*D*) and in panels 2ii–2iv of Figure 2, LIML and Jackknife IV generate very little bias in their estimates of β_1, outperforming 2SLS. Across all three DGPs, 2SLS produces mean estimates roughly 2.3–5.8 percent larger than the true parameter, while the centers of LIML's and JIVE's distributions are within 0.4 percent of the true parameter. Injecting random forests into the first stage, on average, produces more biased estimates than naïve (endogenous) OLS—generating coefficient estimates that are 32–56 percent larger than the true estimates. In short, one can worsen endogeneity issues by using ML-infused 2SLS estimators.

[14] The target parameter β_1 equals 1 throughout (indicate with a thin dashed line). Each distribution results from 1,000 iterations of the simulation. Table 1 summarizes each of these method-by-DGP combinations (*i.e.*, $14 \times 4 = 56$) with the mean and standard error from each.

[15] LIML also performs well, but with slightly larger variance.

[16] It is worth noting that the Jackknife IV estimator yields *very* high variance in this low-complexity DGP, as do Neural Networks.

4.2 Decomposing the bias

To diagnose the sources of bias from different methods, we show below one can decompose the wedge between β_1 and $\hat{\beta}_1^{2SLS}$ into three components,

$$\text{Wedge} = \hat{\beta}_1^{2SLS} - \beta_1 = f\Big(\beta_1 \text{Cov}(\hat{x}, e),\ \text{Cov}(\hat{x}, u),\ \frac{1}{\text{Var}(\hat{x})}\Big), \tag{5}$$

where f is non-decreasing with respect to each of its arguments. Each component of the wedge offers insights into how first-stage methods differentially produce biases—and delivers helpful intuition regarding the pitfalls that may arise in 2SLS applications that include ML-based first stages.

To see the component parts of the bias drawing $\hat{\beta}_1^{2SLS}$ away from β_1, suppose again that the parameter of interest is β_1—the causal effect of x on y in

$$y = \beta_0 + \beta_1 x + u\ . \tag{6}$$

Suppose also that x is endogenous for some reason, *i.e.*, $\text{Cov}(x,\ u) \neq 0$. The 2SLS estimate of β_1 simply comes from estimating

$$y = \beta_0 + \beta_1 \hat{x} + w\ , \tag{7}$$

where $\hat{x}$ is the first-stage-based prediction of x from some set of valid instruments $\mathbf{z} = z_1,\ z_2,\ \ldots,\ z_p$.

Because we estimate the second stage in (7) via OLS, the estimate for β_1 can be written

$$\hat{\beta}_1{}^{2SLS} = \beta_1 + \frac{\text{Cov}(\hat{x}, w)}{\text{Var}(\hat{x})}\ . \tag{8}$$

Using (6) and (7), we can rewrite w as

$$\begin{aligned} w &= y - (\beta_0 + \beta_1 \hat{x}) \\ &= \beta_0 + \beta_1 x + u - \beta_0 - \beta_1 \hat{x} \\ &= \beta_1 (x - \hat{x}) + u \\ &= \beta_1 e + u\ , \end{aligned} \tag{9}$$

where e is the first-stage residual—the difference between x and $\hat{x}$.

Using this expression for w, we can decompose the covariance in (8) into two components:

$$\begin{aligned} \text{Cov}(\hat{x}, w) &= \text{Cov}(\hat{x}, \beta_1 e + u) \\ &= \beta_1 \text{Cov}(\hat{x}, e) + \text{Cov}(\hat{x}, u)\ . \end{aligned} \tag{10}$$

If the first-stage predictions ($\hat{x}$) come from OLS, then $\text{Cov}(\hat{x}, e)$ is mechanically zero. The sec-

ond term, $\text{Cov}(\hat{x}, u)$, is typically small when $\hat{x}$ comes from a linear-combination of valid instruments.

Finally, substituting (10) into (8) yields a helpful expression for the 2SLS estimate for β_1, which we can write as

$$\hat{\beta}^{2SLS} = \beta_1 + \frac{\beta_1 \text{Cov}(\hat{x}, e) + \text{Cov}(\hat{x}, u)}{\text{Var}(\hat{x})} \ . \tag{11}$$

OLS guarantees that $\text{Cov}(\hat{x}, e)$ is zero and, with valid instruments, that $\text{Cov}(\hat{x}, u)$ is small. Whether $\text{Var}(\hat{x})$ is "small" is typically of little consequence with OLS (as $\beta_1 \text{Cov}(\hat{x}, e) + \text{Cov}(\hat{x}, u)$ is typically small). However, all three points can generate important issues when we mix ML methods into the first stage of 2SLS. With ML methods, nothing guarantees that $\text{Cov}(\hat{x}, e)$ is zero or that $\text{Cov}(\hat{x}, u)$ is small. Moreover, many ML methods are constructed to *reduce* the variance of predictions—further amplifying bias. This variance-reduction aspect is particularly relevant for nonlinear methods.

The $\beta_1 \text{Cov}(\hat{x}, e)$ component

For the term $\beta_1 \text{Cov}(\hat{x}, e)$ to differ from zero and generate bias, $\beta_1 \neq 0$ and $\text{Cov}(\hat{x}, e) \neq 0$. We assume that the population-regression coefficient β_1 differs from zero.[17] With this assumption imposed, the term $\beta_1 \text{Cov}(\hat{x}, e)$ only generates bias when $\text{Cov}(\hat{x}, e) \neq 0$; β_1 scales the bias and affects its direction.

By construction, OLS produces predictions that are orthogonal to their residuals, *i.e.*, $\text{Cov}(\hat{x}, e) = 0$. This first term is therefore irrelevant when the first stage uses OLS. However, when practitioners adopt other methods in the first stage (*e.g.*, non-linear methods) nothing guarantees first-stage predictions are uncorrelated with their residuals. Put differently, this part of the bias results from using estimators whose predictions correlate with their residuals (rather than resulting from a violation of the exclusion restriction). While it is possible for nonlinear methods to generate $\text{Cov}(\hat{x}, e) = 0$, many do not.

In addition, because $\text{Cov}(\hat{x}, e)$ typically drops out of OLS regression, OLS-based empirical intuition does not help here. One implication of this non-OLS intuition of $\text{Cov}(\hat{x}, e)$ is that the bias generated by it is proportional to the size of the target parameter β_1. Where treatment effects are larger, the bias transmitted through this component is also larger.

To understand the magnitude of $\text{Cov}(\hat{x}, e)$, note that it decomposes into $\text{Cov}(\hat{x}, x)$ and $\text{Var}(\hat{x})$. Thus,

$$\text{Cov}(\hat{x}, e) = \text{Cov}(\hat{x}, x) - \text{Var}(\hat{x}) \ . \tag{12}$$

[17] The case where it exactly equals zero is a measure-zero event that is uninteresting to the researcher.

While $\text{Cov}(\hat{x}, e)$ is not generally signable, its central component (*i.e.*, the covariance between $\hat{x}$ and e, $\text{Cov}(\hat{x}, e)$) is bounded between $-\text{Var}(\hat{x})$ and $\text{Cov}(\hat{x}, x)$.[18] In addition, we can sign $\beta_1 \text{Cov}(\hat{x}, e)$ in fairly general subcases:[19]

$$\begin{aligned}
\text{Sign}\Big\{ \beta_1 \text{Cov}(\hat{x}, e) \Big\} &= \text{Sign}\Big\{ \beta_1 \text{Corr}(\hat{x}, e) \Big\} \\
&= \text{Sign}\Big\{ \beta_1 \sigma_e^{-1} \Big(\text{Corr}(\hat{x}, x)\, \sigma_x - \sigma_{\hat{x}} \Big) \Big\} \\
&= \text{Sign}\Big\{ \beta_1 \Big\} \cdot \text{Sign}\Big\{ \text{Corr}(\hat{x}, x)\, \sigma_x - \sigma_{\hat{x}} \Big\} \\
&= \begin{cases}
(+) & \text{if } \beta_1 > 0 \text{ and } \text{Corr}(\hat{x}, x)\, \sigma_x > \sigma_{\hat{x}} \\
(-) & \text{if } \beta_1 > 0 \text{ and } \text{Corr}(\hat{x}, x)\, \sigma_x < \sigma_{\hat{x}} \\
(-) & \text{if } \beta_1 < 0 \text{ and } \text{Corr}(\hat{x}, x)\, \sigma_x > \sigma_{\hat{x}} \\
(+) & \text{if } \beta_1 < 0 \text{ and } \text{Corr}(\hat{x}, x)\, \sigma_x < \sigma_{\hat{x}} \\
0 & \text{if } \beta_1 = 0 \text{ or } \text{Corr}(\hat{x}, x)\, \sigma_x = \sigma_{\hat{x}}
\end{cases}
\end{aligned} \tag{13}$$

where σ_x refers to the standard deviation of x ($\sigma_{\hat{x}}$ and σ_e are defined similarly).

As (13) reveals, the sign of $\text{Cov}(\hat{x}, e)$ depends on two quantities: (*i*) the sign of β_1, and (*ii*) the sign of $\text{Corr}(\hat{x}, x)\, \sigma_x - \sigma_{\hat{x}}$. It is difficult to generalize the sign of $\text{Cov}(\hat{x}, e)$ without further assumptions. While one may be tempted to assume $\sigma_x > \sigma_{\hat{x}}$, this assumption is not sufficient for signing $\text{Cov}(\hat{x}, e)$, as it still depends upon the magnitude of $\text{Corr}(\hat{x}, x)$.[20] The knife-edge case where $a = 0$ appears unlikely except in cases where either $\beta_1 = 0$ or where $\text{Cov}(\hat{x}, e)$ is mechanically zero (*e.g.*, OLS).

Across the twelve models that we consider in Table 2, only the non-OLS models produce $\text{Cov}(\hat{x}, e) \neq 0$ (it is mechanically zero for OLS-based models)—this is unsurprisingly. Lasso, neural nets, boosted trees, and random forests all produce positive covariance between $\hat{x}$ and e. In other words, in all of our DGPs, the term $\text{Cov}(\hat{x}, e)$ biases $\hat{\beta}$ upward (positively) whenever it is non-zero.[21] Random forest models generate the largest covariance between $\hat{x}$ and e (and consequently the largest $\text{Cov}(\hat{x}, e)$) in each of the DGPs. Depending upon the DGP, Lasso, neural nets, and boosted trees generate the second-highest covariance. Because our *shallow* subcase of neural nets approximates OLS, its covariance between $\hat{x}$ and e approximately zero.

One way to ensure that $\text{Cov}(\hat{x}, e) = 0$ for a nonlinear model is to linearize its output—*e.g.*, by using the ML-based prediction $\hat{x}(\mathbf{z})$ as an *instrument* for x, rather than plugging it into the second stage (*Angrist and Krueger*, 2001; *Chen et al.*, 2020). While this approach forces $\text{Cov}(\hat{x}, e) = 0$, it requires strengthening assumptions (as we discuss in Section 5).

[18] We assume estimates, $\hat{x}$, will have non-negative covariance with the true values, x.
[19] We assume e, x, and $\hat{x}$ have variation and that the predictions $\hat{x}$ positively correlate with the true values x.
[20] Further, this assumption is equivalent to making an assumption on $\text{Cov}(\hat{x}, e)$, which means one is essentially assuming the result. That is, $\text{Var}(x) = \text{Var}(\hat{x}) + \text{Var}(e) + 2\,\text{Cov}(\hat{x}, e)$. That said, in every iteration of our simulations, $\text{Var}(x) > \text{Var}(\hat{x})$.
[21] This upward bias is partly due to the true parameter β_1 being positive.

More broadly, the component of bias due to covariance between first stage predictions ($\hat{x}$) and their residuals (e)—the Cov($\hat{x}, e$) term—accounts for the vast majority of the bias for Lasso and substantial amounts of the bias in random forests, boosted trees, and neural nets (the exact portion of the bias differs across DGPs and iterations). While Cov($\hat{x}, e$) does not account for all of the bias, the non-zero covariance between first-stage predictions and residuals is an important (potentially large) component of the bias of ML-infused 2SLS models.

The Cov($\hat{x}, u$) **component**

Unlike Cov($\hat{x}, e$), the second component of the wedge between $\hat{\beta}_1^{2SLS}$ and β_1 can be non-zero for both OLS-based methods and non-OLS models.[22] However, methods that use *non-linear* predictions of x in the first stage (*i.e.*, ML-assisted 2SLS) require special care to produce low-bias estimates of β_1.

This second term, Cov($\hat{x}, u$), is effectively the exclusion restriction, and any 2SLS-inspired estimator can reduce bias in $\hat{\beta}$ by ensuring Cov($\hat{x}, u$) is approximately zero. Assuming the instruments $\mathbf{z}$ are valid, an arbitrary prediction algorithm can maintain Cov($\hat{x}, u$) ≈ 0 through either of three conditions:

1. **Restrict the algorithm's choice set:** By restricting the learning algorithm to choosing from a set/class of functions where each individual function satisfies the exclusion restriction, one mechanically ensures the first-stage predictions $\hat{x}$ do not covary with the unobserved disturbance u. For example, when we employ OLS in the first stage of 2SLS, the first-stage regression is chosen from a set of class of linear functions that all include valid exclusion restrictions—linear combinations of the exogenous instruments (all linear combinations of exogenous instruments are themselves exogenous).
2. **Extend the exclusion restriction:** One may extend the assumption underlying the exclusion restriction into a much stronger assumption. Rather than only assuming all *linear* combinations of the valid instruments are (which is directly implied by instruments' validity), one could assume that **all** functions of the instruments—nonlinear and linear—satisfy the exclusion restriction. Put differently, this condition requires Cov($f(\mathbf{z}), u$) ≈ 0 for all functions f.
3. **Lean *very* hard on the ML algorithm:** The final option is to simply rely upon the algorithm to find a function that satisfies the exclusion restriction, irrespective of choice set—something akin to closing one's eyes and hoping for the best. While this option makes a rather heroic assumption, as ML algorithms are typically not designed to search for and find valid exclusion restrictions, it is the default scenario. If a practitioner does not enforce condition **1** and does not assume condition **2**, then she is left with **3**.

[22] For example, a miss-specified OLS regression or any 2SLS regression. While 2SLS is a biased estimator (*Cov*($\hat{x}, u$) $\neq 0$), its bias increases substantially when instruments are not exogenous (*Cov*(z, u) $\neq 0$) or not relevant (*Cov*(z, x) ≈ 0).

In summary, sufficiently flexible learning algorithms can recover the endogenous variation in x *using only the instruments*. In fact, most ML training processes incentivize algorithms to do exactly this.

Column b of Table 2 documents the tendency of flexible first-stage models (*e.g.*, tree methods and neural nets) to recover endogeneity. As the learning algorithms permit more flexibility, $\text{Cov}(\hat{x}, u)$ tends to increase (across all DGPs). This covariance and its associated bias are particularly large for tree-based methods (especially random forests) and neural nets with multiple hidden layers. Notably, in Panels b–d of Figure 2, the densities of unrestricted and narrow neural networks are bimodal. As Appendix Figure A3 illustrates, the bimodality results from whether the neural network (i) "chooses" 0 hidden layers (the less biased mode) or (ii) goes deeper (learning the endogenous error and generating more bias).[23,24] This covariance between predictions $\hat{x}$ and the unobserved disturbance u accounts for a substantial amount of the bias in nonlinear methods—demonstrating that the previously discussed first component ($1 = \text{Cov}(\hat{x}, e) \neq 0$) is not the only issue facing these models.

The $\frac{1}{\text{Var}(\hat{x})}$ component

While the first two bias components enter additively, the third component scales their sum. Any method that reduces the variance of the first-stage predictions—reduces $\text{Var}(\hat{x})$—mechanically inflates the bias produced by $\beta_1 \text{Cov}(\hat{x}, e) + \text{Cov}(\hat{x}, u)$.

In the case of properly specified, OLS-based, 2SLS, the variance of the predictions hardly affects bias in β_1, since $\text{Cov}(\hat{x}, e) = 0$ and $\text{Cov}(\hat{x}, u) \approx 0$. However, as most ML algorithms implicitly reduce the variance of their predictions to optimally trade off between out-of-sample bias and variance. This trade off between bias and variance happens *outside of a 2SLS framework*—when practitioners infuse variance-reducing ML methods into 2SLS, the variance reduction actually amplifies bias in the second-stage estimates.

Taking these insights to the results in Panel B of Table 2, notice that variance reduction can cause methods to perform poorly. For example, Lasso-assisted 2SLS produces the lowest variance $\hat{x}$ in two of the three high-complexity cases—in cases 2 and 3, Lasso-based 2SLS has the highest $1/\text{Var}\hat{x}$-based amplifier of the bias. This high degree of bias amplification generates notable bias in Lasso, relative to many other methods (evident in Figure 2). So while Lasso's $\text{Cov}(\hat{x}, e)$ and

[23] This result highlights the importance of allowing neural networks to choose no hidden layers.

[24] Another, related, concern familiar to the ML literature is *overfit*. Overfit models tend to produce larger values of $\text{Cov}(\hat{x}, u)$ than models that have been cross-validated. Though cross-validation is best/standard practice for machine-learning methods in prediction problems, here it retains importance by preventing the algorithms from overfitting the target variable x in the first stage (even when out-of-sample performance is no longer the goal). We use five-fold cross-validation (CV) to tune the hyperparameters for Lasso-, tree-, and neural-net-based methods. Our neural-net cross-validation departs from standard five-fold CV. In Appendix Section A.3 we detail our cross-validation process for training neural net. One might further avoid overfit by applying holdout-style methods—only generating predictions for observation i when i is not in the training set. JIVE, split-sample IV, and *Chen et al.* (2020) all feature this additional safeguard. We do not employ these holdout-based methods because our goal in this paper is to simulate the results of a researcher using off-the-shelf ML tools in the first stage of 2SLS.

$\text{Cov}(\hat{x}, u)$ are less than or equal to those of many other methods, the amplification produced by variance-reduction in $\hat{x}$ ultimately causes Lasso to have substantial bias. Notably, post-Lasso-based 2SLS produces less bias, partly due to the fact that it includes less variance reduction.

Worse yet, tree-based methods substantially reduce variance *and* produce relatively large $\text{Cov}(\hat{x}, e)$ and $\text{Cov}(\hat{x}, u)$ components—resulting in very large bias in their parameter estimates (even larger than naïve OLS).

5 Discussion: Potential solutions

This paper examines the implications of plugging off-the-shelf ML methods into a 2SLS framework. In many cases, injecting ML into the first stage of 2SLS generates substantial bias.

While there are many approaches to combining instrumental-variable intuition and machine learning, they relax the traditional 2SLS structure and require different, generally stronger, identifying assumptions.[25] Among the current options, the closest in spirit to our question of "What are the implications of inserting ML into 2SLS?" is the "machine learning split-sample" (MLSS) estimator proposed by *Chen et al.* (2020).[26]

With two fairly simple expansions of the traditional 2SLS framework, MLSS mitigates *most* biases generated by naïvely plugging ML methods into the first stage. However, as with other more ML-forward methods, the solution is not without the cost of substantially strengthening the exclusion restriction. Specifically, *Chen et al.* (2020) proposes augmenting 2SLS with two simple techniques: restrict ML-based predictions to be explicitly out of sample (using split-sample methods), and use the ML-generated predictions as a "synthetic" instrument that then enters linearly in the first stage.

The idea for out-of-sample (split-sample) ML predictions follows the lead of Jackknife IV and Split Sample IV. By introducing out-of-sample methods to the ML-prediction exercise, *Chen et al.* aims to prevent the ML algorithm from fitting the first-stage errors—shutting down the bias generated by $\text{Cov}(\hat{x}, u)$. The drawback, however, is that this out-of-sample step likely increases variability (as seen in the JIVE results of Figure 2).

The second component of *Chen et al.* involves a "zeroth stage" (before the first stage), in which the practitioner trains an ML algorithm to predict x using the instruments $\mathbf{z}$.[27] The predictions from this zeroth stage are then used as the instrument within a traditional 2SLS framework. The benefit here is that the resulting linear first stage—linearizing the results of a potentially

[25] See, for examples, MLSS (*Chen et al.*, 2020), DeepIV (*Hartford et al.*, 2017), DeepGMM (*Bennett et al.*, 2020), KIV (*Singh et al.*, 2019), Adversarial Estimation of Riesz Representers (*Chernozhukov, Newey, Singh, and Syrgkanis*, 2020), Neural Estimation of SEM (*Liao, Chen, Yang, Dai, Wang, and Kolar*, 2020), and Non-Parametric IV (*Kilbertus, Kusner, and Silva*, 2020).

[26] *Angrist and Frandsen* (2020) also applies split-sample methods to several ML algorithms (*i.e.*, post-Lasso, random forest)—both in the first stage of 2SLS and while synthesizing instruments in a stage that precedes the first stage.

[27] Note that this zeroth stage is identical to first stages that naïvely insert ML methods into 2SLS.

forbidden regression—guarantees that $\text{Cov}(\hat{x}, e) = 0$, shutting down one avenue in which bias enters. Importantly, this method assumes that no learnable function of instruments meaningfully predicts the structural disturbance u. As the ML algorithm's function space grows, it can cover all possible functions of the instruments (*e.g.*, most neural networks are universal approximators), which requires strengthening the exclusion restriction to all possible functions of the instruments. This strengthened exclusion restriction is generally much stronger (and likely more difficult to justify) than the typical identifying assumption assumed in 2SLS applications.

Overall, solutions exist for the bias components that we identify in this paper, but each solution comes at a cost. Some solutions are fairly cheap (*e.g.*, out-of-sample prediction methods require sufficient overlap and can increase noise). Other solutions require much more from the practitioner (*e.g.*, strengthening a linear exclusion restriction to an infinite-dimensional function space). Some researchers may be fine making these stronger identifying assumptions. However, more work needs to be done to explore and bound the relative benefits and costs of these procedures in practice—for instance, the bias from instruments that nonlinearly violate an expanded exclusion restriction (potentially discoverable by nonlinear ML algorithms) but satisfy the traditional linear exclusion restriction.

6 Conclusion

Our results show that naïvely inserting machine-learning algorithms into the first stage of a 2SLS structure often produces bias. While some channels of this bias will be familiar to an economics audience, others may be less familiar—resulting from the intersection of "classical" econometric tools with ML methods and algorithms. In terms of minimum bias, we find that the most-successful cases of integrating machine learning into 2SLS are largely restricted to instrument selection or modification (*e.g.*, post-Lasso) before entering a traditional 2SLS framework. However, inserting highly nonlinear methods (*e.g.*, random forests, boosted trees, and relatively deep neural nets) into the first stage of 2SLS drives estimates of causal parameters to be *more* biased than much simpler and more-easily interpreted alternatives. Importantly, such methods can even yield more bias than a naïve, endogenous OLS regression.

We mechanically decompose the bias within ML-augmented 2SLS estimates into three terms:

- $\beta\ \text{Cov}(\hat{x}, e)$: When first-stage predictions are not orthogonal to their residuals (relevant for nonlinear methods), $\text{Cov}(\hat{x}, e) \neq 0$. This wedge increases with the magnitude of the target parameter, β.
- $\text{Cov}(\hat{x}, u)$: Flexible ML methods can overfit x in the first stage, unintentionally recovering endogenous variation in x. This *overfit* first stage produces fitted values ($\hat{x}$) that are no longer exogenous from the structural disturbance u—even when the first-stage instruments are exogenous.

- $1/\text{Var}(\hat{x})$: ML-based prediction methods typically reduce the variance of their predictions. As such, ML-augmented 2SLS often reduces the variance of $\hat{x}$, which amplifies any non-zero wedge between the estimate and the estimand coming from $\beta\,\text{Cov}(\hat{x}, e) + \text{Cov}(\hat{x}, u)$.

Overall, we find that many of the methods pioneered in the 1990s and 2000s (*e.g.*, split-sample IV, LIML, and JIVE) reduce bias resulting from weak or over-identified IV for a linear first stage.[28] At the same time, we find that less-traditional methods that *mindfully* incorporate machine learning techniques (*e.g.*, PCA-synthesized instruments and post-Lasso) generate improvements over 2SLS across a variety of DGPs.[29] Importantly, we show that ML thoughtlessly injected into the first stage can also produce substantial bias—possibly worsening bias over naïve OLS.

The fundamental problem comes from recognizing the first stage of 2SLS solely as a prediction problem without recognizing that it is also part of a larger estimation *system*.

ML methods can produce fantastic predictions, but they were not developed or optimized for a two-stage procedure that generates minimally biased causal estimates. For example, variance reduction in prediction typically improves out-of-sample prediction accuracy but can actually inflate parameter bias in the second stage of 2SLS. Solutions have been proposed to these—and other—issues from ML-infused 2SLS. However, if one approaches these problems without the caution that we suggest—*e.g.*, blindly relying upon ML to learn the variation in the first stage—bias often results. Unfortunately, the solutions to these ML-in-2SLS problems can require strengthening exclusion restrictions—often requiring the practitioner to strengthen the exclusion restriction where complexity is highest (*e.g.*, high-dimensional data). Because ML methods are typically adopted in complex data environments that are not well understood, employing these methods without safeguards can force researchers to make heroic assumptions.

Given the performance of existing off-the-shelf estimators, naïvely injecting ML into 2SLS appears to produce little gain relative to its costs/biases. More broadly, applying ML methods to 2SLS requires the practitioner to face issues present in both the prediction *and* causal-inference settings—and additional, less-familiar issues from their interaction.

[28] *Angrist and Frandsen* (2020) comes to a similar conclusion. Focusing more on heterogeneous treatment effects and ML-assisted covariate selection, *Angrist and Frandsen* does not decompose the bias associated with ML methods applied to the instruments in the first-stage prediction problem (Section 4.2 and summarized above). Others have also found that JIVE and LIML are the best choice in the weak-many-instruments case—*Hansen and Kozbur* (2014) explores a regularized JIVE, and *Carrasco and Tchuente* (2015) develops a regularized LIML approach.

[29] See *Ackerberg and Devereux* (2009) for a discussion of tools for addressing over-identified instrumental variables estimation and *Andrews, Stock, and Sun* (2019) for a discussion of weak instruments in linear IV regression.

7 Figures

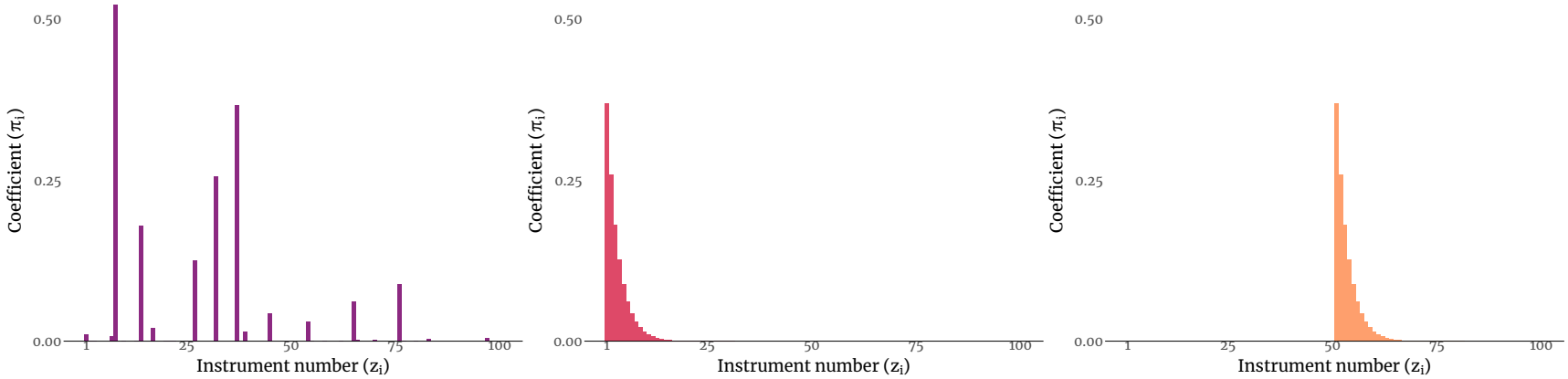
Figure 1: The "high-complexity" DGP: 100 instruments of varying strength

Order of instruments' coefficients (π)

(i) *Subcase 1:* 'Shuffled' coefficients

(ii) *Subcase 2:* Coefficients decline from z_1

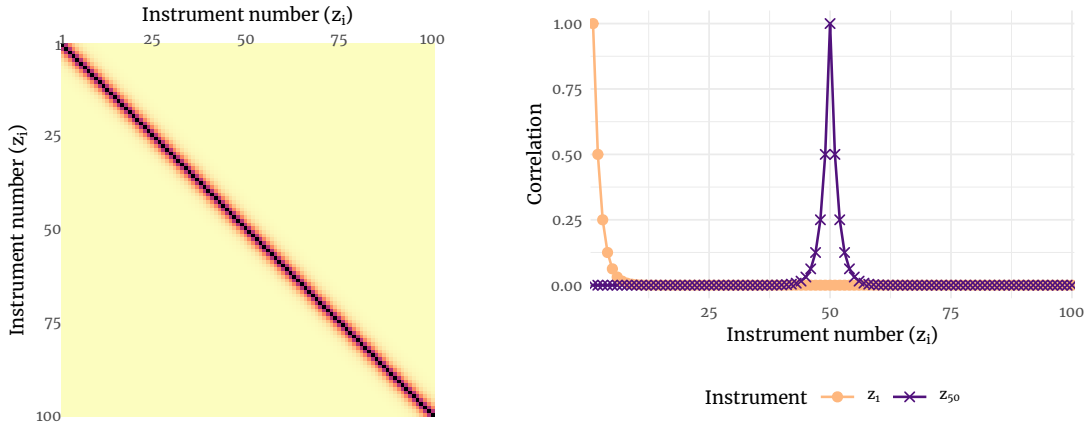
(iii) *Subcase 3:* Coefficients decline from z_{50}



Instruments' correlation (Σ_z)

(iv) Correlation between the 100 instruments

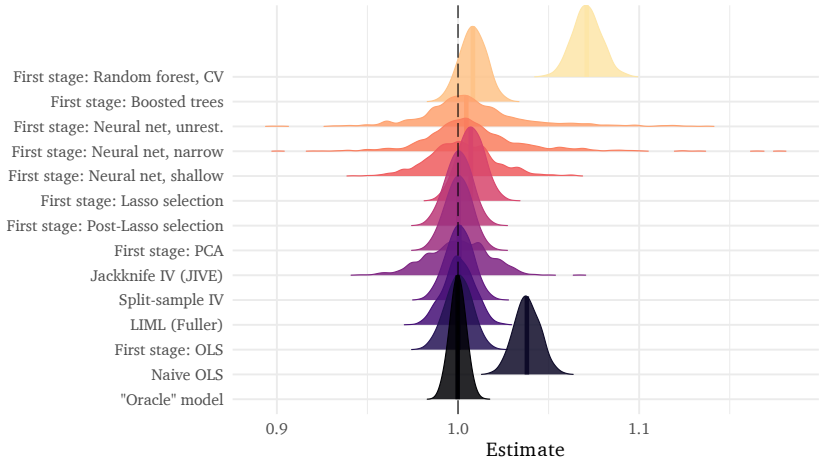
(v) Correlation of z_1 and z_{50} with other instruments



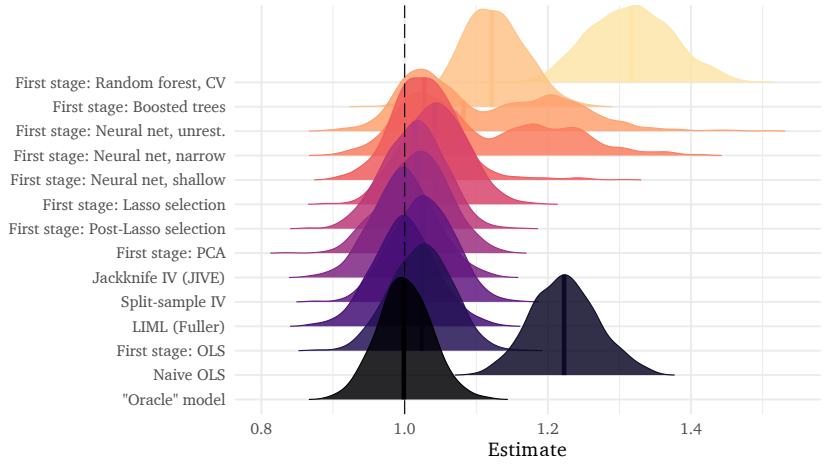
The top panels—1i, 1ii, and 1iii—illustrate the instruments' coefficients ($\boldsymbol{\pi}$) in the DGP for the first stage of each of the three subcases (*i.e.*, how the instruments $\mathbf{z}$ relate to x in each subcase). While the three subcases differ in their first-stage coefficients, they share the same variance-covariance structure Σ_z, which 1iv depicts ($\text{Var}(z_i) = 1$, $\forall i$). Figure 1v individually plots the correlation between two instruments (z_1 and z_{50}) and all of the other instruments. The instruments z_1 and z_{50} are specifically of interest because they are the strongest instruments in subcases 2 and 3, respectively.

Figure 2: Main results—$\hat{\beta}$ distributions across competing two-stage methods

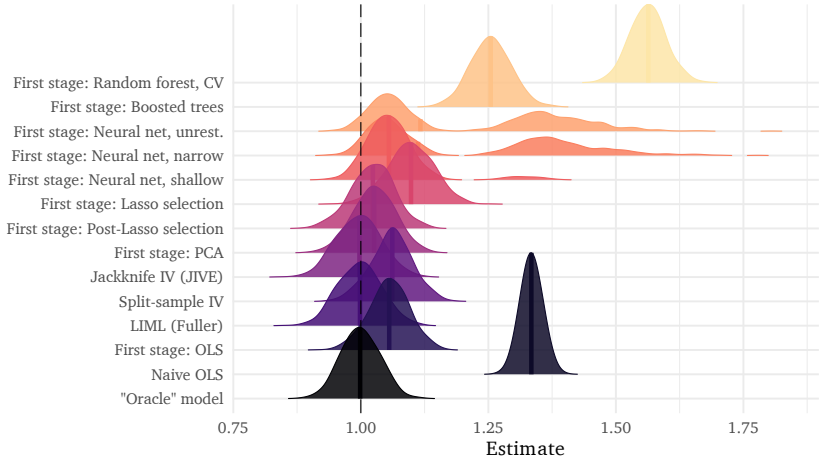
(i) *Low-complexity* case: 7 strong instruments



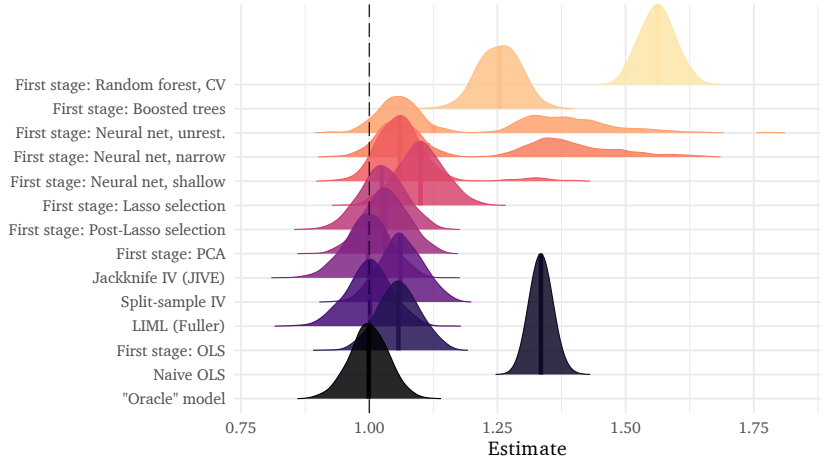
(ii) *High-complexity* case 1: 100 instruments; coefficients 'shuffled'



(iii) *High-complexity* case 2: 100 instruments; strength decreases from z_1



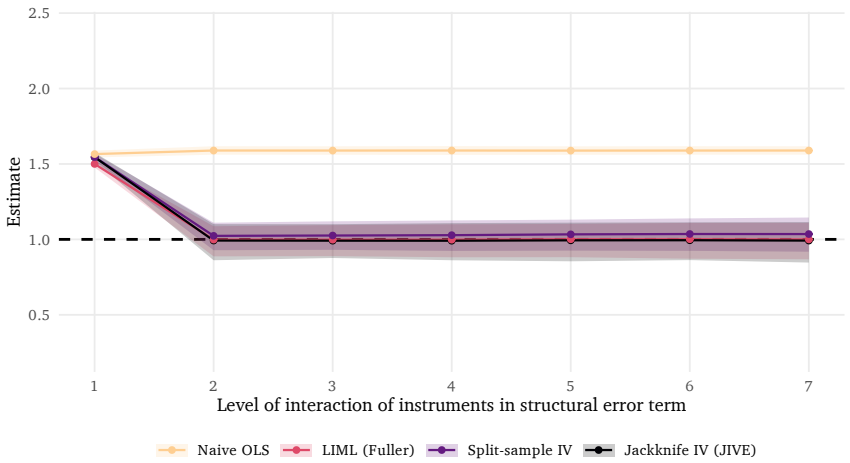
(iv) *High-complexity* case 3: 100 instruments; strength decreases from z_{50}



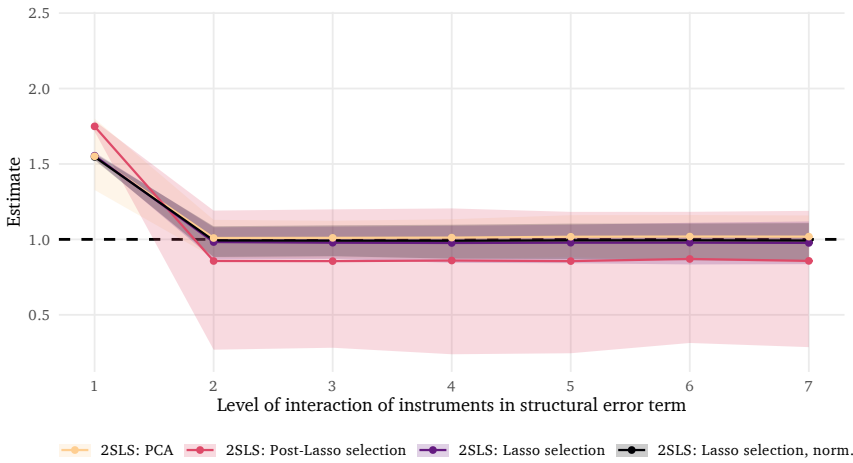
Each individual distribution represents the density of estimates from 1,000 iterations of simulation. The true value of the target parameter β is 1 (the dashed vertical line). The DGP underlying subfigure 2i uses 7 strong and exogenous instruments. For subfigures 2ii, 2iii, and 2iv, the models have access to 100 exogenous instruments of varying strengths. The three 'high-complexity' cases differ in their DGPs' first-stage coefficients but share a common variance structure among the 100 instruments (Σ_z)—following and extending *Belloni et al.* (2012). The *Oracle model* refers to a model using only the exogenous portion of x (the endogenous regressor). In *Naive OLS*, we estimate the parameter without regard for the endogeneity of x. The remaining methods each estimate the target parameter with a different variety of approaches toward 2SLS or related methods. Table 1 contains the mean and standard deviation for each distribution.

Figure 3: Exclusion-restriction violations via higher-order interactions among instruments ('low-complexity case' of 7 strong instruments)

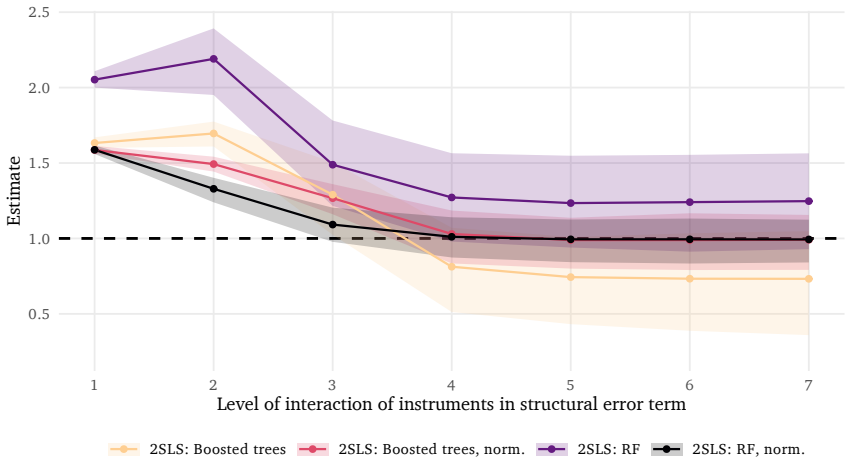
(i) 'Standard' linear estimators: OLS, LIML, SSIV, and JIVE



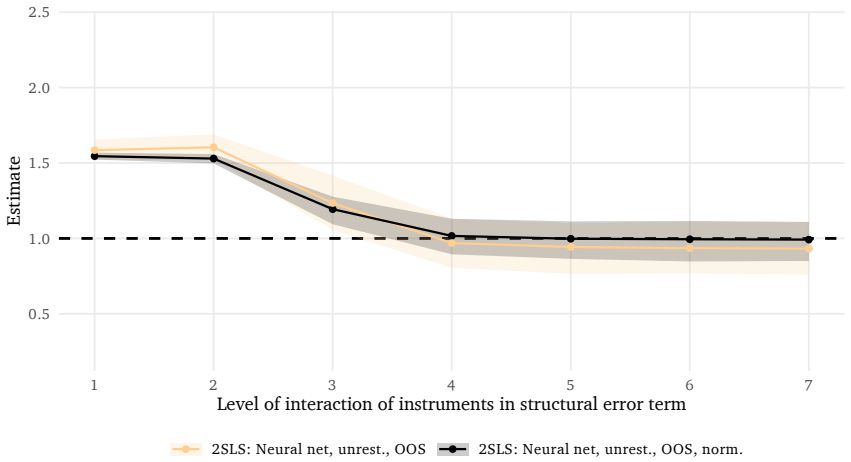
(ii) 'Selection' methods: Lasso, post-Lasso, and PCA



(iii) Trees: Random forests and boosted trees—with and without normalization



(iv) Neural networks: With and without normalization



The DGPs underlying these figures add a k-term interaction between the instruments to the structural error. *E.g.*, when the x-axis equals 3, we add the interaction $x_1 \times x_2 \times x_3$ to the error; $k = 1$ is a linear exclusion-restriction violation. Interactions with $k > 1$ do not violate the exclusion restriction of linear methods but do violate the exclusion restriction for methods that require a higher-order/expanded exclusion restriction (*i.e.*, conditional independence). Each subfigure illustrates the performance of the given estimators from 1,000 iterations. The solid lines and dots mark the mean of the estimator-DGP combination; the shaded bands give the boundaries between the 2.5th and 97.5th percentiles. The true value is 1. *Normalized* methods ('norm.') use the stated method to synthesize a single instrument from the seven instruments *prior to the first stage* (as in *Chen et al.* (2020)). See Figure A2 for the densities of 2-, 3-, 4-, and 5-way interactions.

8 Tables

Table 1: Simulation results: Mean and standard deviation for methods and DGPs

	Low-complexity case	*High-complexity* cases		
	7 strong instruments	100 *mixed* instruments		
	(*A*)	(*B*)	(*C*)	(*D*)
Naive OLS	1.038	1.335	1.334	1.223
	(0.007)	(0.020)	(0.019)	(0.046)
First stage: OLS	1.000	1.058	1.056	1.023
	(0.007)	(0.040)	(0.039)	(0.042)
LIML (Fuller)	1.000	1.000	0.998	1.000
	(0.008)	(0.044)	(0.043)	(0.043)
Split-sample IV	1.001	1.062	1.060	1.025
	(0.007)	(0.041)	(0.040)	(0.043)
Jackknife IV (JIVE)	1.000	0.998	0.996	1.000
	(0.017)	(0.044)	(0.044)	(0.044)
First stage: PCA	1.000	1.032	1.026	1.016
	(0.007)	(0.042)	(0.041)	(0.045)
First stage: Post-Lasso selection	1.000	1.026	1.023	1.013
	(0.007)	(0.044)	(0.042)	(0.042)
First stage: Lasso	1.007	1.100	1.098	1.042
	(0.007)	(0.045)	(0.045)	(0.045)
First stage: Neural net	1.008	1.215	1.209	1.110
	(0.029)	(0.180)	(0.176)	(0.105)
First stage: Neural net, shallow	1.002	1.069	1.065	1.030
	(0.018)	(0.066)	(0.067)	(0.049)
First stage: Neural net, narrow	1.008	1.213	1.210	1.100
	(0.027)	(0.183)	(0.185)	(0.103)
First stage: Boosted trees	1.008	1.254	1.255	1.121
	(0.007)	(0.041)	(0.039)	(0.047)
First stage: Random forest, CV	1.071	1.562	1.563	1.316
	(0.008)	(0.033)	(0.034)	(0.058)

This table provides the means and standard deviations of the distributions illustrated in Figure 2. Each **column** contains a separate DGP: (a) contains the *low-complexity* DGP with 7 (equally) strong instruments; (b)–(d) contain the three *high-complexity* cases with 100 instruments of mixed strengths. **Rows** differ by estimator. For each DGP-estimator combination, we summarize the estimates for the parameter of interest (β) across 1,000 iterations using a mean and standard deviation (the standard deviation is in parentheses).

Table 2: Simulation results: Decomposing bias components (means from simulation)

		Bias components							
	$(a+b)c$	a	b	c					
Estimator	Bias	$\text{Cov}(\hat{x}, e)$	$\text{Cov}(\hat{x}, u)$	$1/\text{Var}(\hat{x})$	$\text{Var}(\hat{x})$	$\text{Var}(x)$	$\text{Cov}(x, \hat{x})$	$\text{Corr}(x, \hat{x})$	$\text{Cov}(x, u)$
Panel A DGP: *Low-complexity* case									
Naive OLS	0.04	0	1.01	0.04	26.41	26.41	26.41	1	1.01
First stage: OLS	0	0	0.01	0.04	25.41	26.41	25.41	0.98	1.01
Split-sample IV	0	0	0.02	0.04	25.42	26.41	25.42	0.98	1.01
Jackknife IV (JIVE)	0	0	0.01	0.05	20.76	21.74	20.76	0.98	1.01
First stage: PCA	0	0	0.01	0.04	25.41	26.41	25.41	0.98	1.01
First stage: Post-Lasso selection	0	0	0.01	0.04	25.41	26.41	25.41	0.98	1.01
First stage: Lasso selection	0.01	0.17	0.01	0.04	25.08	26.41	25.25	0.98	1.01
First stage: Neural net	0.01	0.1	0.05	0.05	20.52	21.67	20.61	0.98	0.99
First stage: Neural net, narrow	0.01	0.09	0.04	0.05	20.52	21.67	20.61	0.98	0.99
First stage: Neural net, shallow	0	0	0.03	0.05	20.71	21.67	20.71	0.98	0.99
First stage: Boosted trees	0.01	0.01	0.2	0.04	25.51	26.41	25.52	0.98	1.01
First stage: Random forest, CV	0.07	1.08	0.62	0.04	24.01	26.41	25.09	1	1.01
Panel B DGP: *High-complexity* case 1									
Naive OLS	0.22	0	0.15	1.56	0.64	0.64	0.64	1	0.15
First stage: OLS	0.02	0	0.01	1.77	0.56	0.64	0.56	0.94	0.15
Split-sample IV	0.03	0	0.01	1.79	0.56	0.64	0.56	0.93	0.15
Jackknife IV (JIVE)	0	0	0	1.77	0.57	0.65	0.57	0.94	0.15
First stage: PCA	0.02	0	0.01	2.02	0.5	0.64	0.5	0.88	0.15
First stage: Post-Lasso selection	0.01	0	0.01	1.79	0.56	0.64	0.56	0.93	0.15
First stage: Lasso selection	0.04	0.02	0	1.92	0.52	0.64	0.54	0.93	0.15
First stage: Neural net, shallow	0.03	0	0.02	1.77	0.57	0.64	0.57	0.94	0.15
First stage: Neural net, narrow	0.1	0.01	0.05	1.75	0.57	0.64	0.58	0.96	0.15
First stage: Neural net	0.11	0.01	0.06	1.74	0.58	0.64	0.58	0.96	0.15
First stage: Boosted trees	0.12	0.03	0.04	1.91	0.52	0.65	0.55	0.95	0.15
First stage: Random forest, CV	0.32	0.06	0.09	2.04	0.49	0.65	0.56	0.99	0.15
Panel C DGP: *High-complexity* case 2									
Naive OLS	0.33	0	0.35	0.95	1.06	1.06	1.06	1	0.35
First stage: OLS	0.06	0	0.03	1.66	0.6	1.06	0.6	0.76	0.35
Split-sample IV	0.06	0	0.03	1.76	0.57	1.06	0.57	0.73	0.35
Jackknife IV (JIVE)	0	0	0	1.65	0.61	1.06	0.61	0.76	0.35
First stage: PCA	0.03	0	0.02	1.75	0.57	1.06	0.57	0.74	0.35
First stage: Post-Lasso selection	0.02	0	0.01	1.75	0.57	1.06	0.57	0.74	0.35
First stage: Lasso selection	0.1	0.04	0.01	2.11	0.48	1.06	0.52	0.73	0.35
First stage: Neural net	0.21	0.03	0.13	1.49	0.7	1.06	0.73	0.84	0.35
First stage: Neural net, narrow	0.21	0.04	0.12	1.52	0.68	1.06	0.71	0.84	0.35
First stage: Neural net, shallow	0.06	0	0.04	1.63	0.62	1.06	0.62	0.76	0.35
First stage: Boosted trees	0.25	0.07	0.07	1.83	0.55	1.06	0.62	0.81	0.36
First stage: Random forest, CV	0.56	0.16	0.22	1.51	0.66	1.06	0.82	0.98	0.35
Panel D DGP: *High-complexity* case 3									
Naive OLS	0.34	0	0.35	0.95	1.06	1.06	1.06	1	0.35
First stage: OLS	0.06	0	0.04	1.66	0.61	1.06	0.61	0.76	0.35
Split-sample IV	0.06	0	0.04	1.76	0.57	1.06	0.57	0.73	0.35
Jackknife IV (JIVE)	0	0	0	1.64	0.61	1.06	0.61	0.76	0.36
First stage: PCA	0.03	0	0.02	1.76	0.57	1.06	0.57	0.73	0.35
First stage: Post-Lasso selection	0.03	0	0.02	1.74	0.58	1.06	0.58	0.74	0.35
First stage: Lasso selection	0.1	0.04	0.01	2.1	0.48	1.06	0.52	0.73	0.35
First stage: Neural net	0.22	0.03	0.13	1.49	0.69	1.06	0.73	0.84	0.36
First stage: Neural net, narrow	0.21	0.04	0.12	1.53	0.67	1.06	0.71	0.84	0.36
First stage: Neural net, shallow	0.07	0	0.05	1.63	0.62	1.06	0.62	0.76	0.36
First stage: Boosted trees	0.25	0.07	0.07	1.83	0.55	1.06	0.62	0.81	0.36
First stage: Random forest, CV	0.56	0.16	0.22	1.51	0.66	1.06	0.82	0.98	0.35

A cell's value provides the given statistic's mean (**column**) in 1,000 iterations of the given combination of DGP (**Panel**) and estimator (**row**). We omit LIML as it is not a two-stage method and thus does not produce $\hat{x}$.

References

Ackerberg, D. A., and P. J. Devereux, Improved JIVE estimators for overidentified linear models with and without heteroskedasticity, *Review of Economics and Statistics*, *91*(2), 351–362, doi:10.1162/rest.91.2.351, 2009.

Anderson, T. W., and H. Rubin, Estimation of the Parameters of a Single Equation in a Complete System of Stochastic Equations, *The Annals of Mathematical Statistics*, *20*(1), 46–63, doi:10.1214/aoms/1177730090, 1949.

Andrews, I., J. Stock, and L. Sun, Weak instruments in iv regression: Theory and practice, *Annual Review of Economics*, *11*, 727–753, 2019.

Angrist, J., and B. Frandsen, Machine Labor, NBER Working Paper No. 26584, 2020.

Angrist, J., and J.-S. Pischke, *Mostly Harmless Econometrics: An Empiricist's Companion*, Princeton University Press, Princeton, New Jersey, 2009.

Angrist, J. D., and A. B. Krueger, Split-sample instrumental variables estimates of the return to schooling, *Journal of Business & Economic Statistics*, *13*(2), 225–235, doi:10.1080/07350015.1995.10524597, 1995.

Angrist, J. D., and A. B. Krueger, Instrumental variables and the search for identification: From supply and demand to natural experiments, *Journal of Economic Perspectives*, *15*(4), 69–85, doi:10.1257/jep.15.4.69, 2001.

Angrist, J. D., G. W. Imbens, and A. B. Krueger, Jackknife instrumental variables estimation, *Journal of Applied Econometrics*, *14*(1), 57–67, 1999.

Belloni, A., V. Chernozhukov, and C. Hansen, Lasso methods for gaussian instrumental variables models, doi:10.2139/ssrn.1908409, mIT Department of Economics Working Paper No. 11-14, 2011.

Belloni, A., D. Chen, V. Chernozhukov, and C. Hansen, Sparse models and methods for optimal instruments with an application to eminent domain, *Econometrica*, *80*(6), 2369–2429, doi:10.3982/ecta9626, 2012.

Belloni, A., V. Chernozhukov, and C. Hansen, Inference on treatment effects after selection among high-dimensional controls, *The Review of Economic Studies*, *81*(2), 608–650, doi:10.1093/restud/rdt044, 2013.

Bennett, A., N. Kallus, and T. Schnabel, Deep generalized method of moments for instrumental variable analysis, 2020.

Bevis, L. E., and K. Villa, Intergenerational transmission of maternal health: Evidence from cebu, the philippines, *Journal of Human Resources*, pp. 0819–10,372R2, doi:10.3368/jhr.58.1.0819-10372r2, 2020.

Biewen, M., and P. Kugler, Two-stage least squares random forests with an application to angrist and evans (1998), iZA Discussion Paper No. 13613, 2020.

Breiman, L., Arcing the edge, *Tech. rep.*, Technical Report 486, Statistics Department, University of California at Berkeley, 1997.

Breiman, L., Random forests, *Machine Learning*, *45*(1), 5–32, doi:10.1023/a:1010933404324, 2001.

Carrasco, M., and G. Tchuente, Regularized LIML for many instruments, *Journal of Econometrics*, *186*(2), 427–442, doi:10.1016/j.jeconom.2015.02.018, 2015.

Chen, D. L., and S. Yeh, Government expropriation increases economic growth and racial inequality: Evidence from eminent domain, doi:10.2139/ssrn.2977074, tSE Working Paper No. 16-693, 2020.

Chen, J., D. L. Chen, and G. Lewis, Mostly harmless machine learning: Learning optimal instruments in linear iv models, 2020.

Chen, W., X. Chen, C.-T. Hsieh, and Z. Song, A forensic examination of china's national accounts, *Brookings Papers on Economic Activity*, *2019*(1), 77–141, doi:10.1353/eca.2019.0001, 2019.

Chernozhukov, V., C. Hansen, and M. Spindler, Valid post-selection and post-regularization inference: An elementary, general approach, *Annual Review of Economics*, *7*(1), 649–688, doi:10.1146/annurev-economics-012315-015826, 2015.

Chernozhukov, V., D. Chetverikov, M. Demirer, E. Duflo, C. Hansen, W. Newey, and J. Robins, Double/debiased machine learning for treatment and structural parameters, *The Econometrics Journal*, *21*(1), C1–C68, doi:10.1111/ectj.12097, 2018.

Chernozhukov, V., W. Newey, R. Singh, and V. Syrgkanis, Adversarial estimation of riesz representers, 2020.

Derenoncourt, E., Can you move to opportunity? evidence from the great migration, unpublished, 2019.

Farley, B., and W. Clark, Simulation of self-organizing systems by digital computer, *Transactions of the IRE Professional Group on Information Theory*, *4*(4), 76–84, doi:10.1109/tit.1954.1057468, 1954.

Friedman, J. H., Greedy function approximation: A gradient boosting machine, *Annals of Statistics*, pp. 1189–1232, 2001.

Friedman, J. H., Stochastic gradient boosting, *Computational Statistics & Data Analysis*, *38*(4), 367–378, doi:10.1016/s0167-9473(01)00065-2, 2002.

Fuller, W. A., Some properties of a modification of the limited information estimator, *Econometrica*, *45*(4), 939–953, doi:10.2307/1912683, 1977.

Hansen, C., and D. Kozbur, Instrumental variables estimation with many weak instruments using regularized JIVE, *Journal of Econometrics*, *182*(2), 290–308, doi:10.1016/j.jeconom.2014.04.022, 2014.

Hansen, C., J. Hausman, and W. Newey, Estimation with many instrumental variables, *Journal of Business & Economic Statistics*, *26*(4), 398–422, doi:10.1198/073500108000000024, 2008.

Hartford, J., G. Lewis, K. Leyton-Brown, and M. Taddy, Deep IV: A flexible approach for counterfactual prediction, in *Proceedings of the 34th International Conference on Machine Learning, PMLR*, vol. 70, edited by D. Precup and Y. W. Teh, pp. 1414–1423, 2017.

Hastie, T., R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer New York, doi:10.1007/978-0-387-84858-7, 2009.

Ho, T. K., Random decision forests, in *Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1)*, pp. 278–282, IEEE Computer Society, 1995.

Ioffe, S., and C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

James, G., D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*, Springer New York, doi:10.1007/978-1-4614-7138-7, 2013.

Kilbertus, N., M. J. Kusner, and R. Silva, A class of algorithms for general instrumental variable models, 2020.

Kingma, D. P., and J. Ba, Adam: A method for stochastic optimization, 2017.

Lee, D. S., J. McCrary, M. J. Moreira, and J. Porter, Valid t-ratio inference for iv, 2020.

Liao, L., Y.-L. Chen, Z. Yang, B. Dai, Z. Wang, and M. Kolar, Provably efficient neural estimation of structural equation model: An adversarial approach, 2020.

Liu, R., Z. Shang, and G. Cheng, On deep instrumental variables estimate, 2020.

Mason, L., J. Baxter, P. Bartlett, and M. Frean, Boosting algorithms as gradient descent, *Advances in neural information processing systems*, *12*, 512–518, 1999.

McCulloch, W. S., and W. Pitts, A logical calculus of the ideas immanent in nervous activity, *The bulletin of mathematical biophysics*, *5*(4), 115–133, 1943.

Mueller-Smith, M., The criminal and labor market impacts of incarceration, unpublished, 2015.

Mullainathan, S., and J. Spiess, Machine learning: An applied econometric approach, *Journal of Economic Perspectives*, *31*(2), 87–106, doi:10.1257/jep.31.2.87, 2017.

Ng, S., and J. Bai, Selecting instrumental variables in a data rich environment, *Journal of Time Series Econometrics*, *1*(1), doi:10.2202/1941-1928.1014, 2009.

Pearson, K., LIII. on lines and planes of closest fit to systems of points in space, *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *2*(11), 559–572, doi:10.1080/14786440109462720, 1901.

Ribeiro, M. T., S. Singh, and C. Guestrin, "why should i trust you?", in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, doi:10.1145/2939672.2939778, 2016.

Santosa, F., and W. W. Symes, Linear inversion of band-limited reflection seismograms, *SIAM Journal on Scientific and Statistical Computing*, *7*(4), 1307–1330, doi:10.1137/0907087, 1986.

Singh, A., K. Hosanagar, and A. Gandhi, Machine learning instrument variables for causal inference, in *Proceedings of the 21st ACM Conference on Economics and Computation*, ACM, doi:10.1145/3391403.3399466, 2020.

Singh, R., M. Sahani, and A. Gretton, Kernel instrumental variable regression, 2019.

Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, *The Journal of Machine Learning Research*, *15*(1), 1929–1958, 2014.

Storm, H., K. Baylis, and T. Heckelei, Machine learning in agricultural and applied economics, *European Review of Agricultural Economics*, *47*(3), 849–892, doi:10.1093/erae/jbz033, 2019.

Tibshirani, R., Regression shrinkage and selection via the lasso, *Journal of the Royal Statistical Society: Series B (Methodological)*, *58*(1), 267–288, doi:10.1111/j.2517-6161.1996.tb02080.x, 1996.

Turing, A. M., Intelligent machinery, 1948.

Wang, S., Q. Wang, and J. Zhao, Deep neural networks for choice analysis: Extracting complete economic information for interpretation, unpublished. arXiv: 1812.04528, 2019.

Winkelried, D., and R. Smith, Principal components instrumental variable estimation, doi:10.17863/CAM.984, cambridge Working Papers in Economics: *CWPE1119*, 2011.

Wooldridge, J., *Econometric Analysis of Cross Section and Panel Data*, MIT Press, Cambridge, Mass, 2010.

Xu, L., Y. Chen, S. Srinivasan, N. de Freitas, A. Doucet, and A. Gretton, Learning deep features in instrumental variable regression, 2020.

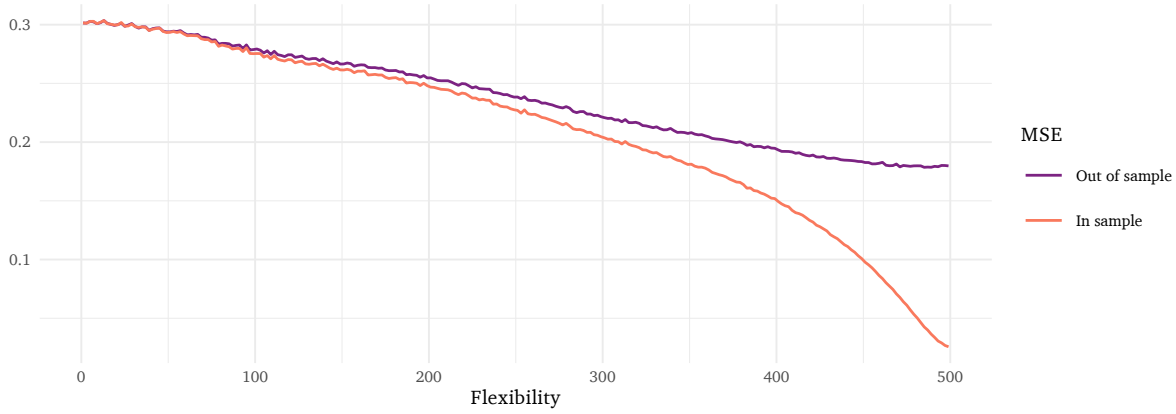
Zhao, S., D. Witten, and A. Shojaie, In defense of the indefensible: A very naive approach to high-dimensional inference, arXiv: 1705.05543, 2020.

Appendix A

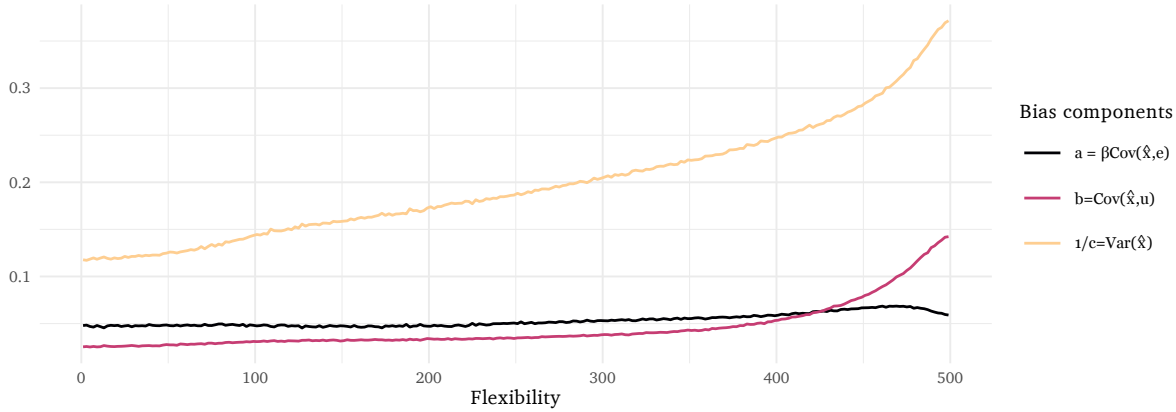
A.1 Appendix: Figures

Figure A1: Predictions *vs.* estimation: Comparing cross-validated prediction performance with bias in random-forest-based 2SLS

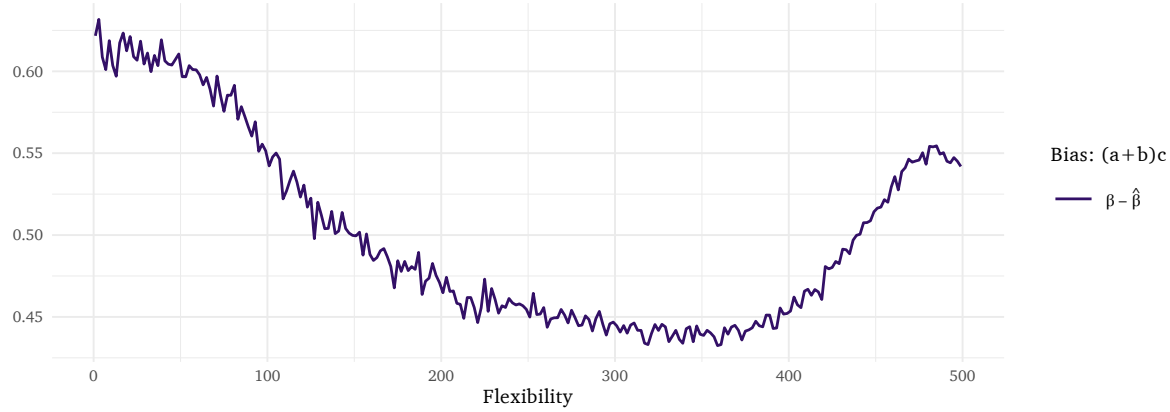
(i) In- and out-of-sample MSE for predictions of x



(ii) The three 'bias components' in estimating β



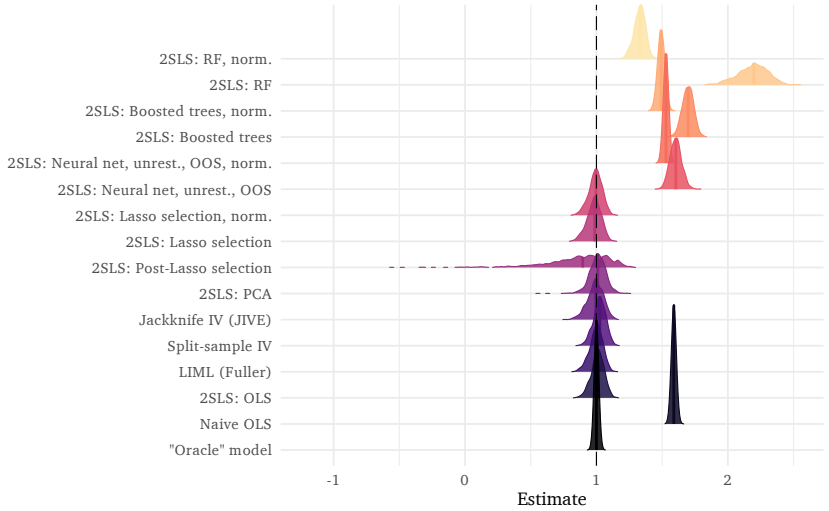
(iii) Bias in $\hat{\beta}$ as a function of model flexibility ($\beta_1 = 1$)



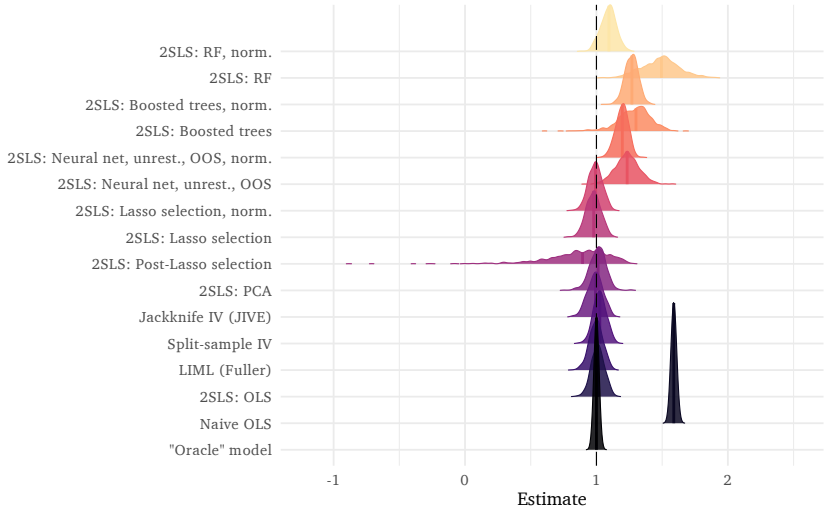
Notes

Figure A2: Distributions of estimates with "exclusion-restriction violations" from k-term interactions ('low-complexity case)

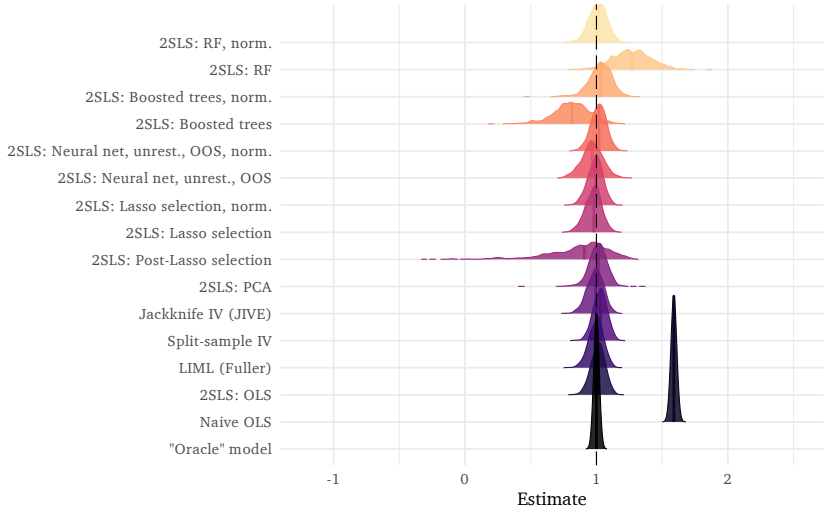
(i) Two-term 'exclusion-restriction violation': $x_1 \times x_2$



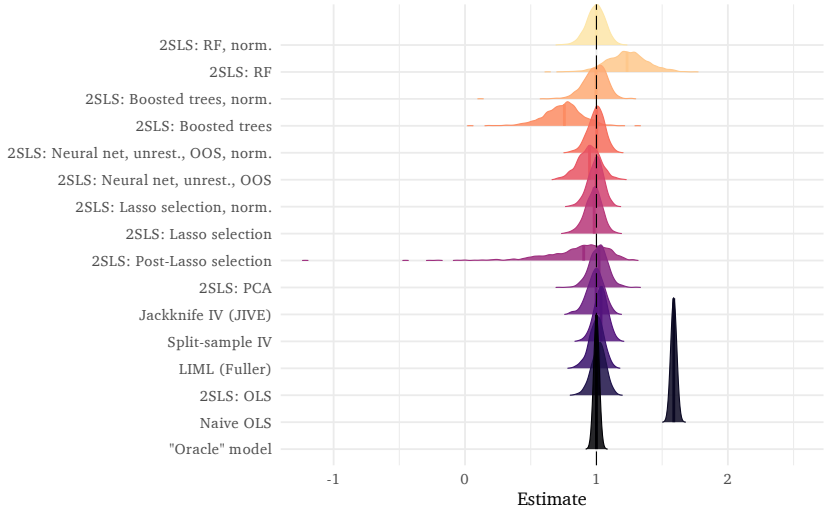
(ii) Three-term 'exclusion-restriction violation': $x_1 \times x_2 \times x_3$



(iii) Four-term 'exclusion-restriction violation': $x_1 \times x_2 \times x_3 \times x_4$



(iv) Five-term 'exclusion-restriction violation': $x_1 \times x_2 \times x_3 \times x_4 \times x_5$



This figure illustrates the densities of the estimates portrayed in Figure 3. Post-lasso's high-variance in this experiment come from using the so-called plug-in lambda value that prevents overselection of poor-performing instruments. To calculate this plug-in penalty, two parameters are user-chosen: c and γ and multiple theoretical values are used. We used the suggested values as detailed in *Belloni et al.* (2012) and *Belloni et al.* (2013). This lambda plug-in value is calculated to be equal to $(1.1 * 2)\sqrt{N} * \Phi(\frac{1-(\gamma)}{2*nz})$ and $\gamma \equiv \frac{.1}{ln(N \vee nz)}$. In our case, the number of instruments $nz = 7$, the number of observations $N = 1000$, and $\Phi \sim$ *Quantile Normal*. This results in $\lambda \approx 36$. By using the suggested higher levels of λ, post-lasso can result in a higher chance of under-fit, with the extreme example being a second-stage regression model of the form $f(X|Z) = 0$. Given our variables are independent and relatively strong, regularization is highly unlikely to improve fit, and large lambda may result in under-specification for the first stage.

A.2 Appendix: Math

A.2.1 Wedge a covariance and correlation

Recall that $e = x - \hat{x}$, *i.e.*, e is the first-stage prediction's residual.

$$\begin{aligned}
\text{Corr}(\hat{x}, e) &= \frac{\text{Cov}(\hat{x}, e)}{\sigma_{\hat{x}}\sigma_e} \\
&= \frac{\text{Cov}(\hat{x}, x)}{\sigma_{\hat{x}}\sigma_e} - \frac{\text{Var}(\hat{x})}{\sigma_{\hat{x}}\sigma_e} \\
&= \frac{\text{Cov}(\hat{x}, x)}{\sigma_{\hat{x}}\sigma_e}\frac{\sigma_x}{\sigma_x} - \frac{\sigma_{\hat{x}}}{\sigma_e} \\
&= \frac{\text{Corr}(\hat{x}, x)\,\sigma_x}{\sigma_e} - \frac{\sigma_{\hat{x}}}{\sigma_e} \\
&= \sigma_e^{-1}\Big(\text{Corr}(\hat{x}, x)\,\sigma_x - \sigma_{\hat{x}}\Big) \qquad \text{(A1)}
\end{aligned}$$

A.3 Appendix: MLP/Neural cross-validation procedure

Unlike many of the other methods explored in this paper, MLP (Multi-Layer Perceptrons) are difficult to cross-validate in a consistent way. This is for three main reasons.

The first reason is due to one of Neural methods' advantages for prediction problems: namely that they are highly adaptable to many different problem spaces, varying both in more traditional hyper-parameters such as learning-rate and neural network width, but also in much more nuanced choices such as optimization method or input structure. 'Neural Networks'; despite the term's usage in many settings, is actually less of a single model and more a label placed on an entire class of iteratively-optimized models. The work's aim has been to use 'off-the-shelf' machine learning methods to understand what empirical concerns exist when placing these models naively in an otherwise-recognizable econometric instrumental variables setting, but for a neural network, the off-the-shelf model is highly dependent on the problem at hand. Unfortunately, this advantage of MLPs and other Neural Networks makes a full grid-search of the hyper-parameter space intractable. This requires us to restrict the grid-space somewhat to create a tractable solution, while allowing the model a good shot at choosing the 'correct' specification. This restriction potentially handicaps the neural model's flexibility, and may produce higher average out-of-sample loss than the full set of model specifications could potentially produce.

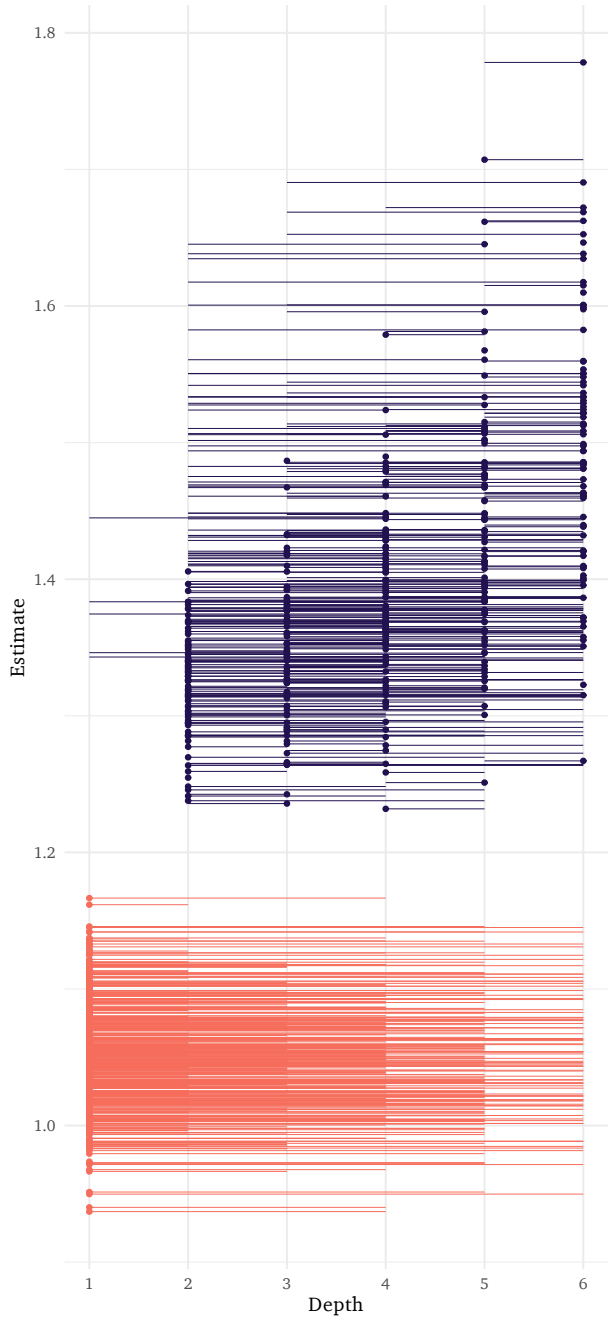
Second, Neural Networks are computationally expensive to train - each combination of hyperparameters must be trained separately over many iterations, and for most optimization procedures it is useful to utilize different re-orderings of the data-set to reach a satisfactory loss-minimizing point. Even for less complex data such as ours with relatively few observations, cross-validating even 100 hyperparameter combinations over 1000 synthetic datasets leads to prohibitively long training periods given our computational resources.

Last, neural networks are sensitive to their initialization point, which is not the case for any of our other methods included in this analysis. Unlike most other methods, neural-class models can find values for one of a number of potential loss-minimizing local optima for its parameters, and while those local optima can perform similarly well, they do not necessarily produce identical predictions and can fail on different subsets of data in different ways.

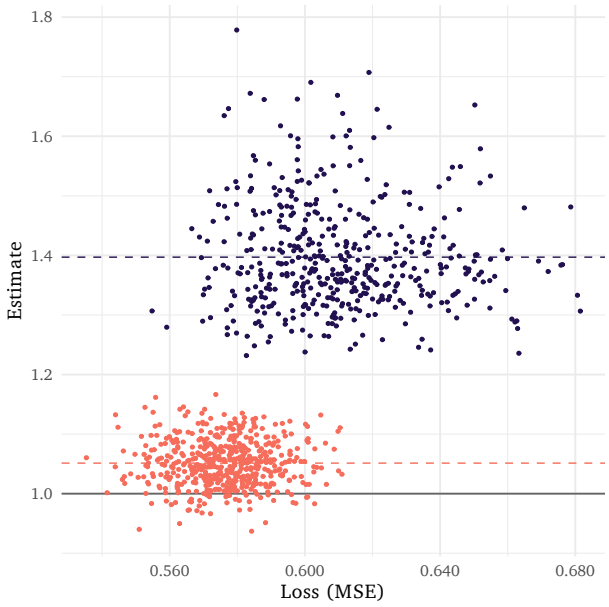
These differences make analyzing how cross-validating a neural network dictates hyper-parameters given a reasonable loss function more interesting because the choices a 5-fold cross-validation approach might make are indicative of how the most flexible model chooses a specification given different search spaces.

Figure A3: Explaining unrestricted/narrow neural networks' bimodal distributions of $\hat{\beta}$

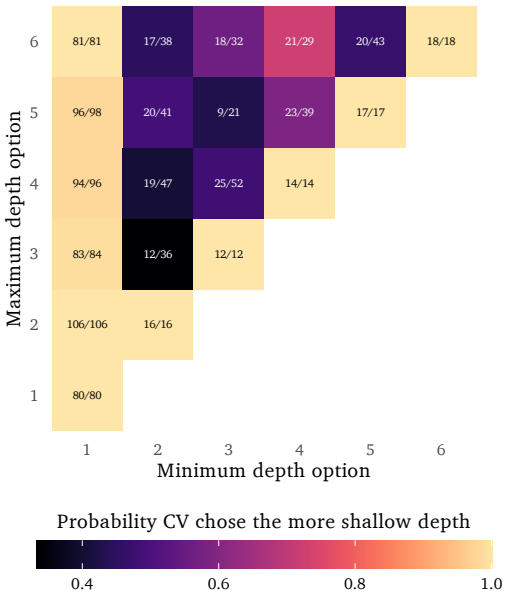
(i) Comparing bias in $\hat{\beta}$: Approximately linear (no hidden layers) *vs.* 'deeper' neural networks



(ii) Comparing bias in $\hat{\beta}$ and out-of-sample loss: Approximately linear *vs.* 'deeper' models



(iii) Cross validation's likelihood of choosing more shallow models when choosing between two options



The y axis of Panel **a** depicts the second-stage estimate $\hat{\beta}$, and the x axis represents the depths of the neural networks cross validated in each of the 1,000 iterations. "Depth 1" implies no hidden layers—directly linking the input and output (approximating linear regression). Horizontal line segments in **a** connect the two possible depths that the model chose between. The solid dot marks the chosen depth (by cross validation). In **a** and **b** we color the two subsets separately to illustrate the source of the bimodal distributions in Figure 2. Panel **c** gives the probability that a model chose the more shallow (less deep) model for each combination of choices (probabilities are equal to the fractions in the cells). For instance, the top-left corner tells us that when facing a choice between a depth-1 model and a depth-6 model, CV chose the depth-1 model 81 out of 81 times. Each of these figures uses the results from 1,000 iterations of neural-net based 2SLS *high-complexity case 2* (Panel C of Figure 2iii; 100 mixed-strength instruments with strength decreasing from z_1) with no restrictions on the hyperparameter space (*unrestricted* in tables 1–2). The results are very similar for the other high-complexity DGPs.

For our cross-validation, we use a 5-fold cross-validation procedure to match our other methods, and use mean-squared error as our loss function. We fix a few hyperparameters in place—we use no regularization on the weights, and use an "Adam" optimizer with it's out-of-box/off-the-shelf learning rate of .001. We trained all models over 40 epochs, and used a batch-size of 10 observations. One unusual step we take is to introduce a leaky rectified linear activation function (ReLU) activation function to connect hidden layers. This was done to prevent the model from suffering from "dying weights" which is when parameters accidentally force a large number of activations to inappropriately "ignore" activations due to $a(x) = 0$ for all or many values of x.[30]

We began by creating three separate hyper-parameter search spaces distinguished by maximum-allowable width and depth. The 'shallow/wide' neural networks are allowed to choose from hidden-layer representations that are 16, 32, 64, 256, and 512 nodes in hidden layer width respectively. This model is then restricted to contain at most a single hidden layer, but is also allowed to choose from a model that maps inputs to outputs directly, using a linear activation function. This functional form is sensible given x is linear in z for our DGPs. Excluding the 0-hidden layer case would prevent the cross-validation procedure from finding the easiest approximation for a linear functional form. The cross-validation procedure allows differences in regulation by choosing between a dropout rate of .1 or .2.

The 'narrow/deep' neural network is instead allowed to choose from a representation with 2,3,4 or 5 hidden layers each with a number of nodes (width) equal to 16,32, or 64. This model, too, is allowed to choose from the simple linear mapping of inputs to output, for the same reasons as above.

The last search space, referred to as the 'unrestricted' neural network is allowed to choose from any combination of the hyperparameters offered to the narrow or shallow networks- from 0 through 5 hidden layers and using the full complement of widths.

To get a sense of how these search-spaces choose models on average, these three search spaces were used to cross-validate over our first 25 datasets, from which we generated a full list of 125 folds. These cross-dataset folds were then used to find, for each search space, the average out of sample MSE across all 125 folds.

From the set of available models available to every search space and for each iteration, we chose two models at random weighted by their average out of sample MSE. These probabilities were chosen using the weighted upper-tail normal CDF, normalized such that all weights for a given search space sum to one. Formally, where i is a given set of hyperparameters, j is a search-space and μ_j is the mean out of sample MSE for a given search space:

$$p_{i,j}^{chosen} = \int_{z_{i,j}^{mse}}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{z}{2}} dz \,, \tag{A2}$$

$$z_{i,j}^{mse} := \frac{mse_{i,j} - \mu_j}{se(mse_j)} \,. \tag{A3}$$

For each iteration and using the probabilities above, two models are chosen at random for each search space, and then cross-validated again using a 5-fold procedure. The "winning" model is chosen by lowest out-of-sample MSE, and is used to predict $\hat{x}$ for the first stage.

For a visual explanation and overview of the results from the selection method, see fig. A3.

[30] ReLU was tried initially, however using ReLU on our data seemingly led to a complete shutdown of the predictive power when we attempted it on our weaker-instrument setting. Even using dropout, the MLP's performance was poor in predicting out of sample. We cannot be certain that the dying weights problem was the cause of this issue, but adding a slight negative slope for activations less than 0 seems to have mitigated the behavior.

A.3.1 Low-bias methods

Because machine learning algorithms are designed to minimize loss (maximizing fit), the fact that $\text{Cov}(\hat{x}, e) \neq 0$ is partially by design. To see this fact, consider any prediction method that minimizes mean-squared error (MSE)—conditional on the training data $\{x, \mathbf{z}\}$:

$$\text{MSE}(\hat{x}, x|x, \mathbf{z}) = \underbrace{\Big(x - \text{E}[\hat{x} \mid x, \mathbf{z}]\Big)^2}_{\left(\text{Bias of } x \text{ for } \hat{x}\right)^2} + \underbrace{\text{E}\Big[\ (\hat{x} - \text{E}[\hat{x} \mid x, \mathbf{z}])^2 \ \Big|\ x, \mathbf{z}\ \Big]}_{\text{Cond. Var}(\hat{x})} + \underbrace{\text{E}\Big[\ \varepsilon^2 \ \Big|\ x, \mathbf{z}\ \Big]}_{\text{Cond. Var}(\varepsilon)} \tag{A4}$$

where ε is the irreducible error—the unknowable disturbance from the DGP of x, *i.e.*, $x = f(\mathbf{z}) + \varepsilon$.[31]

Equation (A4) highlights that in an MSE-minimization problem, $\hat{x}$ is the only component of MSE that a learning algorithm can change (x, $\mathbf{z}$, and ε are all data dependent). This fact leads to the widely discussed variance-bias tradeoff: an arbitrary estimator will generally face a negotiate between low-variance predictions and low-bias predictions. Many traditional econometric estimators result from prioritizing zero bias and then selecting the minimum-variance estimator from this class of unbiased estimators. As (A4) points out: these estimators could reduce their out-of-sample MSE by *accepting* some bias and reducing variance. This tradeoff is at the heart of the prediction improvement many out-of-the-box algorithms offer relative to plain OLS.

A.4 Appendix: Neural approaches to measuring causal response

Neural networks and their offspring offer just such a route to explore data that falls outside of the traditional bounds, whether that is to use transformers for text data, or convolutional neural networks (CNNs) for imaging. However, many of the same problems apply to these tools as were outlined in the paper. In order to make full use of them in a two stage approach, the same stringent restrictions are required to generate meaningful and unbiased coefficients in a two-stage framework. Indeed, when running a cross-validated feed-forward network to produce a meaningful first stage estimation with our high-complexity data most simulations with any hidden layers simply reproduced an approximation of β close to that of naive OLS.[32] There is a burgeoning field of research in machine learning that strives to understand IV problems under less-parametric (though generally still somewhat parametric) causal structures and these methods are seeing success in both simulated and real-world data. The downside to using these powerful methods is that they require a new framework in which to understand them, and make interpretation of treatment effects more challenging.

The first of the recent batch of machine learning instrumental variables papers is referred to as "Deep IV" *Hartford et al.* (2017). The authors throw away the linear functional form for $x = f(z, u)$ in the "first stage," but assume linearly additive confounding variables and learn the causal structure with a two-part one-pass neural network model. The authors do this by recasting the econometric approach to instrumental variables into two interlinked problem spaces - estimating the conditional distribution $g(x|z)$ and then using the approximation of x given z to predict y. This creates difficulties because such methods produce good counterfactual predictions, but have a harder time matching the clean interpretable causal effect of X on y when compared to traditional econometric approaches. Further, because of the flexibility in functional form, models of this category tend to have more trouble outside of $supp(z_{train})$ or $supp(g(x|z_{train}))$ that are observed in a training sample - and it's difficult to apply a post-analysis structure to such a model to gather understanding on counterfactuals where z_{test} is considerably different than z_{train}. Further, many other methods have been created and can be used to estimate instrument-identified causal effects using a similar semi-parametric two-stage function that can identify complex functional forms in either first or second stages *Bennett et al.* (2020) and *Xu, Chen,*

[31] These expectations are conditional on the given dataset; ε and $\hat{x}$ are conditionally independent by definition. The expectation term is conditional on data observed, so for simplicity, the term $E_D(\hat{x}(z; D))$ will simply be referred to as $\hat{x}$.

[32] See Appendix Section A.3 for full details of the MLP methods.

Srinivasan, de Freitas, Doucet, and Gretton (2020) and improve on edge-of-support marginal effects.[33] Both of these papers and Deep IV are able to produce causal inferences using images as instruments—something that a regression would not be able to meaningfully do without some form of pre-model dimensionality reduction.

Neural approaches to causal inference are also not limited to use semi-parametric structural forms for heterogenous treatment effects. *Kilbertus et al.* (2020) created a neural network to identify the total set of conditional causal effects given a fully non-parametric instrumental variable analysis. (With the very reasonable assumption placed on the function of unobserved noise that it does not feature infinite discontinuities, for example.)

In spite of the massive technical improvements these models have made, trying to extract a beta-equivalent from the existing models is difficult, though interpretable machine learning methods do exist. Unfortunately, extracting meaningful information using prediction-explanation methods such as *Ribeiro, Singh, and Guestrin* (2016) about how a result is produced, or to infer what kind of economic information can be gathered from the weights within a model, neural networks are hard to interpret *Wang, Wang, and Zhao* (2019). This makes comparing such models to traditional 2SLS or econometric approaches for ATE approximation difficult—and produces complications in choosing benchmarks as to how to evaluate them.

[33] The methodology contained in *Xu et al.* (2020) is particularly useful, because it does not predict X directly, but rather applies Neural Networks to the task of learning polynomial forms to pass through first and second stages in a 2SLS (with an L2 penalty) and may mostly avoid components a and b as described earlier.