WORKING PAPER · NO. 2020-01

ivmte: An R Package for Implementing Marginal Treatment Effect Methods

Joshua Shea and Alexander Torgovitsky

JANUARY 2020

Becker Friedman Institute
FOR ECONOMICS AT UCHICAGO

5757 S. University Ave.
Chicago, IL 60637
Main: 773.702.5599
bfi.uchicago.edu

ivmte: An R Package for Implementing Marginal Treatment Effect Methods

by Joshua Shea and Alexander Torgovitsky

Abstract Instrumental variable (IV) strategies are widely used to estimate causal effects in economics, political science, epidemiology, and many other fields. When there is unobserved heterogeneity in causal effects, standard linear IV estimators only represent effects for complier subpopulations (Imbens and Angrist, 1994). Marginal treatment effect (MTE) methods (Heckman and Vytlacil, 1999, 2005) allow researchers to use additional assumptions to extrapolate beyond these subpopulations. In this paper, we introduce the **ivmte** package (Shea and Torgovitsky, 2019), which provides a flexible framework for implementing MTE methods in both point identified and partially identified settings.

Introduction

A central empirical task in economics and other fields is to determine the effect (the *causal* effect) of one variable on another. This is often complicated by the fact that the effecting variable (the treatment) is not only not randomly assigned, it is *chosen* purposely by an economic agent with information unavailable to the researcher. For example, in the application discussed later, the treatment is the number of children a family decides to have, and the empirical question is the effect that bearing more children has on the mother's labor force participation. Since having a child and working are joint decisions a family makes using their own private information, strategies such as propensity score matching are unlikely to eliminate systematic unobserved differences between families with more children and those with fewer children. Different empirical strategies are needed to credibly identify a causal effect.

One extremely popular strategy is to use an instrumental variable (IV) model. An IV (or *instrument*) is an observed variable that is correlated with the treatment variable, but uncorrelated with confounding unobservable differences. A well-known example of an instrument for fertility is the same-sex instrument introduced by Angrist and Evans (1998). This instrument is a binary variable that is equal to 1 if a family's first two children had the same sex (female-female or male-male) and is 0 otherwise.[1] The key assumption of an IV model is that the sex of the second child is as-good-as-randomly assigned, and therefore independent of any confounding unobservable differences across families, while still impacting a family's decision to have a third child due to a preference for having both a male and female child. The intuition is that by comparing the work decisions of families whose first two children were the same sex to families whose children were mixed sex, one picks up only the differences that are caused by the decision to have additional children.

While IV strategies have been widely discussed for many decades (Stock and Trebbi, 2003), it wasn't until the 1980s that researchers started to focus on IV models with unobserved heterogeneity in treatment effects.[2] In an influential paper, Imbens and Angrist (1994) showed that under a simple model of treatment choice behavior, standard linear IV estimators represent the causal effect for a specific well-defined subpopulation they described as *the compliers*. The compliers are the individuals whose treatment choice would have been different had their as-good-as-randomly-assigned instrument been different. In the fertility application, they are the families who would have had a third child if and only if their first two children had the same sex.

An important implication is that standard IV estimators represent (at best) causal effects only for specific subpopulations. These subpopulations might not be relevant for a researcher's empirical question. For example, suppose that a researcher is estimating the effect of fertility on labor supply as part of a broader investigation into the economic impacts of expanding child tax credits. There is no reason to expect that those impacted by a tax change will be similar to the same-sex compliers. Developing a coherent answer to this problem requires a model for extrapolating from the same-sex compliers to another subpopulation.

In a series of papers, Heckman and Vytlacil (1999, 2005, 2007a,b) developed the concept of the *marginal treatment effect* (MTE) and showed how it can be used to model this type of extrapolation nonparametrically. Carneiro et al. (2011) and Brinch et al. (2012, 2017) showed how to apply their idea to identify and estimate semiparametric MTE models. Mogstad et al. (2018) extended these approaches to provide a general moment-based framework that also accommodates partial identification (bounds) in cases when the researcher's assumptions are not strong enough (or the data is not rich enough) to

[1] Using the same-sex instrument requires restricting the analysis to families with two or more children.

[2] An early example is Heckman (1976). See also Heckman and Robb (1985), Björklund and Moffitt (1987) and Manski (1990).

pin down a unique conclusion.

In this paper, we describe the R package **ivmte**, which implements MTE methods using the Mogstad et al. (2018) framework. The package provides a flexible environment for using IV methods for rigorous policy evaluation in the presence of rich unobserved heterogeneity.

Model and theoretical background

The **ivmte** package is designed for the popular binary treatment IV model discussed by Imbens and Angrist (1994) and Heckman and Vytlacil (2005). Here, we provide a brief introduction sufficient for understanding the purpose and usage of **ivmte**. For more detail, see Mogstad and Torgovitsky (2018).

The model is about the impact of a binary treatment $D_i \in \{0,1\}$ on individual i's observed outcome variable, Y_i. Let $Y_i(0)$ and $Y_i(1)$ denote the unobserved potential outcomes for Y_i if individual i had received $D_i = 0$ or 1, respectively, so that $Y_i = D_i Y_i(1) + (1 - D_i)Y_i(0)$. The researcher is interested in features of the distribution of the causal effect, $Y_i(1) - Y_i(0)$. The researcher has access to some observable covariates, X_i, but they are concerned that D_i is still dependent with $Y_i(0)$ or $Y_i(1)$ even after conditioning on X_i, so that the unconfoundedness (selection-on-observables) assumption (see e.g. Barnow et al., 1980; Rosenbaum and Rubin, 1983; Heckman et al., 1996) does not hold.

However, the researcher also has access to an instrumental variable, Z_i. The instrument influences individual i's treatment choice with $D_i(z)$ denoting their unobserved potential treatment choice if Z_i were set to z. Their observed treatment choice is related to these potential choices via $D_i = \sum_{z \in \mathcal{Z}} \mathbb{1}[Z_i = z] D_i(z)$, where $\mathcal{Z}$ is the support of Z_i. In contrast to D_i, the instrument is assumed to be as-good-as-randomly-assigned, conditional on X_i, in the sense that Z_i is independent of $(Y_i(0), Y_i(1), \{D_i(z)\}_{z \in \mathcal{Z}})$, conditional on X_i.

Imbens and Angrist (1994) introduced an additional assumption that they described as *monotonicity*. This says that for any pair of instrument values z and z', either $D_i(z) \geq D_i(z')$ for all individuals i, or else $D_i(z') \geq D_i(z)$ for all individuals i. That is, a shift from z to z' either pushes every individual towards treatment, or else pushes every individual away from treatment. Whichever direction holds, the monotonicity condition maintains that there are no individuals who deviate from this ordering, a requirement sometimes described as "no defiers."[3]

Vytlacil (2002) showed that the monotonicity condition is *equivalent* to the latent variable selection model

$$D = \mathbb{1}[U_i \leq p(X_i, Z_i)], \tag{1}$$

where U_i is a continuously distributed unobservable propensity to take treatment, and $p(x,z) \equiv \mathbb{P}[D_i = 1 | X_i = x, Z_i = z]$ is the *propensity score*. The latent variable U_i is independent of Z_i, conditional on X_i, and customarily normalized to be uniformly distributed on $[0,1]$.[4] It encodes information about individual i's potential treatment choices $\{D_i(z)\}_{z \in \mathcal{Z}}$. The fact that this information can be aggregated into a single unobservable is due to the monotonicity condition, which creates a one-dimensional ordering of individuals according to their latent propensity to take treatment. Here, we have written the selection model in its conventional (but perhaps counter-intuitive) form with smaller values of U_i representing individuals who are *more likely* to take treatment.

The advantage of the latent variable model (1) is that it facilitates modeling unobserved heterogeneity in the effect of D_i on Y_i. The key objects for this purpose are the marginal treatment response (MTR) functions

$$m_0(u,x) \equiv \mathbb{E}[Y_i(0) | U_i = u, X_i = x] \quad \text{and} \quad m_1(u,x) \equiv \mathbb{E}[Y_i(1) | U_i = u, X_i = x]. \tag{2}$$

The MTR functions describe how expected treated and untreated outcomes vary conditional on both observed covariates, X_i, and the unobserved latent propensity to take treatment, U_i. The marginal treatment effect (MTE) of Heckman and Vytlacil (1999, 2005, 2007a,b) is the difference of the two MTR functions: $m_1(u,x) - m_0(u,x)$. For example, if the MTE is declining in u, then individuals who are less likely to choose treatment experience smaller treatment effects than those who are more likely to choose treatment. Thus, the MTE describes the idea of selection-on-*unobservables*, where the unobservable in question is an individual's latent propensity to take treatment.

Many interesting parameters can be constructed by taking weighted averages of the MTR functions.

[3] Despite the name "monotonicity," the condition would be more accurately described as "uniformity," since it restricts heterogeneity in how the instrument impacts treatment choice (Heckman et al., 2006; Mogstad et al., 2019).

[4] See Heckman and Vytlacil (2005), Matzkin (2007), or Mogstad and Torgovitsky (2018) for a detailed discussion of the normalization.

For example, the average treatment effect (ATE) can be written as

$$\mathbb{E}[Y_i(1) - Y_i(0)] = \mathbb{E}[m_1(U_i, X_i) - m_0(U_i, X_i)] = \mathbb{E}\left[\int_0^1 m_1(u, X_i) - m_0(u, X_i)\, du\right], \tag{3}$$

since the distribution of U_i is uniform and independent of X_i. Similarly (but less obviously), the average treatment effect on the treated (ATT) can be written as

$$\mathbb{E}[Y_i(1) - Y_i(0)|D_i = 1] = \mathbb{E}\left[\int_0^1 [m_1(u, X_i) - m_0(u, X_i)] \times \frac{\mathbb{1}[u \leq p(X_i, Z_i)]}{\mathbb{P}[D_i = 1]}\, du\right], \tag{4}$$

see e.g. Heckman and Vytlacil (2005). Following Mogstad et al. (2018), we view both (3) and (4) as special cases of the general form

$$\beta^\star \equiv \mathbb{E}\left[\int_0^1 m_0(u, X_i)\omega_0^\star(u, X_i, Z_i)\, du\right] + \mathbb{E}\left[\int_0^1 m_1(u, X_i)\omega_1^\star(u, X_i, Z_i)\, du\right], \tag{5}$$

where $\omega_0^\star$ and $\omega_1^\star$ are weights that are either known to the researcher (as in (3)) or directly identified in the data (as in (4)). Those authors describe $\beta^\star$ as the *target parameter*: It is the quantity that the researcher seeks to know. Heckman and Vytlacil (2005), Mogstad et al. (2018), and Mogstad and Torgovitsky (2018) provide extensive discussions and many examples of target parameters, along with their weighting functions, $\omega^\star$.

A pair of MTR functions determines unique values for unobservable quantities like the target parameter, $\beta^\star$. At the same time, a pair of MTR functions also determines unique values for *observable* quantities. For example, the cross-moment between Y_i and Z_i can be written as

$$\mathbb{E}[Y_i Z_i] = \mathbb{E}\left[\int_0^1 m_0(u, X_i) Z_i \mathbb{1}[u > p(X_i, Z_i)]\, du\right] + \mathbb{E}\left[\int_0^1 m_1(u, X_i) Z_i \mathbb{1}[u \leq p(X_i, Z_i)]\, du\right]. \tag{6}$$

The right-hand side of (6) is similar to the right-hand side of (5), except with different weights. The key difference is the left-hand side: Whereas the target parameter on the left-hand side of (5) cannot be directly estimated from the data—it is the object we *want to know*—the left-hand side of (6) can easily be estimated by taking a sample mean. Thus, equations like (6) provide a means for *fitting* the MTR functions, and thus for inferring the target parameter, $\beta^\star$.

More generally, Mogstad et al. (2018) show that for any known or identified function s, the cross-moment $\mathbb{E}[Y_i s(D_i, X_i, Z_i)]$ can be written in a form like (6) and therefore used to fit the MTR functions. The case shown in (6) corresponds to taking $s(d, x, z) = z$. They refer to these moments as *IV–like estimands* with the specification of s called an *IV–like specification*. As they show, the estimands for common classes of estimators, such as ordinary least squares (OLS) and two-stage least squares (TSLS), can all be written as IV–like estimands for particular choices of s.

Estimation

Mogstad et al. (2018) observed that the right-hand sides of both (5) and (6) are linear functions of the unknown MTR functions (m_0, m_1). This facilitates straightforward estimation of (m_0, m_1) and thus $\beta^\star$ through linear-in-parameters specifications. In particular, let $\mathcal{M}$ denote the parameter space for $m \equiv (m_0, m_1)$, and suppose that every $m \in \mathcal{M}$ satisfies

$$m_d(u, x) = \sum_{k=1}^{K_d} \theta_{dk} b_{dk}(u, x) \quad \text{for some } (\theta_{d1}, \ldots, \theta_{dK_d}) \in \mathbb{R}^{K_d} \text{ and each } d = 0, 1, \tag{7}$$

where b_{dk} are known *basis functions*. That is, each MTR function is assumed to live in a class of functions formed by taking linear combinations of the basis functions. This reduces the dimension of the two functions $m \equiv (m_0, m_1)$ to a real vector $\theta \equiv (\theta_1, \theta_2)$ of dimension $K_0 + K_1$. Let $\Theta \subseteq \mathbb{R}^{K_0 + K_1}$ denote the implied parameter space for θ.

Linear-in-parameters specifications like (7) are commonplace in statistical models. For example one could specify

$$m_0(u, x) = \theta_{01} + \theta_{02}u + \theta_{03}x + \theta_{04}ux \quad \text{and} \quad m_1(u, x) = \theta_{11} + \theta_{12}u + \theta_{13}u^2 + \theta_{14}x + \theta_{15}x^2 \tag{8}$$

which corresponds to $K_0 + K_1 = 4 + 5 = 9$ parameters. The assumption used in the **ivmte** package is that $\mathcal{M}$ contains only MTR functions that are either polynomials or polynomial B–splines in u. This is certainly a special case of (7), but one that is popular both as a parametric restriction (e.g. a polynomial, like (8)) and as an approximating basis for nonparametric sieve estimation (e.g. Chen, 2007). The

advantage of restricting attention to this case is that the integral over u in (6) can then be computed analytically.

The benefit of using a linear-in-parameters specification is that it preserves the linearity in weighting expressions like (5) and (6). In particular, if m satisfies (7), then

$$\beta^\star = \theta' \Gamma^\star \quad \text{where} \quad \Gamma^\star_{dk} \equiv \mathbb{E}\left[\int_0^1 b_{dk}(u, X_i)\omega^\star_d(u, X_i, Z_i)\, du\right],$$

and $\quad \mathbb{E}[Y_i s(D_i, X_i, Z_i)] = \theta' \Gamma_s$

$$\text{where} \quad \Gamma_{sdk} \equiv \mathbb{E}\left[s(d, X_i, Z_i)\int_0^1 b_{dk}(u, X_i)\mathbb{1}[u \leq p(X_i, Z_i)]^d \mathbb{1}[u > p(X_i, Z_i)]^{1-d}\, du\right].$$

Given a collection $\mathcal{S}$ of IV–like specifications s, we can use the latter expression to form a generalized method of moments ("GMM"; Hansen, 1982) estimator of θ:

$$\hat{\theta} = \text{argmin}_\theta \left(\hat{\beta}_\mathcal{S} - \hat{\Gamma}_\mathcal{S}\theta\right)' \hat{\Omega} \left(\hat{\beta}_\mathcal{S} - \hat{\Gamma}_\mathcal{S}\theta\right) \tag{9}$$

where $\hat{\beta}_\mathcal{S}$ is a vector of sample analogs of $\mathbb{E}[Y_i s(D_i, X_i, Z_i)]$ for each $s \in \mathcal{S}$, $\hat{\Gamma}_\mathcal{S}$ contains sample analogs of Γ_{sdk}, arranged into $|\mathcal{S}|$ rows, and $\hat{\Omega}$ is a weighting matrix.[5] We then form an estimator of the target parameter as

$$\hat{\beta}^\star \equiv \hat{\theta}' \hat{\Gamma}^\star, \tag{10}$$

where $\hat{\Gamma}^\star$ is a vector of sample analogs of $\Gamma^\star_{dk}$. Assuming that a unique solution to (9) exists, it can be solved for analytically and then simply plugged in to (10).

However, in many interesting cases, the solution to (9) will not be unique, for example if the dimension of θ is larger than $|\mathcal{S}|$. This happens naturally in IV models when the support of the instrument is small relative to the dimension of θ (Mogstad and Torgovitsky, 2018). This *does not* mean that the model and data have no information about $\beta^\star$. Rather, it means that there is more than one value of $\beta^\star$ that is consistent with both the model and data, a situation referred to as *partial identification* (see e.g. Manski, 2003; Tamer, 2010). For such cases, the **ivmte** package allows a user to construct bounds on $\beta^\star$. As we show in the empirical illustration below, it also gives the user sufficient flexibility in adjusting the target parameter ($\beta^\star$) and assumptions ($\mathcal{M}$) to make the bounds as narrow as they desire.

The bounds are computed in two steps. First, the following ℓ_1–norm analog of (9) is solved

$$\hat{Q} \equiv \min_{\theta \in \Theta} \sum_{s \in \mathcal{S}} |\hat{\beta}_s - \theta' \hat{\Gamma}_s|. \tag{11}$$

Then, a lower bound on $\beta^\star$ is estimated by solving

$$\hat{\beta}^\star_{\text{lb}} \equiv \min_{\theta \in \Theta} \theta' \hat{\Gamma}^\star \quad \text{subject to} \quad \sum_{s \in \mathcal{S}} |\hat{\beta}_s - \theta' \hat{\Gamma}_s| \leq \hat{Q}(1 + \tau), \tag{12}$$

while an estimate of the upper bound, $\hat{\beta}^\star_{\text{ub}}$, is constructed by solving the analogous maximization problem. Both (11) and (12) can be reformulated as linear programs, and thus are easy to solve reliably and relatively quickly.[6] The tuning parameter τ is a non-negative constant used in the asymptotic theory (Mogstad et al., 2018). By construction, both (11) and (12) are always feasible as long as the parameter space Θ is non-empty.

Methodology

Based on the framework discussed in the previous section, Mogstad and Torgovitsky (2018) propose the following methodology for IV models:

1. Start with a specific empirical question and choose a target parameter $\beta^\star$ that answers that question. This choice determines the form of the weights $\omega^\star$ in (5).

[5] See Hansen (1982) for the classical reference, or Stock and Watson (2011, Chapter 18) for an introductory treatment.

[6] This is a result of switching from a weighted Euclidean norm in (9) to an ℓ_1 norm in (11). If instead we used the Euclidean norm in (11), then (12) would be quadratically-constrained quadratic program, which is a special case of a second-order cone program (see e.g. Boyd and Vandenberghe, 2004). While still a convex program, these are slower to solve, so at current we do not allow for this option in **ivmte**. A natural and expected consequence of this is that when there is a unique solution to (9), the estimated lower and upper bounds in (12) can differ from the point estimate (10).

2. Impose some assumptions on the MTR functions by choosing the set $\mathcal{M}$ and its finite-dimensional counterpart, Θ. This can involve parametric assumptions, as shown in (8). It can also involve shape restrictions, such as boundedness, monotonicity and separability. The assumptions allowed for in the **ivmte** package are described in the next section.
3. Choose some features of the data to fit by choosing a collection of functions $s \in \mathcal{S}$ and matching $\mathbb{E}[Y_i s(D_i, X_i, Z_i)]$ for each s. There are many choices for s. The **ivmte** package allows for any finite collection $\mathcal{S}$ such that each $s \in \mathcal{S}$ can each be expressed as a component of the vector of coefficients from a TSLS regression. This is a general class that includes coefficient estimates from simple IV and ordinary least squares (OLS) as special cases. Mogstad and Torgovitsky (2018) recommend choosing $\mathcal{S}$ to include "common practice" estimators, such as OLS and linear IV, which will typically be reported anyway as part of the empirical analysis.
4. If there are more moments than parameters, i.e. $|\mathcal{S}| \geq (K_0 + K_1)$, then solve (9) and construct a point estimate $\hat{\beta}^\star$ via (10). Otherwise, construct bounds on the target parameter $[\hat{\beta}^\star_{\text{lb}}, \hat{\beta}^\star_{\text{ub}}]$ by solving (11) and then (12). Note that constructing a point estimate can be seen as estimating "bounds" $\hat{\beta}^\star_{\text{lb}} = \hat{\beta}^\star_{\text{ub}} = \hat{\beta}^\star$.
5. Adjust steps 1–3 and repeat step 4 until satisfied with the trade-off between the strength of assumptions and the strength of conclusions. If the researcher finds that the estimated bounds $[\hat{\beta}^\star_{\text{lb}}, \hat{\beta}^\star_{\text{ub}}]$ are too wide to be of interest, then they have three options: (i) Return to step 1 and change the target parameter to reflect a less ambitious question; (ii) return to step 2 and make stronger assumptions; and (iii) return to step 3 and use more or different information from the data. Alternatively, if the estimated bounds are narrow or a point, then the researcher might wish to consider the converse of these three options: Asking a more ambitious question, making weaker assumptions, or using larger cuts from the data.

The ivmte package

The **ivmte** package is available in CRAN, and can be installed and loaded as usual:

```
install.packages("ivmte")
library("ivmte")
```

The most up-to-date version can be installed directly from the GitHub repository by:

```
devtools::install_github("jkcshea/ivmte")
```

Note that the package also requires one of the following packages for solving linear programs: **gurobi**, **cplexAPI** (Roettger et al., 2019), or **lpSolveAPI** (Konis, 2019). The first package requires a Gurobi (Gurobi Optimization, Inc., 2015) license, while the second requires a CPLEX (IBM, 2010) license. These are available at no cost for academic researchers. Alternatively, **lpSolveAPI** is freely available through CRAN and does not require a license. All of the examples shown in this paper were computed using **gurobi**.

The main command in **ivmte** is called `ivmte`. It requires the following arguments

```
ivmte(data, target, m0, m1, ivlike, propensity)
```

where `data` is a dataframe; `target` specifies the target parameter, $\beta^\star$; `m0` and `m1` are formulas indicating the specifications for the MTR functions; `ivlike` is a formula or list of formulas that determine which IV–like estimands (moments) are fit; and `propensity` is a formula that specifies how the propensity score is estimated. In this section, we discuss how to construct these arguments to implement the methodology of the previous section. Along the way, we cover a large number of additional options that provide extra functionality.

Specifying the target parameter

The `target` option can be set to one of `ate`, `att`, `atu`, `late` or `genlate`, which correspond respectively to the average treatment effect (ATE), the average treatment on the treated (ATT), the average treatment on the untreated (ATU), the local average treatment effect (LATE; Imbens and Angrist, 1994) and the generalized LATE (Heckman and Vytlacil, 2005; Mogstad et al., 2018). The choice of this argument determines the target parameter, $\beta^\star$. Here we focus on `late` and `genlate`, both of which require specifying arguments in addition to `target`.

The general form of a $z_{\text{from}} \to z_{\text{to}}$ LATE parameter is given by

$$\mathbb{E}\left[\int_{p(X_i, z_{\text{from}})}^{p(X_i, z_{\text{to}})} [m_1(u, X_i) - m_0(u, X_i)] \frac{1}{p(X_i, z_{\text{to}}) - p(X_i, z_{\text{from}})} du\right], \tag{13}$$

where we have combined the two weights in (5) for brevity. This is the form of $\beta^\star$ used if `target == late`. The user must pass `late.from` and `late.to`, which should be named lists. The `late.from` argument indicates the identity and value of z_{from} in (13), which one would typically choose to be a component of the instrument. Similarly, `late.to` indicates the identity and value of z_{to}, which one would typically choose to be the same component evaluated at a different point. The user can also compute a LATE that is conditional on an observed subgroup, in which case $\beta^\star$ has the form

$$\mathbb{E}\left[\int_{p(X_i,z_{\text{from}})}^{p(X_i,z_{\text{to}})} (m_1(u,X_i) - m_0(u,X_i)) \frac{1}{(p(X_i,z_{\text{to}}) - p(X_i,z_{\text{from}}))}\, du \middle| X_{i,c} = x\right]. \tag{14}$$

This is done by setting `target == late` and also passing a named list `late.X` to indicate the variable $X_{i,c}$ and value x in (14).

The generalized LATE is defined as

$$\mathbb{E}\left[\int_{u_{\text{lb}}}^{u_{\text{ub}}} (m_1(u,X_i) - m_0(u,X_i)) \frac{1}{(u_{\text{ub}} - u_{\text{lb}})}\, du\right], \tag{15}$$

for values $u_{\text{lb}}, u_{\text{ub}} \in [0,1]$, where again we have combined the two integrals in (5). The user can set the target parameter to the form (15) by passing `target = genlate` and including scalar arguments `genlate.lb` and `genlate.ub` to specify u_{lb} and u_{ub}. The optional named list `late.X` can also be passed here to turn (15) into a conditional-on-covariates version like (14).

The user can also define their own target parameters by directly specifying the weight functions $\omega_d^\star$ in (5), in which case `target` is ignored. To facilitate computation, these weight functions are required to be constant splines in u, i.e.

$$\omega_d^\star(u,X_i,Z_i) = \sum_{j=1}^{J_d} \mathbb{1}[\kappa_{d,(j-1)}(X_i,Z_i) < u \leq \kappa_{d,j}(X_i,Z_i)]\bar{\omega}_{d,j}^\star(X_i,Z_i),$$

where $k_{d,0}(X_i,Z_i) \equiv 0$, and $k_{d,J_d}(X_i,Z_i) \equiv 1$ always. The user sets these weights by passing the $J_0 - 1$ knot functions $(\kappa_{0,1},\ldots,\kappa_{0,(J_0-1)})$ as a list via `target.knots0` and the J_0 level functions $(\bar{\omega}_{0,1}^\star,\ldots,\bar{\omega}_{d,J_0}^\star)$ as a list via `target.weight0`. The analogous options `target.knots1` and `target.weight1` for the treated ($d = 1$) weights that also need to be passed. In all of these specifications, a constant (scalar numeric) can be passed instead of a function to indicate a function that does not vary with (X_i,Z_i).

Specifying the MTR functions

The required `m0` and `m1` arguments accept specifications for the MTR functions using the standard R formula syntax familiar from functions like `lm` or `glm`. However, these formulas involve an unobservable variable whose default name is `u`.[7] Typical specifications will involve combinations of `u` and other covariates. For example,

```
m0 = ~ var1 + u + I(var1*u) + I(u^2)
```

specifies an equation that is quadratic in the unobservable `u` and linear in `var1`, with a first order interaction between `u` and `var1`. Note that the left-hand side of these formulas is empty. Also note the use of `I()` to inhibit the interpretation of `*` and `^` as formula operators.

Currently, `ivmte` requires specifications of `m0` and `m1` to either be polynomials or B-splines in `u`.[8] B-splines are incorporated using the function `uSplines`, which is an interpreter that utilizes the **splines2** package (Wang and Yan, 2018). An example of the syntax is

```
m1 = ~ var1 + uSplines(degree = 0, knots = c(0.2, 0.5, 0.8))
```

which would specify m_1 to be linear in `var1` and piecewise constant in `u` with jumps at the specified knot points.[9] Splines can be interacted with other variables and intermingled with other polynomials, for example

```
m0 = ~ u + I(u^2)
    + var1 : uSplines(degree = 2, knots = c(0.3, 0.4, 0.5, 0.7))
```

[7] The name can be changed with the `uname` option.

[8] This is to ensure that the integrals over u in formulas like (5) and (6) can be computed analytically. For general specifications of the MTR functions these integrals would need to be computed numerically.

[9] As Wang and Yan (2018) describe in their vignette, the only difference between the `bSpline` function in **splines2** and the `bs` function in the core package **splines** is that `bSpline` allows for degree 0 splines, i.e. piecewise constant functions. This turns out to be particularly useful for our purposes because piecewise constant functions have a special place in the theory developed by Mogstad et al. (2018); see their Proposition 4.

would specify a quadratic function of `u` and a linear function of `var1` whose slope varies with u according to a quadratic B-spline with knot points at `.3`, `.4`, `.5`, and `.7`.

In addition to functional form restrictions, `ivmte` also allows the user to require the MTR and/or MTE functions to be bounded and/or monotone in u. The bounds are imposed through the arguments `m0.lb`, `m0.ub`, `m1.lb`, `m1.ub`, `mte.lb`, and `mte.ub`. Note that the default action of `ivmte` is to set the upper and lower bounds on `m0` and `m1` to the largest and smallest values of the response variable observed in the data, which also implies values for `mte.lb` and `mte.ub`. Monotonicity in u, in either an increasing or decreasing sense, is set through the boolean arguments `m0.dec`, `m0.inc`, `m1.dec`, `m1.inc`, `mte.dec`, and `mte.inc`. These arguments are set to `FALSE` by default. The shape constraints are only available when `point` is `FALSE`, so that bounds are being computed.

These shape constraints (boundedness and monotonicity) are enforced through an "auditing" procedure. The procedure is designed to circumvent the difficulty of determining whether a polynomial function is bounded or monotone at every point in its domain. It starts by imposing the desired shape constraints on the MTR functions at all points on a well-spaced, relatively coarse *constraint grid*. After solving the linear programs that produce the bound estimates $\hat{\beta}^{\star}_{\text{lb}}$ and $\hat{\beta}^{\star}_{\text{ub}}$, the solution MTR functions are checked ("audited") for these shape restrictions on a much finer *audit grid*. If the solutions fail to satisfy the shape restrictions anywhere on the audit grid, the linear programs are re-solved with an expanded constraint grid that contains some of the points in the audit grid where the restrictions were violated. The procedure repeats until the solutions pass the audit, or until a maximum number of iterations are reached.

The user can adjust the size of the initial constraint grid through the arguments `initgrid.nu` and `initgrid.nx`. These control the initial number of points at which to impose the constraints for `u`, via `initgrid.nu`, and all others variables included in the specification of `m0` and `m1`, via `initgrid.nx`. For the latter, the points are drawn randomly from the distribution in the dataframe. The default values of `initgrid.nu` and `initgrid.nx` are both 20, so that the total initial constraint grid size is 400.

The user can also adjust the size of audit grid through the arguments `audit.nu` and `audit.nx`. The default for `audit.nu` is 25, while the default for `audit.nx` is set at 2500. Thus, the solution MTR functions must satisfy the shape constraints on an audit grid with 62,500 points. In the event that an audit fails, the number of violating points that are added from the constraint grid to audit grid (for *each* shape constraint) is given by `audit.add`, which has a default of 100.

The audit is terminated after the solution MTR functions satisfy the constraints on the entire audit grid, or after `audit.max` rounds of the audit procedure, which has a default value of 25. If `audit.max` is hit, the user should investigate the `audit.grid$violations` field of the list that `ivmte` returns. This reports the points of the audit grid at which the shape restrictions are violated. Small regions of violation can likely be ignored without seriously affecting the estimated bounds $\hat{\beta}^{\star}_{\text{lb}}$ and $\hat{\beta}^{\star}_{\text{ub}}$. If the violations occur on a large region, the user can re-run the procedure with a larger audit grid, by adjusting `audit.nu` and/or `audit.nx`. Alternatively, they can let the audit procedure run for longer by increasing `audit.max`.

Specifying IV–like estimands

The IV–like estimands refer to the collection of functions $s \in \mathcal{S}$ used to fit the model in either (9) or (11)–(12). While any function $s(d, x, z)$ can be used, there are good arguments for choosing s such that the moment it generates, i.e. $\mathbb{E}[Y_i s(D_i, X_i, Z_i)]$ coincides with a well-understood estimator, such as OLS or IV (for further discussion, see Section 5.6 of Mogstad and Torgovitsky, 2018). Thus, for `ivmte` we have developed a system through which s can be specified as any component of a coefficient vector resulting from a TSLS regression with Y_i as the response variable.

The main driver of this system is the required argument `ivlike`. This can be any vector of formulas such that the same variable (Y_i) appears on the left-hand side of each equation. For example,

```
ivlike = c(y ~ d,
           y ~ d | z,
           y ~ d + x | z + I(z^2) + x)
```

has three formulas, with the second two specified using the `|` syntax familiar from the `ivreg` command in the **AER** package (Kleiber and Zeileis, 2018). The first formula is an OLS regression of `y` on `d` and a constant. The second formula uses `z` as an instrument for `d`, as in just-identified IV regressions. The third formula uses `z` and `z^2` as instruments for `d`, with `x` serving as a covariate that instruments for itself.

The default behavior of `ivmte` is to include all of the estimated coefficients from each specification as functions $s \in \mathcal{S}$. In the example above, this would mean the coefficients on the constant and `d` in the first and second specifications, and the coefficients on the constant, `d` and `x` in the third, for a total of 7 moments to match. The user can change this behavior with the `components` argument. This

argument expects a list of the same length as `ivlike`, with the jth component of the list being a vector that indicates which coefficients should be included from the jth IV–like specification in `ivlike`. In the example above, we could have used

```
components = l(intercept, d, c(d,x))
```

to indicate that we want only the coefficient on the constant from the first specification, only the coefficient on `d` in the second, and both the coefficients on `d` and `x` in the third, for a total of 4 moments. Note that `intercept` is used to refer to the implied constant term in the formula specifications, and so should be viewed as a reserved word when it comes to naming data columns.[10]

Conditioning subsets for the IV–like specification can be set through the optional `subset` argument. This option expects a list of the same length as `ivlike`, with each component of the list representing a logical statement. For example,

```
subset = l(z == 1, , x %in% c(2,3))
```

would estimate the first IV–like specification only on the subset with `z == 1`, the second for all observations, and the third only for the subset for which either `x == 2` or `x == 3`. This provides a mechanism for using conditional moments.

Propensity score estimation

Fitting equalities like (6) requires first estimating the propensity score, $p(x,z) \equiv \mathbb{P}[D_i = 1|X_i = x, Z_i = z]$. This is communicated through the `propensity` argument. Typically, the user will pass a formula for `propensity` in which the treatment variable appears on the left-hand side, for example

```
propensity = d ~ x + z
```

By default, this estimates a logit model using `glm` with the specified right-hand side variables, but the user can change this to probit or linear by passing `link = "probit"` or `link = "linear"`.

Alternatively, the user can estimate the propensity score before running `ivmte`, save estimates of $p(X_i, Z_i)$ in their dataframe as a new column, say `p`, and then pass `propensity = p`. When this is done, the user must also indicate the name of the treatment variable through the argument `treat`. When a formula is passed for `propensity`, the treatment variable is inferred to be the response variable from this formula, and the `treat` argument is ignored unless it doesn't match the inferred variable, in which case an error is thrown.

Solving

The default behavior of the `ivmte` command is to estimate bounds via (11) and (12). These problems can be solved using linear programming. The solver is set through the option `lpsolver`, which currently accepts the following values: `gurobi`, `cplexAPI`, and `lpSolveAPI`.[11] If no value is passed for `lpsolver`, then `ivmte` searches for the three packages in the order given above and uses the first one that is found. The value of the tuning parameter, τ, in (12) is set to 0 by default, and can be changed with the `criterion.tol` argument.

To construct a point estimate via (9) and (10), the user should pass `point = TRUE`. In this case, a linear GMM estimate of $\hat{\beta}^\star$ is constructed. If there are more moments than parameters, the default behavior of the `ivmte` command is to use the optimal two-step weighting. This can be changed to the identity weighting by passing `point.eyeweight = TRUE`.

Confidence intervals

The `ivmte` command can construct confidence intervals by resampling (bootstrapping or subsampling). The number of replications is determined by the argument `bootstraps`, which is set to 0 by default so that confidence intervals are not computed. The size of the resampled dataset is determined by `bootstraps.m`, which is set to the sample size of `data` by default. The default behavior is to draw the resampled data with replacement from `data`, but this can be toggled with the boolean argument `bootstraps.replace`.

[10] The 'l' function is a generalization of the 'list' function, and allows the user to list variables and expressions without having to enclose them by quotation marks.

[11] The **gurobi** package is included with Gurobi (Gurobi Optimization, Inc., 2015), while **cplexAPI** and **lpSolveAPI** are available from CRAN. The version requirements of **ivmte** as of this writing are: **gurobi** 7.5-1 or later, **cplexAPI** 1.3.3 or later, and **lpSolveAPI** 5.5.2 or later.

Confidence intervals are reported for all levels in `levels`, which has the default of `c(.99,.95,.90)`. If the user passes `point = TRUE`, the reported intervals are formed from the bootstrap distribution of (10) after re-centering the moment conditions (Hall and Horowitz, 1996; Brown and Newey, 2002). Conducting statistical inference on the bounds computed by (11)–(12) is more delicate due to their non-standard asymptotic distributions.[12] At current, there does not exist a solution for problems like (11)–(12) that is both theoretically satisfactory and computationally tractable. Instead, `ivmte` implements the forward and reverse bootstrap procedures discussed by Andrews and Han (2009).[13] While these are known to *not* be valid in general, they may still provide a reasonable indication of statistical uncertainty for the user. In addition to confidence intervals for each level in `levels`, `imvte` also returns a p-value, computed as the smallest α such that a $1-\alpha$ confidence interval would not contain 0.

Specification tests

When the criterion function (9) or (11) is non-zero, and `bootstraps` is a positive number, the `imvte` command will also conduct a test of the null hypothesis that the model is correctly specified. In the point-identified case (`point == TRUE`), the test used is the well-known Hansen (1982) overidentification test for GMM using the adjustment for bootstrapping discussed by Hall and Horowitz (1996). In the partially-identified case, the test used is the "re-sampling" test of Bugni et al. (2015). In either case, `ivmte` returns a p-value for the null hypothesis of correct specification. The user can turn off the specification test by passing `specification.test = FALSE`.

Output

The return of `ivmte` is a named list with a large number of fields.[14] The most important fields are `pointestimate` and `bounds`, which return (10) or (12), depending on whether `point` is `TRUE` or `FALSE`. If confidence intervals are being computed, these are returned in the fields `pointestimate.ci` or `bounds.ci`, with the p-value returned in the field `pvalue`. Other fields that may be useful for diagnostics or debugging are `sset`, which contains the results of running the IV–like specifications, `propensity`, which contains the results of the propensity score estimation, `audit.criterion`, which gives the value $\hat{Q}$ in (11), and `audit.grid$violations`, which reports points at which the audit procedure failed to secure compliance with the desired shape restrictions.

As it works through the various stages of estimation, `ivmte` reports messages that indicate its progress. While useful at first, and for debugging, these can become a bit excessive. They can be turned off by passing `noisy = FALSE`.

Empirical illustration

We illustrate the motivation and usage of **ivmte** with data from Angrist and Evans (1998).[15] The data comes from the 1980 Census Public Use Micro Samples (PUMS); a detailed description can be found in Angrist and Evans (1998). Our illustration uses three main variables: `worked` is an indicator for whether a woman worked for pay in the year prior to the survey, `morekids` is an indicator for whether a woman has exactly two children (`morekids = 0`) or three or more children (`morekids = 1`), and `samesex` is an indicator that is 1 if the first two children had the same sex. Later, we will also use the woman's year of birth, `yob`, to demonstrate specifications with covariates. Our interest is in the effect of fertility (`morekids`) on labor supply (`worked`).

A simple linear regression of `worked` on `morekids` returns the following:

```
lm(data = AE, worked ~ morekids)

Call:
lm(formula = worked ~ morekids, data = AE)
```

[12] See Canay and Shaikh (2017) for a recent survey on inference in partially identified models.

[13] The default is to compute and report the results from both backward and forward procedures. This behavior can be changed by passing `ci.type = "backward"` or `ci.type = "forward"`.

[14] In case memory usage is an important issue to the user, we have included an option `smallreturnlist` that can be set to `TRUE` to limit the number of objects that are returned.

[15] The original data can be downloaded from https://economics.mit.edu/files/1199 or from http://sites.bu.edu/ivanf/files/2014/03/m_d_806.dta_.zip. The data we use is restricted to women who were at least 20 years old at their first birth. The cleaned subsample data with only the variables relevant to the current analysis is included as data with **ivmte**.

```
Coefficients:
(Intercept)     morekids
     0.5822      -0.1423
```

The coefficient on `morekids` probably overstates the causal impact of fertility on labor supply, since women who choose to have more children likely do so in part because their labor market prospects are weaker. Indeed, an IV regression using `samesex` as an instrument for `morekids` returns a coefficient on `morekids` that is substantially smaller in magnitude:[16]

```
ivreg(data = AE, worked ~ morekids | samesex)

Call:
ivreg(formula = worked ~ morekids | samesex, data = AE)

Coefficients:
(Intercept)     morekids
    0.56315     -0.08484
```

Moreover, if there is heterogeneity in the effect of fertility on working, then this latter estimate only reflects the same-sex compliers, that is, those women who would have a third child if and only if their first two had the same sex. The "first stage" regression of `morekids` on `samesex` shows that this group is rather small, comprising less than 6% of the population.

```
lm(data = AE, morekids ~ samesex)

Call:
lm(formula = morekids ~ samesex, data = AE)

Coefficients:
(Intercept)      samesex
    0.30214      0.05887
```

If our research question requires knowing a quantity involving the entire population, such as the ATE or the ATT, then this linear IV estimate is not particularly helpful.

The **ivmte** package can be used to extrapolate from the small complier group to larger groups by providing a coherent framework under which additional assumptions can be imposed. Suppose for example that we assume that the MTR functions are both quadratic in u, so that the pair is characterized by six parameters. Since both `morekids` and `samesex` are binary, we only have four moments at our disposal to identify these six parameters, so the model is not point identified. However, we can use `ivmte` to estimate bounds on the ATE:

```
results <- ivmte(data = AE,
                 ivlike = c(worked ~ morekids + samesex + morekids*samesex),
                 target = "ate",
                 m0 = ~ u + I(u^2),
                 m1 = ~ u + I(u^2),
                 propensity = morekids ~ samesex)
#>
#> LP solver: Gurobi ('gurobi')
#>
#> Obtaining propensity scores...
#> Generating target moments...
#>     Integrating terms for control group...
#>     Integrating terms for treated group...
#> Generating IV-like moments...
#>     Moment 1...
#>     Moment 2...
#>     Moment 3...
#>     Moment 4...
#> Performing audit procedure...
#>     Generating audit grid...
#>     Generating initial constraint grid...
#>
#>     Audit count: 1
#>     Minimum criterion: 0
```

[16] The `ivreg` command requires the **AER** package.

```
#>     Obtaining bounds...
#>     Violations: 0
#>     Audit finished.
#>
#> Bounds on the target parameter: [-0.2862919, 0.1050867]
```

As a comparison, Manski's (1990) nonparametric IV bounds on the ATE are $[-.548, .393]$. The bounds produced by **ivmte** are much tighter because they impose a parametric assumption on the model primitives which smooths out the extreme cases characterized by Manski's bounds. Nevertheless, the bounds are still quite wide. However, Mogstad et al. (2018) show that they are sharp (best possible) in the sense of fully exhausting the information contained in the model and the data. Thus, if the researcher is unsatisfied with the width of the bounds, they have two paths to satisfaction: (i) Make stronger assumptions, or (ii) Ask a less ambitious question by changing the target parameter.

A natural way to strengthen the assumptions is to eliminate the quadratic terms in the MTR specifications, so that there are only four parameters:

```
results <- ivmte(data = AE,
                 ivlike = worked ~ morekids + samesex + morekids*samesex,
                 target = "ate",
                 m0 = ~ u,
                 m1 = ~ u,
                 propensity = morekids ~ samesex,
                 noisy = FALSE)
#>
#> Bounds on the target parameter: [-0.07791036, -0.07791036]
#> Audit terminated successfully after 1 round.
```

The bounds have collapsed to a point. This makes sense since we have not changed `ivlike`, so we still have four moments, but relative to the quadratic case we have reduced the number of parameters from six to four (Brinch et al., 2012, 2017). If we had done this moment-counting exercise ahead of time, we could have instead called

```
results <- ivmte(data = AE,
                 ivlike = worked ~ morekids + samesex + morekids*samesex,
                 target = "ate",
                 m0 = ~ u,
                 m1 = ~ u,
                 propensity = morekids ~ samesex,
                 point = TRUE,
                 noisy = FALSE)
#>
#> Point estimate of the target parameter: -0.07791036
```

which implements (9)–(10) rather than (11)–(12). Since the number of moments is equal to the number of parameters, the main difference is in computation time and how confidence intervals are constructed.[17]

Linearity is a strong assumption, and one might be uncomfortable with the fact that it allows for complete extrapolation from the 6% of the population represented in the LATE to the entire population represented in the ATE. As an alternative, consider combining the quadratic case with shape restrictions. For example, we could assume that the MTRs must generate an MTE curve that is negative and increasing:

```
results <- ivmte(data = AE,
                 ivlike = worked ~ morekids + samesex + morekids*samesex,
                 target = "ate",
                 m0 = ~ u + I(u^2),
                 m1 = ~ u + I(u^2),
                 mte.inc = TRUE,
                 mte.ub = 0,
                 propensity = morekids ~ samesex,
                 noisy = FALSE)
#>
#> Bounds on the target parameter: [-0.08484221, -0.06323574]
#> Audit terminated successfully after 1 round.
```

[17] In an "overidentified" case with more moments than parameters, the estimates can be different due to the differences between the quadratic and absolute loss functions employed in (9) and (11).

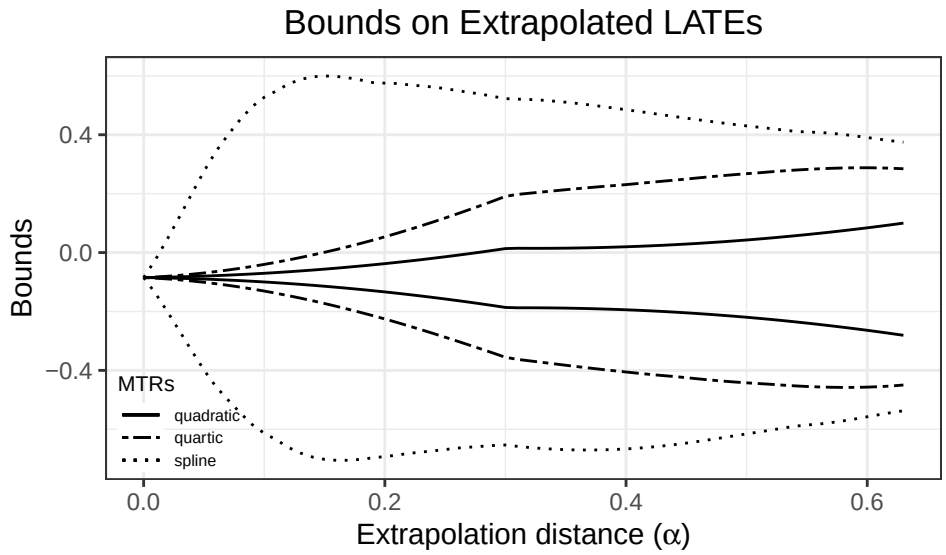

Figure 1: Bounds on extrapolated LATEs as a function of the extrapolation distance (α) for three different specifications of the MTR functions.

The assumption behind this shape restriction is that the effect of having another child on working is negative, and is more negative for women who are more likely to have more children. This assumption narrows the bounds considerably, from $[-.286, .105]$ to $[-.085, -.063]$.

We now have wide bounds from the quadratic assumption, very narrow bounds (in fact, a point) from the linearity assumption, and relatively narrow bounds from the quadratic specification with additional shape restrictions. Perhaps the researcher finds all of these assumptions to be too strong but is also not happy with the degree of agnosticism reflected in the quadratic bounds. This researcher has only one option if they wish to keep their IV strategy: Change the target parameter. Instead of the ATE, which requires extreme extrapolation, they could look at something more local, but still less local than the LATE. For example, they could consider *extrapolated LATEs*, i.e. generalized LATEs (15) with $u_{\text{lb}} = \max\{p(0) - \alpha, 0\}$ and $u_{\text{ub}} = \min\{p(1) + \alpha, 1\}$ for different non-negative values of α (Mogstad et al., 2018, Section 4.2). For $\alpha = 0$, the extrapolated LATE is equivalent to the usual LATE, while as $\alpha \to \max\{p(0), 1 - p(1)\}$, it returns to the ATE.

Figure 1 reports bounds on extrapolated LATEs as a function of α for three different specifications of the MTR functions. The tightest specification is the unconstrained quadratic used above. The quartic specification takes the quadratic and adds third and fourth order terms to both MTR functions. The spline specification is a flexible quadratic spline with nine knots. The plot was created with the following code:

```
# Get unique values of the propensity score from the previous run
p <- predict(results$propensity$model,
             newdata = data.frame(samesex = c(0,1)),
             type = "response")

# Function for computing genlate bounds at different values
loopivmte <- function(args, alphalist) {
    plotmat <- matrix(ncol = 3, nrow = length(alphalist))
    for (i in 1:length(alphalist)) {
        args[["genlate.lb"]] <- max(p[1] - alphalist[i], 0)
        args[["genlate.ub"]] <- min(p[2] + alphalist[i], 1)
        r <- do.call(ivmte, args)
        plotmat[i,] = c(alphalist[i], r$bound)
    }
    plotdf <- data.frame(plotmat)
    colnames(plotdf) <- c("a", "lb", "ub")
    return(plotdf)
}

# Set up function arguments as a list so it can be easily changed
args <- list(data = AE,
```

```
            ivlike = worked ~ morekids + samesex + morekids*samesex,
            target = "genlate",
            m0 = ~ u + I(u^2),
            m1 = ~ u + I(u^2),
            propensity = morekids ~ samesex,
            noisy = FALSE)

alphalist <- seq(from = 0, to = max(p[1], (1-p[2])), by = .01)

# Run the quadratic case
plotquadratic <- loopivmte(args, alphalist)

# Run the quartic case
args[["m0"]] <- ~ u + I(u^2) + I(u^3) + I(u^4)
args[["m1"]] <- args[["m0"]]
plotquartic <- loopivmte(args, alphalist)
plotdf <- merge(plotquadratic, plotquartic,
               by="a", suffixes = c(".quad", ".quartic"))

# Run the spline case
args[["m0"]] <- ~ uSplines(degree = 2,
                           knots = seq(from = .1, to = .9, by = .1))
args[["m1"]] <- args[["m0"]]
plotspline <- loopivmte(args, alphalist)
colnames(plotspline)[2:3] <- c("lb.spline", "ub.spline")
plotdf <- merge(plotdf, plotspline, by="a")

library("ggplot2")
pl <- ggplot() +
    xlab(expression(paste("Extrapolation distance (", alpha, ")"))) +
    ylab("Bounds") +
    ggtitle("Bounds on Extrapolated LATEs") +
    theme_bw() +
    theme(plot.title = element_text(hjust = 0.5)) +
    geom_line(data = plotdf,
              aes(x = a, y = lb.quad, linetype="quadratic")) +
    geom_line(data = plotdf,
              aes(x = a, y = ub.quad, linetype="quadratic")) +
    geom_line(data = plotdf,
              aes(x = a, y = lb.quartic, linetype="quartic")) +
    geom_line(data = plotdf,
              aes(x = a, y = ub.quartic, linetype="quartic")) +
    geom_line(data = plotdf,
              aes(x = a, y = lb.spline, linetype="spline")) +
    geom_line(data = plotdf,
              aes(x = a, y = ub.spline, linetype="spline")) +
    labs(linetype = "MTRs") +
    scale_linetype_manual(breaks = c("quadratic", "quartic", "spline"),
                          values = c("solid", "twodash", "dotted")) +
    theme(legend.position = c(0.01, 0.01),
          legend.justification = c(0,0),
          legend.background = element_rect(fill = alpha('white', 0)),
          legend.key = element_rect(fill = alpha('white', 1)),
          legend.text = element_text(size=6),
          legend.title = element_text(size=8),
          legend.margin = margin(.05, .05, .05, .05, "cm"),
          legend.key.height = unit(.3, "cm"))
ggsave("extrapolation.pdf", width = 5, height = 3, units = "in")
```

As expected, the bounds are always ordered in width with quadratic being narrowest and the quadratic spline being widest. More importantly, observe that for all specifications, the bounds widen from a point at $\alpha = 0$ (the LATE) towards the ATE bounds as $\alpha \to 1$. This is important because it shows how **ivmte** allows the researcher to achieve bounds of any width they desire, while still being constrained by the reality that stronger conclusions require stronger assumptions. Given this freedom, it is unlikely that the researcher's optimal location on the assumptions-conclusions frontier is the

corner solution of reporting only nonparametrically point-identified parameters such as the LATE, which reflect both the weakest assumptions and the weakest conclusions.

In applications, it is common to also include covariates, X_i. For IV strategies, these serve two roles. First, they can increase the credibility of the as-good-as-randomly-assigned assumption required for the instrument. Second, as in other contexts, covariates can reduce sampling uncertainty to the extent that they soak up residual variation in the outcome and/or treatment variables. Here, for the sake of simplicity, we focus on specifications with just the single covariate, `yob`, since this is sufficient to demonstrate our points.

As a first step, we go back to the quadratic specification with the ATE as the target parameter, but now fully interacted in `yob`:

```
results <- ivmte(data = AE,
                 ivlike = worked ~ (morekids + samesex + morekids*samesex)*yob,
                 target = "ate",
                 m0 = ~ u + yob + u*yob + I(u^2) + I(u^2)*yob,
                 m1 = ~ u + yob + u*yob + I(u^2) + I(u^2)*yob,
                 propensity = morekids ~ yob + samesex + samesex*yob,
                 noisy = FALSE)
#>
#> Bounds on the target parameter: [-0.2790478, 0.09365855]
#> Audit terminated successfully after 1 round.
```

The bounds are quite similar to the previous bounds that we obtained without covariates. We can impose more structure by eliminating interaction terms in the MTR functions:

```
results <- ivmte(data = AE,
                 ivlike = worked ~ (morekids + samesex + morekids*samesex)*yob,
                 target = "ate",
                 m0 = ~ u + yob + u*yob + I(u^2),
                 m1 = ~ u + yob + u*yob + I(u^2),
                 propensity = morekids ~ yob + samesex + samesex*yob,
                 noisy = FALSE)
#>
#> Bounds on the target parameter: [-0.1206799, 0.03139476]
#> Audit terminated successfully after 1 round.
```

This is a *separability* assumption, which is common in empirical economics (see e.g. Mogstad and Torgovitsky, 2018, Section 6.2). Intuitively, it restricts the quadratic impact of unobservable selection heterogeneity to be the same for women born in different years, while still allowing for both the level and linear relationship to vary with `yob`. As the bounds in this example show, separability assumptions can contain a great deal of information.

When using covariates especially, one might wish to use multiple or different IV-like specifications, so as to avoid using moments that are noisily estimated. This will typically widen the bounds, since the parameters are being fit to fewer moments. For example, here we drop two of the eight moments:

```
results <- ivmte(data = AE,
                 ivlike = worked ~ (morekids + samesex + morekids*samesex)*yob,
                 components = l(morekids, samesex, morekids:samesex,
                                samesex:yob, morekids:yob, morekids:samesex:yob),
                 target = "ate",
                 m0 = ~ u + yob + u*yob + I(u^2),
                 m1 = ~ u + yob + u*yob + I(u^2),
                 propensity = morekids ~ yob + samesex + samesex*yob,
                 noisy = FALSE)
#>
#> Bounds on the target parameter: [-0.1574808, 0.1155318]
#> Audit terminated successfully after 1 round.
```

Note that using a subset of the components is different than changing the `ivlike` formula itself because a short and long regression will not necessarily have the same coefficients on variables they share in common. So, for example, the following leads to different bounds even though the same terms are involved:

```
results <- ivmte(data = AE,
                 ivlike = c(worked ~ 0 + morekids + samesex + morekids*samesex,
                            worked ~ 0 + samesex:yob + morekids:yob
```

```
                                + morekids:samesex:yob),
                target = "ate",
                m0 = ~ u + yob + I(u^2),
                m1 = ~ u + yob + I(u^2),
                propensity = morekids ~ yob + samesex + samesex*yob,
                noisy = FALSE)
#>
#> Bounds on the target parameter: [-0.1222383, 0.1484153]
#> Audit terminated successfully after 1 round.
```

Finally, we demonstrate how `ivmte` constructs confidence intervals. If we estimate the model assuming point identification, as with a linear specification, `ivmte` returns

```
ivmte(data = AE,
      ivlike = worked ~ morekids + samesex + morekids*samesex,
      target = "ate",
      m0 = ~ u,
      m1 = ~ u,
      point = TRUE,
      bootstraps = 100,
      propensity = morekids ~ samesex,
      noisy = FALSE)
#>
#> Point estimate of the target parameter: -0.07791036
#>
#> Bootstrapped confidence intervals (nonparametric):
#>     90%: [-0.1477243, -0.009919181]
#>     95%: [-0.1648582, -0.0004101071]
#>     99%: [-0.1909144, 0.02835332]
#> p-value: 0.06
#> Number of bootstraps: 100
```

While for the general case of bound estimation, `ivmte` returns:

```
ivmte(data = AE,
      ivlike = worked ~ morekids + samesex + morekids*samesex,
      target = "ate",
      m0 = ~ u + I(u^2),
      m1 = ~ u + I(u^2),
      mte.inc = TRUE,
      mte.ub = 0,
      bootstraps = 100,
      propensity = morekids ~ samesex,
      noisy = FALSE)
#>
#> Bounds on the target parameter: [-0.08484221, -0.06323574]
#> Audit terminated successfully after 1 round.
#>
#> Bootstrapped confidence intervals (backward):
#>     90%: [-0.138023, -0.024873]
#>     95%: [-0.1446129, -0.01647855]
#>     99%: [-0.2094434, -0.003587793]
#> p-value: 0
#> Number of bootstraps: 100
```

Acknowledgments

This research was supported in part by National Science Foundation grant SES-1530538. We thank Christine Blandhol and John Bonney for testing the package during its development.

Bibliography

D. W. K. Andrews and S. Han. Invalidity of the bootstrap and the m out of n bootstrap for confidence interval endpoints defined by moment inequalities. *Econometrics Journal*, 12:S172–S199, 2009. ISSN

1368-423X. [p9]

J. D. Angrist and W. N. Evans. Children and Their Parents' Labor Supply: Evidence from Exogenous Variation in Family Size. *The American Economic Review*, 88(3):450–477, June 1998. ISSN 00028282. [p1, 9]

B. S. Barnow, G. G. Cain, A. S. Goldberger, et al. *Issues in the Analysis of Selectivity Bias*. University of Wisconsin, Inst. for Research on Poverty, 1980. [p2]

A. Björklund and R. Moffitt. The Estimation of Wage Gains and Welfare Gains in Self-Selection Models. *The Review of Economics and Statistics*, 69(1):42–49, Feb. 1987. ISSN 00346535. [p1]

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge university press, 2004. [p4]

C. N. Brinch, M. Mogstad, and M. Wiswall. Beyond LATE with a Discrete Instrument. *Working paper*, 2012. [p1, 11]

C. N. Brinch, M. Mogstad, and M. Wiswall. Beyond LATE with a Discrete Instrument. *Journal of Political Economy*, 125(4):985–1039, Aug. 2017. doi: 10.1086/692712. [p1, 11]

B. W. Brown and W. K. Newey. Generalized Method of Moments, Efficient Bootstrapping, and Improved Inference. *Journal of Business & Economic Statistics*, 20(4):507–517, Oct. 2002. ISSN 0735-0015. doi: 10.1198/073500102288618649. [p9]

F. A. Bugni, I. A. Canay, and X. Shi. Specification tests for partially identified models defined by moment inequalities. *Journal of Econometrics*, 185(1):259–282, Mar. 2015. ISSN 0304-4076. [p9]

I. A. Canay and A. M. Shaikh. Practical and Theoretical Advances in Inference for Partially Identified Models. In B. Honore, A. Pakes, M. Piazzesi, and L. Samuelson, editors, *Advances in Economics and Econometrics*, pages 271–306. Cambridge University Press, 2017. doi: 10.1017/9781108227223.009. [p9]

P. Carneiro, J. J. Heckman, and E. J. Vytlacil. Estimating Marginal Returns to Education. *American Economic Review*, 101(6):2754–81, 2011. doi: 10.1257/aer.101.6.2754. [p1]

X. Chen. Chapter 76 Large Sample Sieve Estimation of Semi-Nonparametric Models. In J. J. Heckman and E. E. Leamer, editors, *Handbook of Econometrics*, volume Volume 6, Part 2, pages 5549–5632. Elsevier, 2007. [p3]

Gurobi Optimization, Inc. Gurobi Optimizer Reference Manual. 2015. [p5, 8]

P. Hall and J. L. Horowitz. Bootstrap Critical Values for Tests Based on Generalized-Method-of-Moments Estimators. *Econometrica*, 64(4):891–916, July 1996. ISSN 00129682. [p9]

L. P. Hansen. Large Sample Properties of Generalized Method of Moments Estimators. *Econometrica*, 50(4):1029–1054, July 1982. ISSN 00129682. [p4, 9]

J. J. Heckman. The Common Structure of Statistical Models of Truncation, Sample Selection and Limited Dependent Variables and a Simple Estimator for Such Models. *Annals of Economic and Social Measurement*, 1976. [p1]

J. J. Heckman and R. Robb. Alternative methods for evaluating the impact of interventions. In J. J. Heckman and B. Singer, editors, *Longitudinal Analysis of Labor Market Data*. Cambridge University Press, 1985. [p1]

J. J. Heckman and E. Vytlacil. Structural Equations, Treatment Effects, and Econometric Policy Evaluation. *Econometrica*, 73(3):669–738, 2005. ISSN 1468-0262. [p1, 2, 3, 5]

J. J. Heckman and E. J. Vytlacil. Local Instrumental Variables and Latent Variable Models for Identifying and Bounding Treatment Effects. *Proceedings of the National Academy of Sciences of the United States of America*, 96(8):4730–4734, Apr. 1999. ISSN 00278424. [p1, 2]

J. J. Heckman and E. J. Vytlacil. Chapter 70 Econometric Evaluation of Social Programs, Part I: Causal Models, Structural Models and Econometric Policy Evaluation. In J. J. Heckman and E. E. Leamer, editors, *Handbook of Econometrics*, volume Volume 6, Part 2, pages 4779–4874. Elsevier, 2007a. [p1, 2]

J. J. Heckman and E. J. Vytlacil. Chapter 71 Econometric Evaluation of Social Programs, Part II: Using the Marginal Treatment Effect to Organize Alternative Econometric Estimators to Evaluate Social Programs, and to Forecast their Effects in New Environments. In J. J. Heckman and E. E. Leamer, editors, *Handbook of Econometrics*, volume Volume 6, Part 2, pages 4875–5143. Elsevier, 2007b. [p1, 2]

J. J. Heckman, H. Ichimura, J. Smith, and P. Todd. Sources of selection bias in evaluating social programs: An interpretation of conventional measures and evidence on the effectiveness of matching as a program evaluation. *Proceedings of the National Academy of Sciences*, 93(23):13416–13420, 1996. [p2]

J. J. Heckman, S. Urzua, and E. Vytlacil. Understanding Instrumental Variables in Models with Essential Heterogeneity. *Review of Economics and Statistics*, 88(3):389–432, 2006. doi: 10.1162/rest.88.3.389. [p2]

IBM. *IBM ILOG AMPL Version 12.2*. International Business Machines Corporation, 2010. [p5]

G. W. Imbens and J. D. Angrist. Identification and Estimation of Local Average Treatment Effects. *Econometrica*, 62(2):467–475, Mar. 1994. ISSN 00129682. [p1, 2, 5]

C. Kleiber and A. Zeileis. *AER: Applied Econometrics with R*, 2018. URL https://CRAN.R-project.org/package=AER. R package version 1.2-6. [p7]

K. Konis. *lpSolveAPI: R Interface to 'lp_solve'*, 2019. URL https://CRAN.R-project.org/package=lpSolveAPI. R package version 5.5.2.0-17.4. [p5]

C. F. Manski. Nonparametric Bounds on Treatment Effects. *The American Economic Review*, 80(2): 319–323, May 1990. ISSN 00028282. [p1, 11]

C. F. Manski. *Partial Identification of Probability Distributions*. Springer, 2003. [p4]

R. L. Matzkin. Chapter 73 Nonparametric identification. In J. J. Heckman and E. E. Leamer, editors, *Handbook of Econometrics*, volume Volume 6, Part 2, pages 5307–5368. Elsevier, 2007. [p2]

M. Mogstad and A. Torgovitsky. Identification and Extrapolation of Causal Effects with Instrumental Variables. *Annual Review of Economics*, 10(1), May 2018. doi: 10.1146/annurev-economics-101617-041813. [p2, 3, 4, 5, 7, 14]

M. Mogstad, A. Santos, and A. Torgovitsky. Using Instrumental Variables for Inference About Policy Relevant Treatment Parameters. *Econometrica*, 86(5):1589–1619, 2018. doi: 10.3982/ecta15463. [p1, 2, 3, 4, 5, 6, 11, 12]

M. Mogstad, A. Torgovitsky, and C. R. Walters. Identification of Causal Effects with Multiple Instruments: Problems and Some Solutions. *Working paper*, 2019. [p2]

M. Roettger, G. Gelius-Dietrich, and J. C. Fritzemeier. *cplexAPI: R Interface to C API of IBM ILOG CPLEX*, 2019. URL https://CRAN.R-project.org/package=cplexAPI. R package version 1.3.6. [p5]

P. R. Rosenbaum and D. B. Rubin. The Central Role of the Propensity Score in Observational Studies for Causal Effects. *Biometrika*, 70(1):41–55, Apr. 1983. ISSN 00063444. [p2]

J. Shea and A. Torgovitsky. *ivmte: Instrumental Variables: Extrapolation by Marginal Treatment Effects*, 2019. URL https://CRAN.R-project.org/package=ivmte. R package version 1.1.0. [p1]

J. H. Stock and F. Trebbi. Retrospectives: Who Invented Instrumental Variable Regression? *Journal of Economic Perspectives*, 17(3):177–194, Aug. 2003. doi: 10.1257/089533003769204416. [p1]

J. H. Stock and M. W. Watson. Introduction to Econometrics, 3/E. 2011. [p4]

E. Tamer. Partial Identification in Econometrics. *Annual Review of Economics*, 2(1):167–195, Sept. 2010. doi: 10.1146/annurev.economics.050708.143401. [p4]

E. Vytlacil. Independence, Monotonicity, and Latent Index Models: An Equivalence Result. *Econometrica*, 70(1):331–341, Jan. 2002. ISSN 00129682. [p2]

W. Wang and J. Yan. *splines2: Regression Spline Functions and Classes*, 2018. URL https://CRAN.R-project.org/package=splines2. R package version 0.2.8. [p6]

Joshua Shea
Department of Economics
University of Chicago
United States of America
jkcshea@uchicago.edu

Alexander Torgovitsky
Department of Economics
University of Chicago
United States of America
torgovitsky@uchicago.edu