R and Stata for Causal Mechanisms

Dustin Tingley

Harvard University

March 23, 2012

2012
Proposed General Estimation Algorithm

1. Model outcome and mediator
   - Outcome model: \( p(Y_i \mid T_i, M_i, X_i) \)
   - Mediator model: \( p(M_i \mid T_i, X_i) \)
   - These models can be of any form (linear or nonlinear, semi- or nonparametric, with or without interactions)

2. Predict mediator for both treatment values \((M_i(1), M_i(0))\)

3. Predict outcome by first setting \(T_i = 1\) and \(M_i = M_i(0)\), and then \(T_i = 1\) and \(M_i = M_i(1)\)

4. Compute the average difference between two outcomes to obtain a consistent estimate of ACME

\[
\bar{\delta}(t) \equiv \mathbb{E}(\delta_i(t)) = \mathbb{E}\{Y_i(t, M_i(1)) - Y_i(t, M_i(0))\}
\]

5. Monte-Carlo or bootstrapping to estimate uncertainty
1. Install R/load mediation library/load data. Go here to learn these basics.

2. Fit models for the mediator and outcome variable and store these models.

   ```
   > m <- lm(Mediator ~ Treat + X)
   > y <- lm(Y ~ Treat + Mediator + X)
   ```

3. **Mediation analysis**: Feed model objects into the `mediate()` function. Call a summary of results.

   ```
   > m.out <- mediate(m, y, treat = "Treat", mediator = "Mediator")
   > summary(m.out)
   ```
Data Types Available via mediation

<table>
<thead>
<tr>
<th>Mediator Model Types</th>
<th>Linear</th>
<th>GLM</th>
<th>Ordered</th>
<th>Censored</th>
<th>Quantile</th>
<th>GAM</th>
<th>Survival</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear (lm)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>GLM (glm/bayesglm)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Ordered (polr/bayespolr)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Censored (tobit via vglm)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Quantile (rq)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>GAM (gam)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Survival (survreg)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table: Types of Models That Can be Handled by `mediate`. Stars (*) indicate the model combinations that can only be estimated using the nonparametric bootstrap (i.e. with `boot = TRUE`).
Additional Features

- Treatment/mediator interactions
- Treatment/mediator/pre-treatment interactions and reporting of quantities by pre-treatment values
- Factoral, continuous treatment variables
- Cluster standard errors/adjustable CI reporting/p-values
- Multiple mediators

Please read our vignette file.
Figure: Structure of the R mediation package as of version 4.0.
Plotting
Built in use of native plot function

> plot(m.out)
Sensitivity analysis

Feed the output into the `medsens` function, choose type of sensitivity expression, summarize and plot.

```r
> s.out <- medsens(m.out)
> summary(s.out)
> plot(s.out, "rho")
> plot(s.out, "R2")
```
Causal mediation analysis in Stata

Based on the same algorithm


ssc install mediation

More limited coverage of models (just bc. of time though!)
Syntax: medeff

medeff (equation 1) (equation 2) [if] [in] [[weight]] ,
[sims(integer) seed(integer) vce(vcetype) Level(#) interact(varname)] mediate(varname) treat(varname)

Where “equation 1” or “equation 2” are of the form (For equation 1, the mediator equation):

probit M T x

or

regress M T x
Syntax: medsens

medsens (equation 1) (equation 2) [if exp] [in range]
   [, sims(integer) seed(integer ) Level(#) graph]
   mediate(varname) treat(varname)
Overview

- Sequential ignorability assumption is strong, but buys lots of flexibility
- Proposed simulation approach can cover lots of different modeling situations

Of course, its a strong assumption so you might not want to make it
  - Sensitivity analysis
  - Alternative designs (read this paper) that require different assumptions