Appendix

Stata dofile replicating results reported in empirical example

```
use gss_redist.dta, clear // GSS data (selected variables)
* Listwise deletion
egen miss = rmiss(redincdiff pinc year age rincome educ prestg10 marital region)
keep if miss == 0
* Descriptives
d goveqinc redincdiff pinc year age rincome educ prestg10 marital region
tab1 goveqinc redincdiff pinc year age rincome educ prestg10 marital region
* Naive Approach
ologit redincdiff b1.pinc  i.year c.age
est store n1
ologit redincdiff b1.pinc  i.year c.age c.rincome
est store n2
ologit redincdiff b1.pinc  i.year c.age c.rincome c.educ
est store n3
ologit redincdiff b1.pinc  i.year c.age c.rincome c.educ c.prestg10
est store n4
ologit redincdiff b1.pinc  i.year c.age c.rincome c.educ c.prestg10 i.marital i.region
est store n5
est tab n1 n2 n3 n4 n5, se // Table of naive models
* Linear Predictor Approach
// Step 1: Run "Full" model (including all covariates) and save linear predictor
ologit redincdiff b1.pinc i.year c.age  c.rincome c.educ c.prestg10 i.marital i.region
predict xb, xb
// Step 2: Run reduced models on linear predictor to construct table of models
reg xb b1.pinc  i.year c.age
est store m1
reg xb b1.pinc  i.year c.age c.rincome
```

```
est store m2
reg xb b1.pinc  i.year c.age c.rincome c.educ
est store m3
reg xb b1.pinc  i.year c.age c.rincome c.educ c.prestg10
est store m4
reg xb b1.pinc  i.year c.age c.rincome c.educ c.prestg10 i.marital i.region
est store m5
est tab m1 m2 m3 m4 m5, not nostar // Table of “correct” models
* Bootstrap SE for Linear Predictor Approach
cap progr drop bsxb // erase any stored programs
progr define bsxb, rclass // program starts
// Step 1: Run “Full” model and save linear predictor
ologit redincdiff b1.pinc i.year c.age  c.rincome c.educ c.prestg10 i.marital i.region
cap drop xb
predict xb, xb
// Step 2: Run models and save parameters of interest relevant for inference
reg xb b1.pinc  i.year c.age
return scalar m1_at = _b[2.pinc]
return scalar m1_above = _b[3.pinc]
reg xb b1.pinc  i.year c.age c.rincome
return scalar m2_at = _b[2.pinc]
return scalar m2_above = _b[3.pinc]
reg xb b1.pinc  i.year c.age c.rincome c.educ
return scalar m3_at = _b[2.pinc]
return scalar m3_above = _b[3.pinc]
reg xb b1.pinc  i.year c.age c.rincome c.educ c.prestg10
return scalar m4_at = _b[2.pinc]
return scalar m4_above = _b[3.pinc]
reg xb b1.pinc  i.year c.age c.rincome c.educ c.prestg10 i.marital i.region
```

```
return scalar m5_at = _b[2.pinc]
return scalar m5_above = _b[3.pinc]
drop xb
end // program ends
// Run bootstrap program:
bs        m1_at=r(m1_at) m1_above=r(m1_above) ///
m2_at=r(m2_at) m2_above=r(m2_above) ///
m3_at=r(m3_at) m3_above=r(m3_above) ///
m4_at=r(m4_at) m4_above=r(m4_above) ///
m5_at=r(m5_at) m5_above=r(m5_above) ///
, reps(500) seed(1001): bsxb, all
* Bootstrap SE for Linear Predictor Approach: Indirect Effects and Percent Mediated
cap progr drop bsxbdif // erase any stored programs
progr define bsxbdif, rclass // program starts
// Step 1: Run "Full" model and save linear predictor
ologit redincdiff b1.pinc i.year c.age  c.rincome c.educ c.prestg10 i.marital i.region
cap drop xb
predict xb, xb
// Step 2: Run models and save parameters of interest
reg xb b1.pinc  i.year c.age
scalar m1_at = _b[2.pinc]
scalar m1_above = _b[3.pinc]
reg xb b1.pinc  i.year c.age c.rincome
scalar m2_at = _b[2.pinc]
scalar m2_above = _b[3.pinc]
reg xb b1.pinc  i.year c.age c.rincome c.educ
scalar m3_at = _b[2.pinc]
scalar m3_above = _b[3.pinc]
reg xb b1.pinc  i.year c.age c.rincome c.educ c.prestg10
```

```
scalar m4_at = _b[2.pinc]
scalar m4_above = _b[3.pinc]
reg xb b1.pinc  i.year c.age c.rincome c.educ c.prestg10 i.marital i.region
scalar m5_at = _b[2.pinc]
scalar m5_above = _b[3.pinc]
drop xb
// Step 3: Generate parameters of interest relevant for inference
return scalar m1m2_at = m1_at-m2_at
return scalar m1m2_above = m1_above-m2_above
return scalar m1m3_at = m1_at-m3_at
return scalar m1m3_above = m1_above-m3_above
return scalar m2m3_at = m2_at-m3_at
return scalar m2m3_above = m2_above-m3_above
return scalar p_m1m2_at = (m1_at-m2_at)/m1_at * 100
return scalar p_m1m2_above = (m1_above-m2_above)/m1_above * 100
return scalar p_m1m3_at = (m1_at-m3_at)/m1_at * 100
return scalar p_m1m3_above = (m1_above-m3_above) / m1_above * 100
return scalar p_m2m3_at = (m2_at-m3_at)/m2_at * 100
return scalar p_m2m3_above = (m2_above-m3_above)/m2_above * 100
end // program ends
// Run bootstrap program:
bs        m1m2_at=r(m1m2_at) ///
m1m3_at=r(m1m3_at) ///
m2m3_at=r(m2m3_at) ///
p_m1m2_at=r(p_m1m2_at) ///
p_m1m3_at=r(p_m1m3_at) ///
p_m2m3_at=r(p_m2m3_at) ///
m1m2_above=r(m1m2_above) ///
m1m3_above=r(m1m3_above) ///
```

```
m2m3_above=r(m2m3_above) ///
p_m1m2_above=r(p_m1m2_above) ///
p_m1m3_above=r(p_m1m3_above) ///
p_m2m3_above=r(p_m2m3_above) ///
, reps(500) seed(1001): bsxbdif, all
```