Stata codes for mediation and 4-way decomposition analysis under counterfactual causal framework for case-control study design

Thekke Purakkal, Akhil Soman[1], Kaufman, Jay S[2].

[1]Division of Oral Health and Society, Faculty of Dentistry, McGill University, Montreal Quebec, [2]Department of Epidemiology, Biostatistics and Occupation Health, McGill University, Montreal, Quebec

Date: August 1, 2016; Version 1

Stata macros (e.g., PARAMED) for estimating mediation effects in the presence of exposure mediator interaction under counterfactual causal framework already exist (1). For conducting 4-way decomposition analysis, although the mathematical equations and SAS codes have been provided by VanderWeele 2014 (2), Stata codes have not been written. The below Stata codes were exclusively written for conducting mediation analysis under exposure-mediator interaction (alternative method using mathematical equations), as well as 4-way decomposition analysis for this thesis work. Although the codes given here is specific for binary outcome, mediator and exposure variables, the code can be easily extended to the case where the exposure, mediator and outcome are continuous, categorical, binary or their combinations. The codes have been written using the mathematical formulas for total effect, mediation effects, and 4-way decomposition effects, as well as for calculating various proportions provided by VanderWeele 2015 and 2016 (3, 4). The user is encouraged to cross check these codes with the formulas in these references. The codes for bootstrapping procedure given at the end of the codes can be used to derive the confidence limits for these estimates.

Let Y be a binary outcome. In a case-control study, Y=1 may represent cases and Y=0 may represent controls or non-cases. Let A be a binary exposure, and M a binary mediator. Let C1, C2, be continues covariates, and C3, C4 be binary or categorical. If there are more or fewer covariates, one can add or remove scalars under "//Covariates", "//Assigning levels of covariates" and "//calculating bcc" in the below given code. For a case-control study with rare disease outcome, the line of code for mediator model may be fit only among controls. Alternatively, or, if the outcome is not rare, one can weight the mediator model using sampling weights as suggested by VanderWeele and Vansteelandt 2010 (5).

References

1. VanderWeele TJ. Mediation: Introduction and regresion -based approaches. Explanation in Causal inference: Methods for mediation and Interaction. USA: Oxford University Press; 2015. p. 40-1.
2. VanderWeele TJ. A unification of mediation and interaction: a 4-way decomposition. Epidemiology (Cambridge, Mass). 2014;25(5):749-61.

3. VanderWeele TJ. A unification of mediation and interaction. Explanation in Causal inference: Methods for mediation and interaction. USA: Oxford University Press; 2015. p. 371-96.
4. Erratum: A Unification of Mediation and Interaction: A 4-Way Decomposition. Epidemiology (Cambridge, Mass). 2016;27(5):e36.
5. Vanderweele TJ, Vansteelandt S. Odds ratios for mediation analysis for a dichotomous outcome. American journal of epidemiology. 2010;172(12):1339-48.

```
****** Run code from the line below till the end at a single stretch *******

**Start of code **

set varabbrev off, perm
cap prog drop calc2
prog calc2, rclass

logit Y A##M C1 C2 C3 C4 // Outcome model

scalar t1=_b[1.A]
scalar t2=_b[1.M]
scalar t3=_b[1.A#1.M]

logit M A C1 C2 C3 C4 if Y==0 // Mediator model fit among among controls

scalar b0=_b[_cons]
scalar b1=_b[A]

//Covariates

scalar bc1 = _b[C1]
scalar bc2 = _b[C2]
scalar bc3 = _b[1.C3]
scalar bc4 = _b[1.C4]

// Assigning level of covariates // for continuous covariates, just take the mean of
// their distribution in full sample

sum C1
scalar cc1= r(mean)
sum C2
scalar cc2= r(mean)
scalar cc3=1 // at level 1 of binary covariate C3. Any level can be assigned based on
// requirement
scalar cc4=3 // at level 3 of 3 category covariate C4

// Calculating bcc - Calculating sum of products of coefficients of covariates from
// mediator model and level of covariate

scalar bcc = bc1*cc1 + bc2*cc2 + bc3*cc3+ bc4*cc4

// Additional values assigned

scalar a1=1 // level 1 of exposure A
scalar a0=0 // level 0 of exposure A
scalar m0=0 // level 0 of binary mediator
scalar mstar=0 // level of mediator at which CDE is calculated
```

```
// 2-way decomposition or mediation – Calculating coefficients for natural direct
effect (NDE), natural indirect effect (NIE) and total effect (total)

scalar lnde    = ln((exp(t1*a1)*(1+exp(t2+t3*a1+b0+b1*a0+bcc))) ///
                  /(exp(t1*a0)*(1+exp(t2+t3*a0+b0+b1*a0+bcc))))

scalar lnie    = ln(((1+exp(b0+b1*a0+bcc))*(1+exp(t2+t3*a1+b0+b1*a1+bcc))) ///
                  /((1+exp(b0+b1*a1+bcc))* (1+exp(t2+t3*a1+b0+b1*a0+bcc))))

scalar ltotal  = ln((exp(t1*a1)*(1+exp(b0+b1*a0+bcc))* ///
                 (1+exp(b0+b1*a1+bcc+t2+t3*a1)))/(exp(t1*a0)* ///
                 (1+exp(b0+b1*a1+bcc))* (1+exp(b0+b1*a0+bcc+t2+t3*a0))))

// 4- way decomposition - Calculating coefficients for controlled direct effect (CDE),
// reference interaction(INTref), mediated interaction(INTmed), pure indirect effect
// (PIE)

scalar lcde    = ln(exp(t1 + t3*mstar)*(a1-a0))

scalar lIntref = ln((exp(t1*(a1-a0)-t2*mstar-t3*a0*mstar)* ///
                 (1+exp(b0+b1*a0+bcc+t2+t3*a1))) /(1+exp(b0+b1*a0+bcc)) ///
                 -(exp(-t2*mstar-t3*a0*mstar)*(1+exp(b0+b1*a0+bcc+t2+t3*a0))) ///
                 /(1+exp(b0+b1*a0+bcc)) - exp((t1+t3*mstar)*(a1-a0)) + 1)

scalar lIntmed = ln((exp(t1*(a1-a0)-t2*mstar-t3*a0*mstar)* ///
                 (1+exp(b0+b1*a1+bcc+t2+t3*a1)) /(1+exp(b0+b1*a1+bcc))) ///
                 - (exp(-t2*mstar-t3*a0*mstar)*(1+exp(b0+b1*a1+bcc+t2+t3*a0)) ///
                 /(1+exp(b0+b1*a1+bcc)))- exp(t1*(a1-a0)-t2*mstar-t3*a0*mstar) ///
                 *(1+exp(b0+b1*a0+bcc+t2+t3*a1))/(1+exp(b0+b1*a0+bcc)) ///
                 + exp(-t2*mstar-t3*a0*m0)*(1+exp(b0+b1*a0+bcc+t2+t3*a0)) ///
                  /(1+exp(b0+b1*a0+bcc)))

scalar lpie    = ln((1+exp(b0+b1*a0+bcc))*(1+exp(b0+b1*a1+bcc+t2+t3*a0)) ///
                 /((1 + exp(b0+b1*a1+bcc))*(1+exp(b0+b1*a0+bcc+t2+t3*a0))))

// Calculating coefficients for each 4-way component and total effect

scalar cde_comp    = (exp(t1*(a1-a0)+t2*mstar+t3*a1*mstar)*(1+exp(b0+b1*a0+bcc))/ ///
                     (1+exp(b0+b1*a0+bcc+t2+t3*a0)))-(exp(t2*mstar+t3*a0*mstar)* ///
                     (1+exp(b0+b1*a0+bcc))/(1+exp(b0+b1*a0+bcc+t2+t3*a0)))
scalar INTref_comp = exp(t1*(a1-a0))*(1+exp(b0+b1*a0+bcc+t2+t3*a1)) ///
                     /(1+exp(b0+b1*a0+bcc+t2+t3*a0)) - (1) ///
                     -exp(t1*(a1-a0)+t2*mstar+t3*a1*mstar)*(1+exp(b0+b1*a0+bcc)) ///
                     /(1+exp(b0+b1*a0+bcc+t2+t3*a0))+ exp(t2*mstar+t3*a0*mstar)* ///
                     (1+exp(b0+b1*a0+bcc))/(1+exp(b0+b1*a0+bcc+t2+t3*a0))
scalar INTmed_comp = exp(t1*(a1-a0))*(1+exp(b0+b1*a1+bcc+t2+t3*a1))* ///
                     (1+exp(b0+b1*a0+bcc))/((1+exp(b0+b1*a0+bcc+t2+t3*a0)) ///
                     *(1+exp(b0+b1*a1+bcc)))-(1+exp(b0+b1*a1+bcc+t2+t3*a0))* ///
                     (1+exp(b0+b1*a0+bcc)) /((1+exp(b0+b1*a0+bcc+t2+t3*a0))* ///
                     (1+exp(b0+b1*a1+bcc))) - exp(t1*(a1-a0))* ///
                     (1+exp(b0+b1*a0+bcc+t2+t3*a1)) ///
                      /(1+exp(b0+b1*a0+bcc+t2+t3*a0)) + (1)
scalar pie_comp    = (1+exp(b0+b1*a0+bcc))*(1+exp(b0+b1*a1+bcc+t2+t3*a0)) ///
                      /((1 + exp(b0+b1*a1+bcc))*(1+exp(b0+b1*a0+bcc+t2+t3*a0))) -(1)
scalar total       = (exp(t1*a1)*(1+exp(b0+b1*a0+bcc))* ///
                     (1+exp(b0+b1*a1+bcc+t2+t3*a1))) /(exp(t1*a0)* ///
                     (1+exp(b0+b1*a1+bcc))* (1+exp(b0+b1*a0+bcc+t2+t3*a0)))
```

```
// Retrieving the values of each coefficient calculated above

return scalar lnie=lnie
return scalar lnde=lnde
return scalar ltotal=ltotal

return scalar lcde=lcde
return scalar lIntref=lIntref
return scalar lIntmed=lIntmed
return scalar lpie=lpie

return scalar cde_comp = cde_comp
return scalar INTref_comp = INTref_comp
return scalar INTmed_comp = INTmed_comp
return scalar pie_comp  = pie_comp
return scalar total=total

// Calculating value for total excess relative risk (terr)

scalar terr   = cde_comp +INTref_comp + INTmed_comp + pie_comp

// Calculating the values for each of the 4 components of the total excess risk

scalar errCDE                    = cde_comp*(total-1)/terr
scalar errINTref                 = INTref_comp*(total-1)/terr
scalar errINTmed                 = INTmed_comp*(total-1)/terr
scalar errPIE                    = pie_comp*(total-1)/terr

// Assigning the values for proportions of total excess risk that is due to each of
// the component

scalar PropCDE                   = cde_comp/terr
scalar PropINTref                = INTref_comp/terr
scalar PropINTmed                = INTmed_comp/terr
scalar PropPIE                   = pie_comp/terr

// Assigning the values of overall proportions of risk attributable to mediation,
// interaction, and proportion eliminated

scalar PropMediated             = (pie_comp+INTmed_comp)/terr
scalar PropAttribInteraction    = (INTref_comp+INTmed_comp)/terr
scalar PropEliminated           = (INTref_comp+INTmed_comp+pie_comp)/terr

// Retrieving the values for each of the above

return scalar terr              = terr
return scalar errCDE             = errCDE
return scalar errINTref          = errINTref
return scalar errINTmed          = errINTmed
return scalar errPIE             = pie_comp
return scalar PropCDE            = PropCDE
return scalar PropINTref         = PropINTref
return scalar PropINTmed         = PropINTmed
return scalar PropPIE            = PropPIE
return scalar PropMediated                    = PropMediated
return scalar PropAttribInteraction = PropAttribInteraction
return scalar PropEliminated       = PropEliminated
end
calc2
return list

****End of code ***

***Run the code from start till the line above****
```

```
// Running the above code will give an output of estimates (coefficients) only, of all
// parameters
```

Code for calculating confidence intervals and exponentiated risk estimates

Stata codes for - bootstrap - procedure to calculate the 95% CIs for estimate of each parameter. Any number of repetition (reps) can be assigned. Random seed number (seed) should be assigned.

```
*Code:
bootstrap lcde=r(lcde) lIntref=r(lIntref) lIntmed = r(lIntmed)  lpie=r(lpie) ///
    ltotal=r(ltotal)  lnde=r(lnde) lnie=r(lnie), reps(2000) seed(438766) nodrop: calc2

// Running the above command will create an output of results with estimates of
// observed coefficients, Bootstrap standard error, z, P>|z| and 95% CI in a table of
// rows and 6 columns

// Steps to exponentiate the values of each observed coefficient and corresponding CIs
// in the results table

matrix T= r(table) // captures the real matrix returned by -bootstrap-
matrix list T

// Calculating the exponentiated results

display "CDE=" exp(T[1,1]), "LB=" exp(T[5,1]), "UB=" exp(T[6,1])
display "INTref=" exp(T[1,2]), "LB=" exp(T[5,2]), "UB=" exp(T[6,2])
display "INTmed=" exp(T[1,3]), "LB=" exp(T[5,3]), "UB=" exp(T[6,3])
display "PIE="exp(T[1,4]), "LB=" exp(T[5,4]), "UB=" exp(T[6,4])
display "TE=" exp(T[1,5]), "LB=" exp(T[5,5]), "UB=" exp(T[6,5])
display "NDE=" exp(T[1,6]), "LB=" exp(T[5,6]), "UB=" exp(T[6,6])
display "NIE=" exp(T[1,7]), "LB=" exp(T[5,7]), "UB=" exp(T[6,7])


// Calculating the CIs using bootstrap for 4- way components, 4 components of excess
// relative risks, and proportions

bootstrap cde_comp = r(cde_comp) INTref_comp = r(INTref_comp) ///
          INTmed_comp = r(INTmed_comp) pie_comp = r(pie_comp) ///
          terr = r(terr) errCDE = r(errCDE) errINTref = r(errINTref) ///
          errINTmed = r(errINTmed) errPIE = r(errPIE) PropCDE = r(PropCDE) ///
          PropINTref = r(PropINTref) PropINTmed = r(PropINTmed) ///
          PropPIE = r(PropPIE) PropMediated = r(PropMediated) ///
          PropAttribInteraction = r(PropAttribInteraction) ///
          PropEliminated =r(PropEliminated), reps(2000) seed(438766) ///
          saving(`boot_results') nodrop: calc2
```