Supplementary Web Appendix for: Complier stochastic direct effects: identification and robust estimation

Kara E. Rudolph, Oleg Sofrygin, Mark J. van der Laan

1 Identification proof

Proof.

$$\begin{aligned}
\Psi_{CSDE}(P) &\equiv \{E_0(E_0(E_{g^*_{M|0,W}}\{E_0(Y|W,Z,M)|W,Z\}W,A=1)|W) \\
&\quad -E_0(E_0(E_{g^*_{M|0,W}}\{E_0(Y|W,Z,M)|W,Z\}W,A=0)|W)\} \\
&\quad /\{E_0(E_0(Z|W,A=1)|W)-E_0(E_0(Z|W,A=0)|W)\} \\
\text{By assumption 1,} &\quad P(Z=z\mid W,A=a)=P(Z_a=z\mid W),\ \text{so} \\
&\equiv \{E_0(E_0(E_{g^*_{M|0,W}}\{E_0(Y_{g^*_{M|0,W}}\mid W,Z)\}\mid W,Z_1)\mid W) \\
&\quad -E_0(E_0(E_{g^*_{M|0,W}}\{E_0(Y_{g^*_{M|0,W}}\mid W,Z)\}\mid W,Z_0)\mid W)\} \\
&\quad /[E_0\{E_0(Z_1|W)-E_0(Z_0|W)\}] \\
&\equiv \{E_0(E_0(Y_{1,g^*_{M|0,W}}|W)-E_0(Y_{0,g^*_{M|0,W}}|W))\} \\
&\quad /\{E_0(E_0(Z_1|W)-E_0(Z_0|W))\} \\
&\equiv \frac{E_0(Y_{1,g^*_{M|0,W}}-Y_{0,g^*_{M|0,W}})}{E_0(Z_1-Z_0)} \\
&\equiv \{E_0(Y_{1,g^*_{M|0,W}}-Y_{0,g^*_{M|0,W}}|Z_1-Z_0=1)P(Z_1-Z_0=1) \\
&\quad +E_0(Y_{1,g^*_{M|0,W}}-Y_{0,g^*_{M|0,W}}|Z_1-Z_0=0)P(Z_1-Z_0=0) \\
&\quad +E_0(Y_{1,g^*_{M|0,W}}-Y_{0,g^*_{M|0,W}}|Z_1-Z_0=-1)P(Z_1-Z_0=-1)\} \\
&\quad /E_0(Z_1-Z_0) \\
\text{By assumption 2,} & \\
&\equiv \{E_0(Y_{1,g^*_{M|0,W}}-Y_{0,g^*_{M|0,W}}|Z_1-Z_0=1)P(Z_1-Z_0=1) \\
&\quad +E_0(Y_{1,g^*_{M|0,W}}-Y_{0,g^*_{M|0,W}}|Z_1-Z_0=-1)P(Z_1-Z_0=-1)\} \\
&\quad /E_0(Z_1-Z_0) \\
\text{By assumption 3,} & \\
&\equiv \{E_0(Y_{1,g^*_{M|0,W}}-Y_{0,g^*_{M|0,W}}|Z_1-Z_0=1)P(Z_1-Z_0=1)\} \\
&\quad /E_0(Z_1-Z_0) \\
&\quad Z_1-Z_0\in\{0,1\},\ \text{so} \\
&\equiv \{E_0(Y_{1,g^*_{M|0,W}}-Y_{0,g^*_{M|0,W}}|Z_1-Z_0=1)E(Z_1-Z_0)\} \\
&\quad /E_0(Z_1-Z_0) \\
&\equiv E_0(Y_{1,g^*_{M|0,W}}-Y_{0,g^*_{M|0,W}}|Z_1-Z_0=1)
\end{aligned}$$

By assumptions 4 and 5, we have that Ψ_{CSDE} is defined. □

2 Estimator modifications when there is also a direct effect of A on M

The complier stochastic direct effect estimand and estimation approaches we consider also work in the scenario where M may depend on A conditional on Z: $M=f(W,A,Z,U_M)$.

We describe differences in the estimator details for such a scenario here. In this alternative scenario, A is not an instrument for the total effect of Z on Y, and the estimation approach suggested by Joffe et al. (2008) would also be appropriate.

The true distribution P_0 of O can be factorized as

$$P_0(O) = P_0(Y|W, Z, M)P_0(M|W, A, Z)P_0(Z|W, A)P_0(A|W)P_0(W).$$

2.1 Inverse Probability of Treatment Weighted Estimator

The inverse probability of treatment weights for estimating Ψ_{SDE} are

$$IPTW_{SDE} = \frac{(2A-1)\hat{g}_{M|0,W}}{g_{A|W}g_{M|Z,A,W}}. \tag{1}$$

Let $g_{A,n}$ and $g_{M,n}$ be estimators of $g_{A|W} = P(A = a|W)$ and $g_{M|Z,A,W} = P(M = m|Z, A, W)$, respectively. $g_{A,n}$ can be estimated by predicted probabilities from a logistic regression model of A on W. One could use machine learning in model fitting but we will describe estimation in terms of parametric model fitting for simplicity. $g_{M,n}$ can be estimated by predicted probabilities from a logistic regression model of M on W, A, Z. $\hat{g}_{M|0,W}$ is treated as known, estimated from the observed data, marginalizing out $Z : \sum_{z=0}^{1} P(M = m|Z = z, A = 0, W)P(Z = z|A = 0, W)$ (VanderWeele and Tchetgen Tchetgen, 2017). The IPTW estimate of Ψ_{SDE} is the empirical mean of outcome, Y, weighted by an estimate of $IPTW_{SDE}$.

The inverse probability of treatment weights for estimating Ψ_{FS} are as written in the main text.

The associated variance can be estimated as the sample variance of the estimator's influence curve (IC), which is

$$D_{IPTW}(P) = \frac{D_{IPTW_{SDE}}(P)}{\Psi_{FS}(P)} - \frac{\Psi_{SDE}(P)D_{IPTW_{FS}}(P)}{\Psi^2_{FS}(P)}, \tag{2}$$

and where

$$D_{IPTW_{SDE}}(P) = \frac{(2A-1)\hat{g}_{M|0,W}}{g_{A|W}g_{M|Z,A,W}}Y - \Psi_{SDE} \tag{3}$$

and where

$$D_{IPTW_{FS}}(P) = \frac{2A-1}{g_{A|W}}Z - \Psi_{FS}. \tag{4}$$

2.2 Estimating Equation Estimator

This estimator solves the efficient influence curve (EIC) for Ψ_{CSDE}, which is given by

$$D_{CSDE}(P)(Q_W, g_A, g_Z, \bar{Q}) = \frac{D_{SDE}(P)}{\Psi_{FS}(P)} - \frac{\Psi_{SDE}(P)D_{FS}(P)}{\Psi^2_{FS}(P)}, \tag{5}$$

where

$$
\begin{aligned}
D_{SDE}(P) = & \left(\frac{g_{1|W,Z,M}}{g_{1|W}} - \frac{g_{0|W,Z,M}}{g_{0|W}} \right) \frac{\hat{g}_{M|A=0,W}}{g_{M|Z,A,W}} (Y - \bar{Q}_Y(M, Z, W)) \\
& + \frac{2A-1}{g_{A|W}} (\bar{Q}_M(Z=1, W) - \bar{Q}_M(Z=0, W))(Z - g_Z(1|A, W)) \\
& + (\bar{Q}_Z(A=1, W) - \bar{Q}_Z(A=0, W)) - \Psi_{SDE}
\end{aligned} \tag{6}
$$

and where

$$
D_{FS}(P) = \frac{2A-1}{g_{A|W}}(Z - g_Z(1|A, W)) + \{(g_Z(A=1, W) - g_Z(A=0, W)) - \Psi_{FS}\}. \tag{7}
$$

We first solve D_{SDE} to obtain the EE estimate of Ψ_{SDE}. We calculate the first component of D_{SDE} as follows. Let $g_M = P(M = m|Z, A, W), g_A = P(A = a|W)$, and $g_{A2} = P(A = a|W, Z, M)$. Recall that $\hat{g}_{M|0,W}$ is treated as known, estimated from the observed data, marginalizing out $Z : \sum_{z=0}^{1} P(M = m|Z = z, A = 0, W)P(Z = z|A = 0, W)$ (VanderWeele and Tchetgen Tchetgen, 2017). $g_{M,n}$ can be estimated by predicted probabilities from a logistic regression model of M on Z, A, and W. g_{A2} can be written $\frac{P(A=a|W)P(Z|a,W)P(M|Z,a,W)}{P(M,Z|W)} = \frac{g_{A|W}g_{Z|A,W}g_{M|Z,A,W}}{P(M,Z|W)}$, where $g_{A,n}$ and $g_{M,n}$ can be estimated as described above, $g_{Z,n}$ can be estimated from a logistic regression model of Z on A and W, and where an estimate of $P(Z, M|W)$ is obtained by marginalizing out A : $\left(\sum_{a=0}^{1} P(M = m|Z, A = a, W)P(A = a|W) \right) \left(\sum_{a=0}^{1} P(Z = z|A = a, W)P(A = a|W) \right)$, which can be rewritten in terms of the above estimators $\sum_{a=0}^{1} g_{M,n}g_{A,n} \sum_{a=0}^{1} g_{Z,n}g_{A,n}$. The other components can be calculated as described in the main text.

The second and third components of D_{SDE} and the components of D_{FS} are calculated as described in the main text. The associated variance can be estimated as the sample variance of the EIC, $D_{CSDE}(P)$, which is given in Equation 5.

2.3 Compatible Targeted Minimum Loss-Based Estimator

Recall $\bar{Q}_Y = E(Y|W, Z, M)$, $g_M = P(M = m|Z, A, W), g_A = P(A = a|W)$, and $g_{A2} = P(A = a|W, Z, M)$. Again, $\hat{g}_{M|0,W}$ is treated as known, estimated from the observed data, marginalizing out $Z : \sum_{z=0}^{1} P(M = m|Z = z, A = 0, W)P(Z = z|A = 0, W)$ (VanderWeele and Tchetgen Tchetgen, 2017). Consider submodel $\{\bar{Q}_{Y,n}(M, Z, W)(\epsilon) : \epsilon\}$ defined as: $logit(\bar{Q}_{Y,n}(\epsilon)(M, Z, W)) = logit(\bar{Q}_{Y,n}(M, Z, W)) + \epsilon C_Y$, where $C_Y = \left(\frac{g_{1|W,Z,M}}{g_{1|W}} - \frac{g_{0|W,Z,M}}{g_{0|W}} \right) \frac{\hat{g}_{M|A=0,W}}{g_{M|Z,A,W}}$.

The components of C_Y can be calculated as described in the above subsections and in the main text. The update step for $\bar{Q}_Y$ and the remaining steps for the TMLE estimator are completed as in the main text.

The TMLE solves the efficient influence curve (EIC) for Ψ_{CSDE} (shown in the previous subsection), replacing g_Z and $\bar{Q}_Y$ with g_Z^* and $\bar{Q}_Y^*$. The variance of the TMLE estimator of Ψ_{CSDE} is estimated as the sample variance of $D_{CSDE}(P)$.

3 R code

3.1 Code for ratio of Inverse Probability of Treatment Weighted Estimators

```
#This estimates the complier stochastic direct effect and its variance. It takes the following arguments:
# a is the instrument, 0/1. It is assumed to be exogenous, but the code can be modified to make it conditionally random.
# z is the exposure influenced by the instrument, 0/1. It is a function of a and w
# m is the mediator, 0/1. It is a function of z, w.
# y is the outcome, 0/1, but the code can be modified for any outcome type. It is a function of z, w, m.
# w is a matrix of covariates
# svywt is a vector of weights to be applied to the data.
# zmodel is the parametric model for z.
# mmodel is the parametric model for m.
# ymodel is the parametric model for y.
# qmodel is the parametric model for q.
# gm is the user-specified stochastic intervention on M, conditional on a=0 and w
# za, za1, and za0 are optional arguments that can be included if the user estimates these as part of the stochastic intervention. Otherwise, they are estimated within the function
# uses the constrained regression function if za, za1, and za0 are null

mediptw<-function(a, z, m, y, w, svywt, zmodel, mmodel, ymodel, qmodel, gm, za=NULL, za1=NULL, za0=NULL){

datw<-w

# estimate p(m | w, z)
mz<-predict(glm(formula=mmodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m))), newdata=data.frame(cbind(datw, z=z)), type="response")
mz0<-predict(glm(formula=mmodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m))), newdata=data.frame(cbind(datw, z=0)), type="response")
mz1<-predict(glm(formula=mmodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m))), newdata=data.frame(cbind(datw, z=1)), type="response")

# estimate p(z | w, a)
if(is.null(za) | is.null(za1) | is.null(za0)){
 zfit<-mle.logreg.constrained(formula(zmodel), data.frame(cbind(datw, a=a, z=z)))

```

```
  za0<-predictClogis(cbind(rep(0,nrow(data.frame(datw))), datw), zfit$beta)
  za1<-predictClogis(cbind(rep(1,nrow(data.frame(datw))), datw), zfit$beta)
  za<-predictClogis(data.frame(cbind(a=a, datw)), zfit$beta)
}
else {
  za<-za
  za1<-za1
  za0<-za0
}

pza1<-ifelse(z==1, za1, 1-za1)
pza0<-ifelse(z==1, za0, 1-za0)

# estimate p(a|w,m,z) using previous estimates. Note that p(a|w,m,z) = p(a|
    w,z) bc of exclusion restriction
pa1<-(mean(a)*pza1)/(pza1*mean(a) + pza0*mean(1-a))
pa1z0<-(mean(a)*(1-za1))/((1-za1)*mean(a) + (1-za0)*mean(1-a))
pa1z1<-(mean(a)*za1)/(za1*mean(a) + za0*mean(1-a))

  tmpdat<-data.frame(cbind(datw, a=a))

 #make clever covariate
  psm<-(mz*m) + ((1-mz)*(1-m))

  tmpdat$wts<-((m*gm + (1-m)*(1-gm))/psm)* svywt
  #component that can't go into the weights
  tmpdat$cc<- (pa1/mean(a)) - ((1-pa1)/mean(1-a))

  tmpdat$ccz0<- (pa1z0/mean(a)) - ((1-pa1z0)/mean(1-a))
  tmpdat$ccz1<- (pa1z1/mean(a)) - ((1-pa1z1)/mean(1-a))

  tmpdat$y<-y

  psi1<-sum(tmpdat$y * tmpdat$wts * tmpdat$cc)/sum(svywt)
  eicpsi1<-(tmpdat$cc*tmpdat$wts * tmpdat$y) - psi1

  #estimate denominator
  psi2<-sum(z * tmpdat$cc *svywt)/sum(svywt)
  eicpsi2<-(z * tmpdat$cc * svywt) - psi2

  csde<-psi1/psi2
  csdeeic<-(eicpsi1/psi2) - ((psi1*eicpsi2)/(psi2^2))
  varcsde<-var(csdeeic)/nrow(tmpdat)

 return(list("est"=csde, "var"=varcsde))
}
```

CSDE_iptw.R

3.2 Code for ratio of Estimating Equation Estimators

```r
#This estimates the complier stochastic direct effect and its variance. It takes the following arguments:
# a is the instrument, 0/1. It is assumed to be exogenous, but the code can be modified to make it conditionally random.
# z is the exposure influenced by the instrument, 0/1. It is a function of a and w
# m is the mediator, 0/1. It is a function of z, w.
# y is the outcome, 0/1, but the code can be modified for any outcome type. It is a function of z, w, m.
# w is a matrix of covariates
# svywt is a vector of weights to be applied to the data.
# zmodel is the parametric model for z.
# mmodel is the parametric model for m.
# ymodel is the parametric model for y.
# qmodel is the parametric model for q.
# gm is the user-specified stochastic intervention on M, conditional on a=0 and w
# za, za1, and za0 are optional arguments that can be included if the user estimates these as part of the stochastic intervention. Otherwise, they are estimated within the function
# uses the constrained regression function if za, za1, and za0 are null

medee<-function(a, z, m, y, w, svywt, zmodel, mmodel, ymodel, qmodel, gm, za=NULL, za1=NULL, za0=NULL){

datw<-w

# estimate p(m | w, z)
mz<-predict(glm(formula=mmodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m))), newdata=data.frame(cbind(datw, z=z)), type="response")
mz0<-predict(glm(formula=mmodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m))), newdata=data.frame(cbind(datw, z=0)), type="response")
mz1<-predict(glm(formula=mmodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m))), newdata=data.frame(cbind(datw, z=1)), type="response")

# estimate p(z | w, a)
if(is.null(za) | is.null(za1) | is.null(za0)){
 zfit<-mle.logreg.constrained(formula(zmodel), data.frame(cbind(datw, a=a, z=z)))

  za0<-predictClogis(cbind(rep(0,nrow(data.frame(datw))), datw), zfit$beta)
  za1<-predictClogis(cbind(rep(1,nrow(data.frame(datw))), datw), zfit$beta)
  za<-predictClogis(data.frame(cbind(a=a, datw)), zfit$beta)
}
else {
  za<-za
  za1<-za1
  za0<-za0
}

```

```r
pza1<-ifelse(z==1, za1, 1-za1)
pza0<-ifelse(z==1, za0, 1-za0)

# estimate p(a|w,m,z) using previous estimates. Note that p(a|w,m,z) = p(a|w,z) bc of exclusion restriction
pa1<-(mean(a)*pza1)/(pza1*mean(a) + pza0*mean(1-a))
pa1z0<-(mean(a)*(1-za1))/((1-za1)*mean(a) + (1-za0)*mean(1-a))
pa1z1<-(mean(a)*za1)/(za1*mean(a) + za0*mean(1-a))

  tmpdat<-data.frame(cbind(datw, a=a))

 #get initial Y fit
  yfit<-glm(formula=ymodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m, y=y)))
  tmpdat$qyinit<-cbind(predict(yfit, newdata=data.frame(cbind(datw, z=z, m=m)), type="response"),
    predict(yfit, newdata=data.frame(cbind(datw, z=z, m=1)), type="response"),
    predict(yfit, newdata=data.frame(cbind(datw, z=z, m=0)), type="response") )
  tmpdat$qyinitz0<-cbind(predict(yfit, newdata=data.frame(cbind(datw, z=0, m=1)), type="response"), predict(yfit, newdata=data.frame(cbind(datw, z=0, m=0)), type="response") )
  tmpdat$qyinitz1<-cbind(predict(yfit, newdata=data.frame(cbind(datw, z=1, m=1)), type="response"), predict(yfit, newdata=data.frame(cbind(datw, z=1, m=0)), type="response") )

#make clever covariate
  psm<-(mz*m) + ((1-mz)*(1-m))

  tmpdat$wts<-((m*gm + (1-m)*(1-gm))/psm)* svywt
  #component that can't go into the weights
  tmpdat$cc<- (pa1/mean(a)) - ((1-pa1)/mean(1-a))

  tmpdat$ccz0<- (pa1z0/mean(a)) - ((1-pa1z0)/mean(1-a))
  tmpdat$ccz1<- (pa1z1/mean(a)) - ((1-pa1z1)/mean(1-a))

  tmpdat$y<-y

  epsilon<-coef(glm(y ~ -1 + offset(qlogis(qyinit[,1])) +cc ,weights=wts, family="quasibinomial", data=tmpdat)) #

  eic1<-(tmpdat$cc)*tmpdat$wts * (tmpdat$y - tmpdat$qyinit[,1])

  #integrate out M to get qm
  tmpdat$qm<-tmpdat$qyinit[,2]*gm + tmpdat$qyinit[,3]*(1-gm)
  tmpdat$qmz0<-tmpdat$qyinitz0[,1]*gm + tmpdat$qyinitz0[,2]*(1-gm)
  tmpdat$qmz1<-tmpdat$qyinitz1[,1]*gm + tmpdat$qyinitz1[,2]*(1-gm)

  #initial fit gz
  gz<-cbind(za, za0, za1)

  #make components for second targeting step
```

```
  tmpdat$difqmza<-tmpdat$qmz1 - tmpdat$qmz0

  tmpdat$ga<-ifelse(a==1, mean(a), mean(1-a))
  tmpdat$a<-a
  tmpdat$nota<-1-a

  #integrate out z to get qz
  qz<-cbind((tmpdat$qmz1*gz[,2]) + (tmpdat$qmz0*(1-gz[,2])), (tmpdat$qmz1*
      gz[,3]) + (tmpdat$qmz0*(1-gz[,3])))

  #estimate numerator
  eic2<-((2*a-1)/tmpdat$ga)*svywt*(tmpdat$qmz1 - tmpdat$qmz0) *(z-gz[,1])
  #eic2<-(tmpdat$a*svywt*(tmpdat$qmz1 - tmpdat$qmz0) + tmpdat$nota*svywt*(
      tmpdat$qmz1 - tmpdat$qmz0))*(z-gz[,1])

  #estimate denominator
  eic3<-((qz[,2] - qz[,1])*svywt)
  psi1<-mean(eic1 + eic2+ eic3)
  eicdp1<-eic1 + eic2 + eic3 -psi1

  eic1dp2<-((2*a-1)/tmpdat$ga)*svywt*(z - gz[,1])
  eic2dp2<-((gz[,3] - gz[,2])*svywt)
  psi2<-mean(eic1dp2 + eic2dp2)
  eicdp2<-eic1dp2 + eic2dp2 - psi2


  csde<-psi1/psi2
  csdeeic<-(eicdp1/psi2) - ((psi1*eicdp2)/(psi2^2))
  varcsde<-var(csdeeic)/nrow(tmpdat)

 return(list("est"=csde, "var"=varcsde))
}
```

CSDE_ee.R

3.3 Code for ratio of Targeted Minimum Loss-based Estimators (Efficient TMLE)

```
#This estimates the complier stochastic direct effect and its variance. It
    takes the following arguments:
# a is the instrument, 0/1. It is assumed to be exogenous, but the code can
    be modified to make it conditionally random.
# z is the exposure influenced by the instrument, 0/1. It is a function of
    a and w
# m is the mediator, 0/1. It is a function of z, w.
# y is the outcome, 0/1, but the code can be modified for any outcome type.
    It is a function of z, w, m.
# w is a matrix of covariates
# svywt is a vector of weights to be applied to the data.
# zmodel is the parametric model for z.
# mmodel is the parametric model for m.
# ymodel is the parametric model for y.
# qmodel is the parametric model for q.
```

```
# gm is the user-specified stochastic intervention on M, conditional on a=0 and w
# za, za1, and za0 are optional arguments that can be included if the user estimates these as part of the stochastic intervention. Otherwise, they are estimated within the function
# uses the constrained regression function if za, za1, and za0 are null

medtmle<-function(a, z, m, y, w, svywt, zmodel, mmodel, ymodel, qmodel, gm, za=NULL, za1=NULL, za0=NULL){

datw<-w

# estimate p(m | w, z)
mz<-predict(glm(formula=mmodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m))), newdata=data.frame(cbind(datw, z=z)), type="response")
mz0<-predict(glm(formula=mmodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m))), newdata=data.frame(cbind(datw, z=0)), type="response")
mz1<-predict(glm(formula=mmodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m))), newdata=data.frame(cbind(datw, z=1)), type="response")

# estimate p(z | w, a)
if(is.null(za) | is.null(za1) | is.null(za0)){
 zfit<-mle.logreg.constrained(formula(zmodel), data.frame(cbind(datw, a=a, z=z)))

  za0<-predictClogis(cbind(rep(0,nrow(data.frame(datw))), datw), zfit$beta)
  za1<-predictClogis(cbind(rep(1,nrow(data.frame(datw))), datw), zfit$beta)
  za<-predictClogis(data.frame(cbind(a=a, datw)), zfit$beta)
}
else {
  za<-za
  za1<-za1
  za0<-za0
}

pza1<-ifelse(z==1, za1, 1-za1)
pza0<-ifelse(z==1, za0, 1-za0)

# estimate p(a|w,m,z) using previous estimates. Note that p(a|w,m,z) = p(a|w,z) bc of exclusion restriction
pa1<-(mean(a)*pza1)/(pza1*mean(a) + pza0*mean(1-a))
pa1z0<-(mean(a)*(1-za1))/((1-za1)*mean(a) + (1-za0)*mean(1-a))
pa1z1<-(mean(a)*za1)/(za1*mean(a) + za0*mean(1-a))

  tmpdat<-data.frame(cbind(datw, a=a))

 #get initial Y fit
  yfit<-glm(formula=ymodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m, y=y)))
```

```
  tmpdat$qyinit<-cbind(predict(yfit, newdata=data.frame(cbind(datw, z=z, m=m)), type="response"),
    predict(yfit, newdata=data.frame(cbind(datw, z=z, m=1)), type="response"),
    predict(yfit, newdata=data.frame(cbind(datw, z=z, m=0)), type="response") )
  tmpdat$qyinitz0<-cbind(predict(yfit, newdata=data.frame(cbind(datw, z=0, m=1)), type="response"), predict(yfit, newdata=data.frame(cbind(datw, z=0, m=0)), type="response") )
  tmpdat$qyinitz1<-cbind(predict(yfit, newdata=data.frame(cbind(datw, z=1, m=1)), type="response"), predict(yfit, newdata=data.frame(cbind(datw, z=1, m=0)), type="response") )

#make clever covariate
  psm<-(mz*m) + ((1-mz)*(1-m))

  tmpdat$wts<-((m*gm + (1-m)*(1-gm))/psm)* svywt
  #component that can't go into the weights
  tmpdat$cc<- (pa1/mean(a))  - ((1-pa1)/mean(1-a))

  tmpdat$ccz0<- (pa1z0/mean(a))  - ((1-pa1z0)/mean(1-a))
  tmpdat$ccz1<- (pa1z1/mean(a))  - ((1-pa1z1)/mean(1-a))

  tmpdat$y<-y

  epsilon<-coef(glm(y ~  -1 + offset(qlogis(qyinit[,1])) +cc  ,weights=wts, family="quasibinomial", data=tmpdat))  #
 epsilon<-ifelse(is.na(epsilon), 0, epsilon)

 #update Qy
  tmpdat$qyup<-plogis(qlogis(tmpdat$qyinit) + epsilon*(tmpdat$cc))
  tmpdat$qyupz0m0<-plogis(qlogis(tmpdat$qyinitz0[,2]) + epsilon * tmpdat$ccz0)
  tmpdat$qyupz0m1<-plogis(qlogis(tmpdat$qyinitz0[,1]) + epsilon * tmpdat$ccz0)
  tmpdat$qyupz1m0<-plogis(qlogis(tmpdat$qyinitz1[,2]) + epsilon * tmpdat$ccz1)
  tmpdat$qyupz1m1<-plogis(qlogis(tmpdat$qyinitz1[,1]) + epsilon * tmpdat$ccz1)

  eic1<-(tmpdat$cc)*tmpdat$wts * (tmpdat$y - tmpdat$qyup[,1])

  #integrate out M to get qm
  tmpdat$qm<-tmpdat$qyup[,2]*gm  + tmpdat$qyup[,3]*(1-gm)
  tmpdat$qmz0<-tmpdat$qyupz0m1*gm  + tmpdat$qyupz0m0*(1-gm)
  tmpdat$qmz1<-tmpdat$qyupz1m1*gm  + tmpdat$qyupz1m0*(1-gm)

  #initial fit gz
  gz<-cbind(za, za0, za1)

  #make components for second targeting step
  tmpdat$difqmza<-tmpdat$qmz1 - tmpdat$qmz0

```

```
  tmpdat$ga<-ifelse(a==1, mean(a), mean(1-a))
  tmpdat$a<-a
  tmpdat$nota<-1-a

  fitcz<- glm(z ~ -1 + a:difqmza + nota:difqmza, weights=svywt*(1/tmpdat$ga), family="quasibinomial", data=tmpdat, offset=qlogis(gz[,1]))

  epsiloncz<-coef(fitcz)

  #update gz
  gzup<-cbind(plogis(qlogis(gz[,1]) + I(tmpdat$a==0)*epsiloncz[2]*tmpdat$difqmza + I(tmpdat$a==1)*epsiloncz[1]*tmpdat$difqmza),
    plogis(qlogis(gz[,2]) +  epsiloncz[2]*tmpdat$difqmza),
    plogis(qlogis(gz[,3]) + epsiloncz[1]*tmpdat$difqmza)
    )

  #integrate out z to get qz
  qz<-cbind((tmpdat$qmz1*gzup[,2]) + (tmpdat$qmz0*(1-gzup[,2])), (tmpdat$qmz1*gzup[,3]) + (tmpdat$qmz0*(1-gzup[,3])))

  #estimate numerator
  psi1<-sum((qz[,2]-qz[,1])*svywt)/sum(svywt)

  #eic2<-(tmpdat$a*svywt*(tmpdat$qmz1 - tmpdat$qmz0) + tmpdat$nota*svywt*(tmpdat$qmz1 - tmpdat$qmz0))*(z-gzup[,1])
  eic2<-((2*tmpdat$a-1)/tmpdat$ga)*svywt*(tmpdat$qmz1 - tmpdat$qmz0) *(z-gzup[,1])
  #target gz for denominator
  fitczd<- glm(z ~ -1 + a + nota, weights=svywt*(1/tmpdat$ga), family="quasibinomial", data=tmpdat, offset=qlogis(gz[,1]))
  epsiloinczd<-coef(fitczd)
  gzupd<-cbind(plogis(qlogis(gz[,1]) + I(tmpdat$a==0)*epsilonczd[2] + I(tmpdat$a==1)*epsilonczd[1]),
    plogis(qlogis(gz[,2]) +  epsilonczd[2]),
    plogis(qlogis(gz[,3]) + epsilonczd[1])
    )
  #estimate denominator
  psi2<-sum((gzupd[,3]-gzupd[,2])*svywt)/sum(svywt)


  eic3<-((qz[,2] - qz[,1])*svywt) - psi1
  eicdp1<-eic1 + eic2+ eic3

  eic1dp2<-((2*a-1)/tmpdat$ga)*svywt*(z - gz[,1])
  eic2dp2<-((gzupd[,3] - gzupd[,2])*svywt) - psi2
  eicdp2<-eic1dp2 + eic2dp2

  csde<-psi1/psi2
  csdeeic<-(eicdp1/psi2) - ((psi1*eicdp2)/(psi2^2))
  varcsde<-var(csdeeic)/nrow(tmpdat)

 return(list("est"=csde, "var"=varcsde))
}
```

CSDE_tmle_numdenom.R

3.4 Code for Targeted Minimum Loss-based Estimator that estimates ratio directly (Compatible TMLE)

```r
#This estimates the complier stochastic direct effect and its variance. It takes the following arguments:
# a is the instrument, 0/1. It is assumed to be exogenous, but the code can be modified to make it conditionally random.
# z is the exposure influenced by the instrument, 0/1. It is a function of a and w
# m is the mediator, 0/1. It is a function of z, w.
# y is the outcome, 0/1, but the code can be modified for any outcome type. It is a function of z, w, m.
# w is a matrix of covariates
# svywt is a vector of weights to be applied to the data.
# zmodel is the parametric model for z.
# mmodel is the parametric model for m.
# ymodel is the parametric model for y.
# qmodel is the parametric model for q.
# gm is the user-specified stochastic intervention on M, conditional on a=0 and w
# za, za1, and za0 are optional arguments that can be included if the user estimates these as part of the stochastic intervention. Otherwise, they are estimated within the function
# uses the constrained regression function if za, za1, and za0 are null

medtmle<-function(a, z, m, y, w, svywt, zmodel, mmodel, ymodel, qmodel, gm, za=NULL, za1=NULL, za0=NULL){

datw<-w

# estimate p(m | w, z)
mz<-predict(glm(formula=mmodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m))), newdata=data.frame(cbind(datw, z=z)), type="response")
mz0<-predict(glm(formula=mmodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m))), newdata=data.frame(cbind(datw, z=0)), type="response")
mz1<-predict(glm(formula=mmodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m))), newdata=data.frame(cbind(datw, z=1)), type="response")

# estimate p(z | w, a)
if(is.null(za) | is.null(za1) | is.null(za0)){
 zfit<-mle.logreg.constrained(formula(zmodel), data.frame(cbind(datw, a=a, z=z)))

  za0<-predictClogis(cbind(rep(0,nrow(data.frame(datw))), datw), zfit$beta)
  za1<-predictClogis(cbind(rep(1,nrow(data.frame(datw))), datw), zfit$beta)
```

```r
  za<-predictClogis(data.frame(cbind(a=a, datw)), zfit$beta)
}
else {
  za<-za
  za1<-za1
  za0<-za0
}

pza1<-ifelse(z==1, za1, 1-za1)
pza0<-ifelse(z==1, za0, 1-za0)

# estimate p(a|w,m,z) using previous estimates. Note that p(a|w,m,z) = p(a|w,z) bc of exclusion restriction
pa1<-(mean(a)*pza1)/(pza1*mean(a) + pza0*mean(1-a))
pa1z0<-(mean(a)*(1-za1))/((1-za1)*mean(a) + (1-za0)*mean(1-a))
pa1z1<-(mean(a)*za1)/(za1*mean(a) + za0*mean(1-a))

  tmpdat<-data.frame(cbind(datw, a=a))

 #get initial Y fit
  yfit<-glm(formula=ymodel, family="binomial", data=data.frame(cbind(datw, z=z, m=m, y=y)))
  tmpdat$qyinit<-cbind(predict(yfit, newdata=data.frame(cbind(datw, z=z, m=m)), type="response"),
     predict(yfit, newdata=data.frame(cbind(datw, z=z, m=1)), type="response"),
     predict(yfit, newdata=data.frame(cbind(datw, z=z, m=0)), type="response") )
  tmpdat$qyinitz0<-cbind(predict(yfit, newdata=data.frame(cbind(datw, z=0, m=1)), type="response"), predict(yfit, newdata=data.frame(cbind(datw, z=0, m=0)), type="response") )
  tmpdat$qyinitz1<-cbind(predict(yfit, newdata=data.frame(cbind(datw, z=1, m=1)), type="response"), predict(yfit, newdata=data.frame(cbind(datw, z=1, m=0)), type="response") )

#make clever covariate
  psm<-(mz*m) + ((1-mz)*(1-m))

  tmpdat$wts<-((m*gm + (1-m)*(1-gm))/psm)* svywt
  #component that can't go into the weights
  tmpdat$cc<- (pa1/mean(a))  - ((1-pa1)/mean(1-a))

  tmpdat$ccz0<- (pa1z0/mean(a))  - ((1-pa1z0)/mean(1-a))
  tmpdat$ccz1<- (pa1z1/mean(a))  - ((1-pa1z1)/mean(1-a))

  tmpdat$y<-y

  epsilon<-coef(glm(y ~  -1 + offset(qlogis(qyinit[,1])) +cc  ,weights=wts, family="quasibinomial", data=tmpdat))  #

 #update Qy
  tmpdat$qyup<-plogis(qlogis(tmpdat$qyinit) + epsilon*(tmpdat$cc))
```

```
tmpdat$qyupz0m0<-plogis(qlogis(tmpdat$qyinitz0[,2]) + epsilon * tmpdat$
    ccz0)
tmpdat$qyupz0m1<-plogis(qlogis(tmpdat$qyinitz0[,1]) + epsilon * tmpdat$
    ccz0)
tmpdat$qyupz1m0<-plogis(qlogis(tmpdat$qyinitz1[,2]) + epsilon * tmpdat$
    ccz1)
tmpdat$qyupz1m1<-plogis(qlogis(tmpdat$qyinitz1[,1]) + epsilon * tmpdat$
    ccz1)

eic1<-(tmpdat$cc)*tmpdat$wts * (tmpdat$y - tmpdat$qyup[,1])

#integrate out M to get qm
tmpdat$qm<-tmpdat$qyup[,2]*gm  + tmpdat$qyup[,3]*(1-gm)
tmpdat$qmz0<-tmpdat$qyupz0m1*gm  + tmpdat$qyupz0m0*(1-gm)
tmpdat$qmz1<-tmpdat$qyupz1m1*gm  + tmpdat$qyupz1m0*(1-gm)

#initial fit gz
gz<-cbind(za, za0, za1)

#make components for second targeting step
tmpdat$difqmza<-tmpdat$qmz1 - tmpdat$qmz0

tmpdat$ga<-ifelse(a==1, mean(a), mean(1-a))
tmpdat$a<-a
tmpdat$nota<-1-a

fitcz<- glm(z ~ -1 + a + nota + a:difqmza + nota:difqmza, weights=svywt*
    (1/tmpdat$ga), family="quasibinomial", data=tmpdat, offset=qlogis(gz
    [,1]))

epsiloncz<-coef(fitcz)

#update gz
gzup<-cbind(plogis(qlogis(gz[,1]) + I(tmpdat$a==0)*epsiloncz[2] + I(
    tmpdat$a==0)*epsiloncz[4]*tmpdat$difqmza + I(tmpdat$a==1)*epsiloncz
    [1] + I(tmpdat$a==1)*epsiloncz[3]*tmpdat$difqmza),
  plogis(qlogis(gz[,2]) + epsiloncz[2] + epsiloncz[4]*tmpdat$difqmza),
  plogis(qlogis(gz[,3]) + epsiloncz[1] + epsiloncz[3]*tmpdat$difqmza)
  )

#integrate out z to get qz
qz<-cbind((tmpdat$qmz1*gzup[,2]) + (tmpdat$qmz0*(1-gzup[,2])), (tmpdat$
    qmz1*gzup[,3]) + (tmpdat$qmz0*(1-gzup[,3])))

#estimate numerator
psi1<-sum((qz[,2]-qz[,1])*svywt)/sum(svywt)

eic2<-((2*a-1)/tmpdat$ga)*svywt*(tmpdat$qmz1 - tmpdat$qmz0) *(z-gzup[,1])
#estimate denominator
psi2<-sum((gzup[,3]-gzup[,2])*svywt)/sum(svywt)

eic3<-((qz[,2] - qz[,1])*svywt) - psi1
eicdp1<-eic1 + eic2+ eic3
```

```r

  eic1dp2<-((2*a-1)/tmpdat$ga)*svywt*(z - gz[,1])
  eic2dp2<-((gzup[,3] - gzup[,2])*svywt) - psi2
  eicdp2<-eic1dp2 + eic2dp2

  csde<-psi1/psi2
  csdeeic<-(eicdp1/psi2) - ((psi1*eicdp2)/(psi2^2))
  varcsde<-var(csdeeic)/nrow(tmpdat)

 return(list("est"=csde, "var"=varcsde))
}
```

CSDE_tmle.R

4 Figures for stochastic direct and indirect effects

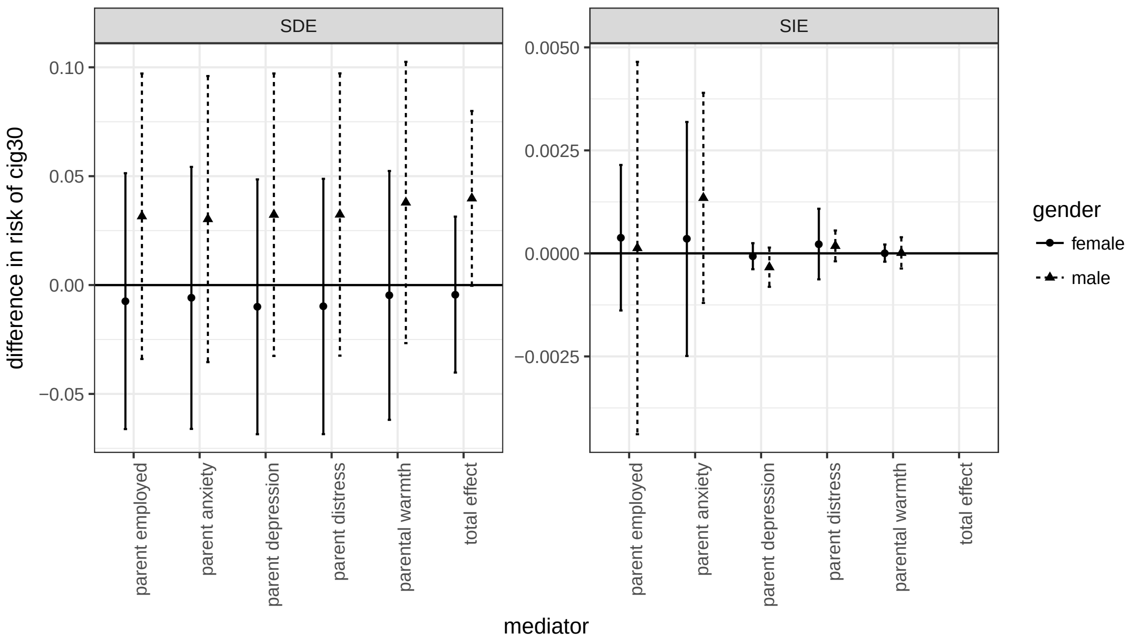

Figure 1: Data-dependent stochastic direct and indirect effect estimates and 95% confidence intervals on past-month cigarette use by mediator. Data from the Moving to Opportunity experiment, interim follow up.

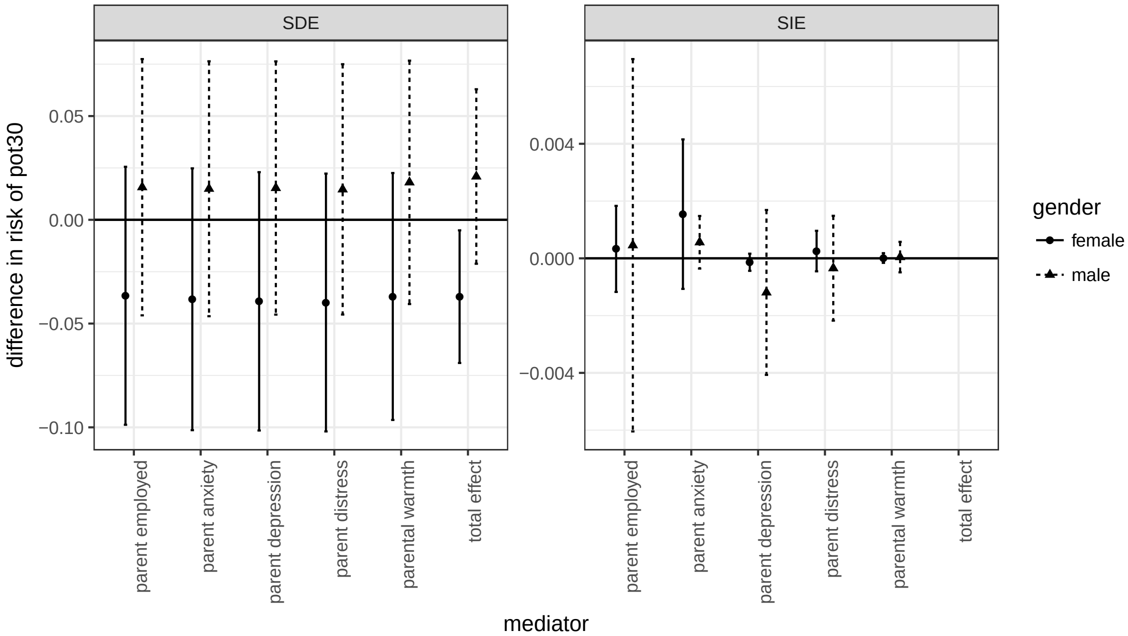

Figure 2: Data-dependent stochastic direct and indirect effect estimates and 95% confidence intervals on past-month marijuana use by mediator. Data from the Moving to Opportunity experiment, interim follow up.

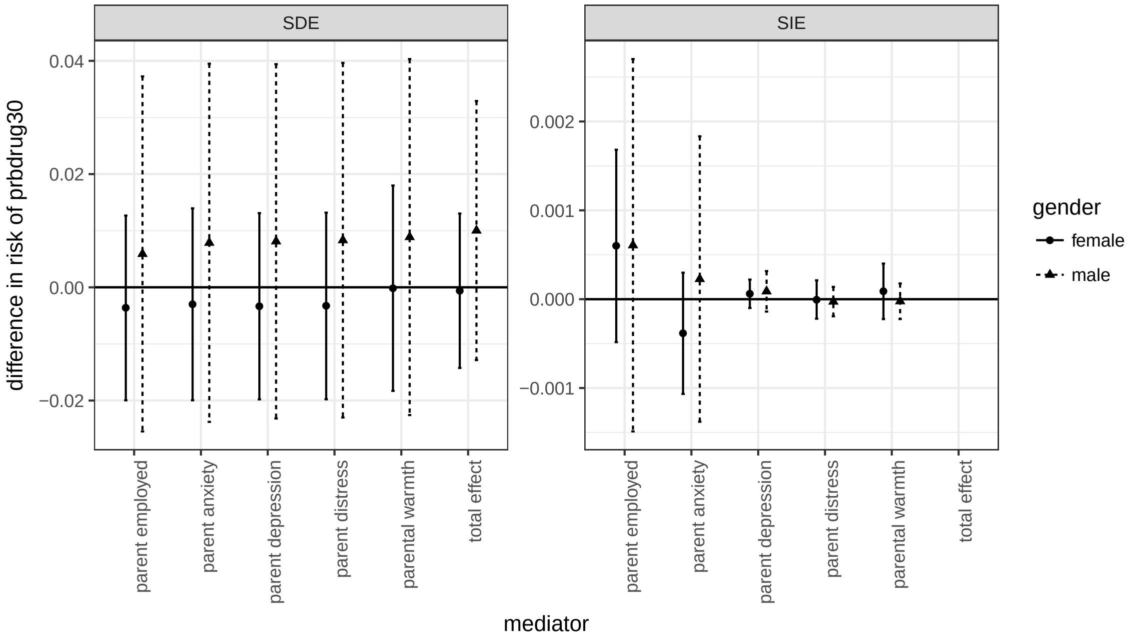

Figure 3: Data-dependent stochastic direct and indirect effect estimates and 95% confidence intervals on past-month problematic drug use by mediator. Data from the Moving to Opportunity experiment, interim follow up.

Bibliography

Joffe, M. M., Small, D., Ten Have, T., Brunelli, S. and Feldman, H. I. (2008) Extended instrumental variables estimation for overall effects. *The international journal of biostatistics*, **4**.

VanderWeele, T. J. and Tchetgen Tchetgen, E. J. (2017) Mediation analysis with time varying exposures and mediators. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **79**, 917–938.