This R code performs a single simulation run as reported in the associated paper "Effect Estimation using Structural Nested Models and G-estimation". In addition, we illustrate how the 'standard' sandwich estimator may be constructed, and demonstrate use of our R command.

```
set.seed(1)

# Generate data

# Sample size

n <- 1000

# Stage 1 data:

X1 <- rnorm(n)

A1 <- rbinom(n,1,1/(1+exp(-X1)))

# Stage 2 data:

X2 <- A1 + rnorm(n)

A2 <- X2 + rnorm(n)

# Stage 3 data:

X3 <- A2 + rnorm(n)

A3 <- rbinom(n,1,1/(1+exp(-X3)))

# Outcome Y = 'baseline' outcome + treatment effects (blips)

Y <- 1 + X1 + A1*(1+X1) + A2*(1+X2) + A3*(1+X3) + rnorm(n)



# G-estimation

# stage 3 (binary)

# treatment model correct:

H3.alpha <- cbind(rep(1,n),X3)
```

```
# estimate treatment model

A3hat <- fitted(glm(A3~0+H3.alpha,binomial))

# pseudo-outcome

Y3 <- Y

# treatment-free model incorrect

H3.beta <- cbind(rep(1,n))

# blip model

H3.psi <- cbind(rep(1,n),X3)

# w-matrix for analysis

W3 <- (diag(A3-A3hat) - (A3-A3hat)*H3.beta %*% solve(t(H3.beta) %*% H3.beta) %*%
t(H3.beta))

# estimate parameters

psi3 <- solve(t(H3.psi) %*% W3 %*% (A3*H3.psi)) %*% t(H3.psi) %*% W3 %*% Y3



# stage 2 (continuous)

# treatment model incorrect

H2.alpha <- rep(1,n)

# estimate treatment model

A2hat <- fitted(lm(A2~0+H2.alpha))

# pseudo-outcome

Y2 <- Y - A3*(H3.psi %*% psi3)

# treatment-free model correct

H2.beta <- cbind(rep(1,n),X1,A1,A1*X1)

# blip model
```

```
H2.psi <- cbind(rep(1,n),X2)

# w-matrix for analysis

W2 <- (diag(A2-A2hat) - (A2-A2hat)*H2.beta %*% solve(t(H2.beta) %*% H2.beta) %*%
t(H2.beta))

# estimate parameters

psi2 <- solve(t(H2.psi) %*% W2 %*% (A2*H2.psi)) %*% t(H2.psi) %*% W2 %*% Y2



# stage 1 (binary)

# treatment model incorrect

H1.alpha <- rep(1,n)

# estimate treatment model

A1hat <- fitted(glm(A1~0+H1.alpha,binomial))

# pseudo-outcome

Y1 <- Y2 - A2*(H2.psi %*% psi2)

# treatment-free incorrect

H1.beta <- cbind(rep(1,n))

# blip model

H1.psi <- cbind(rep(1,n),X1)

# weight-like matrix for analysis

W1 <- (diag(A1-A1hat) - (A1-A1hat)*H1.beta %*% solve(t(H1.beta) %*% H1.beta) %*%
t(H1.beta))

# estimate parameters

psi1 <- solve(t(H1.psi) %*% W1 %*% (A1*H1.psi)) %*% t(H1.psi) %*% W1 %*% Y1
```

```
## ADDENDUM: sandwich estimation of parameter variance

# stage 3:

# bread = E[t(H3.psi)*A3*W3*H3.psi]

B3 <- (1/n)*(t(H3.psi)%*%W3%*%(A3*H3.psi))

# filling = E[U3*t(U3)]

# where U3 is our estimating equation

# for this, construct beta3 and diagonal matrix D3 with entries (A3-A3hat)

beta3 <- solve(t(H3.beta) %*% H3.beta) %*% t(H3.beta) %*% (Y3 - A3*H3.psi%*%psi3)

D3 <- diag(A3-A3hat)

U3 <- t(D3 %*% H3.psi)%*%diag(as.vector(Y3 - A3*H3.psi %*% psi3 - H3.beta %*% beta3))

F3 <- (1/n)*U3%*%t(U3)

# sandwich covariance matrix = (B^-1) * F * t(B)^-1

covmat3 <- (1/n) * solve(B3) %*% F3 %*% solve(t(B3))

# standard error estimate for psi3 then the square root of the corresponding diagonal elements of covmat3

se3 <- sqrt(tail(diag(covmat3),2))


# stage 2:

# bread = E[t(H2.psi)*A2*W2*H2.psi]

B2 <- (1/n)*(t(H2.psi)%*%W2%*%(A2*H2.psi))

# filling = E[U2*t(U2)]

# where U2 is our estimating equation

# for this, construct beta2 and diagonal matrix D2 with entries (A2-A2hat)
```

```
beta2 <- solve(t(H2.beta) %*% H2.beta) %*% t(H2.beta) %*% (Y2 - A2*H2.psi%*%psi2)

D2 <- diag(A2-A2hat)

U2 <- t(D2 %*% H2.psi)%*%diag(as.vector(Y2 - A2*H2.psi %*% psi2 - H2.beta %*% beta2))

F2 <- (1/n)*U2%*%t(U2)

# sandwich covariance matrix = (B^-1) * F * t(B)^-1

covmat2 <- (1/n) * solve(B2) %*% F2 %*% solve(t(B2))

# standard error estimate for psi2 then the square root of the corresponding diagonal elements of covmat2

se2 <- sqrt(tail(diag(covmat2),2))



# stage 1:

# bread = E[t(H1.psi)*A1*W1*H1.psi]

B1 <- (1/n)*(t(H1.psi)%*%W1%*%(A1*H1.psi))

# filling = E[U1*t(U1)]

# where U1 is our estimating equation

# for this, construct beta1 and diagonal matrix D1 with entries (A1-A1hat)

beta1 <- solve(t(H1.beta) %*% H1.beta) %*% t(H1.beta) %*% (Y1 - A1*H1.psi%*%psi1)

D1 <- diag(A1-A1hat)

U1 <- t(D1 %*% H1.psi)%*%diag(as.vector(Y1 - A1*H1.psi %*% psi1 - H1.beta %*% beta1))

F1 <- (1/n)*U1%*%t(U1)

# sandwich covariance matrix = (B^-1) * F * t(B)^-1

covmat1 <- (1/n) * solve(B1) %*% F1 %*% solve(t(B1))

# standard error estimate for psi1 then the square root of the corresponding diagonal elements of covmat1
```

```
se1 <- sqrt(tail(diag(covmat1),2))



# Alternatively, we can use DTRreg directly:

# NB: DTRreg code contained in .doc file DTRreg.doc to comply with Editorial Manager file type restrictions

source("DTRreg.r")



# blip model

blip.mod <- list(~X1,~X2,~X3)

# treatment model

treat.mod <- list(A1~1,A2~1,A3~X3)

# treatment-free model

tf.mod <- list(~1,~X1+A1+A1:X1,~1)

# analysis

mod <- DTRreg(Y, blip.mod, treat.mod, tf.mod,var.est="sandwich",type="SNM")

# full results in summary(mod)

# point estimates stored in coef(mod)
```