Outcome Analysis of "Triplet Matching for Estimating Causal Effects with Three Treatment Arms: A Comparative Study of Mortality by Trauma Center Level"

Giovanni Nattino

November 5, 2018

Analysis of NEDS Data

Data preprocessing.

```
library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 3.4.3
```

```
library(survival)

#Load matched data
load("bestResult_3wayconstr_allData_1-2.Rdata")
matchedDat <- result3wayConstr$matchedData[!is.na(result3wayConstr$matchedData$indexMatch),]

#Data for evidence factor 1: NTC vs. TC
matchedDat$NTC <- (matchedDat$HOSP_TRAUMA %in% c(0))*1
matchedDat$TCI <- (matchedDat$HOSP_TRAUMA %in% c(1))*1

table(matchedDat$NTC, matchedDat$ED_death, dnn = c("NTC","Mortality"))
```

```
##    Mortality
## NTC    0    1
##   0 6048  268
##   1 2839  319
```

```
prop.table(table(matchedDat$NTC, matchedDat$ED_death, dnn = c("NTC","Mortality")),1)
```

```
##    Mortality
## NTC          0          1
##   0 0.95756808 0.04243192
##   1 0.89898670 0.10101330
```

```
#Data for evidence factor 2: TCI vs. TCII
matchedDat2 <- matchedDat[matchedDat$HOSP_TRAUMA %in% c(1,2),
                                  c("indexMatch","TCI","ED_death")]
matchedDatWide <- dcast(matchedDat2,
                        indexMatch ~ TCI , value.var = "ED_death")
tab <- table(matchedDatWide[,"1"], matchedDatWide[,"0"], dnn = c("Outcome TCI", "Outcome TCII"))
tab
```

```
##             Outcome TCII
## Outcome TCI    0    1
##           0 2907  117
```

```
##           1  117   17
```

Functions for sensitivity analysis.

```
#Function implementing Rosenbaum's primal sensitivity analysis for McNemar's test
primalSensAnalysisOneToOne <- function(data, exposure, outcome, indexMatch, Gamma) {

  tab2x2xS <- table(data[,exposure],
                    data[,outcome],
                    data[,indexMatch])

  nonExp_event <- apply(tab2x2xS, 3, FUN = "[", "0", "1")
  nonExp_nonEvent <- apply(tab2x2xS, 3, FUN = "[", "0", "0")
  exp_event <- apply(tab2x2xS, 3, FUN = "[", "1", "1")
  exp_nonEvent <- apply(tab2x2xS, 3, FUN = "[", "1", "0")

  numDiscordant <- sum((nonExp_event==1 & exp_nonEvent==1) |
                         (exp_event==1 & nonExp_nonEvent==1))

  mcNemarStat <- sum(exp_event==1 & nonExp_nonEvent==1)

  ps <- Gamma / (1 + Gamma)
  stat <- (mcNemarStat - ps*numDiscordant)/sqrt(numDiscordant*ps*(1-ps))
  pvalue <- 1 - pnorm(stat)

  return(list(stat = stat,
              pvalue = pvalue))
}

#Function implementing Rosenbaum's primal sensitivity analysis for MH test
primalSensAnalysisOneToK <- function(data, exposure, outcome, indexMatch, Gamma) {

  tab2x2xS <- table(data[,exposure],
                    data[,outcome],
                    data[,indexMatch])

  nonExp_event <- apply(tab2x2xS, 3, FUN = "[", "0", "1")
  nonExp_nonEvent <- apply(tab2x2xS, 3, FUN = "[", "0", "0")
  exp_event <- apply(tab2x2xS, 3, FUN = "[", "1", "1")
  exp_nonEvent <- apply(tab2x2xS, 3, FUN = "[", "1", "0")

  vect_m_s <- (nonExp_event + exp_event)
  vect_n_s <- (nonExp_event + nonExp_nonEvent + exp_event + exp_nonEvent)
  vect_p_s <- vect_m_s * Gamma / (Gamma * vect_m_s + vect_n_s - vect_m_s)

  mhStat <- sum(exp_event==1)
  stat <- (mhStat - sum(vect_p_s))/sqrt(sum(vect_p_s*(1-vect_p_s)))
  pvalue <- 1 - pnorm(stat)

  return(list(stat = stat,
         pvalue = pvalue))
}

#Function implementing Rosenbaum's sensitivity analysis on Fisher's joint test
sensAnalysisEvFactors <- function(data1, data2,
```

```
                                    exposure1, exposure2,
                                    outcome1, outcome2,
                                    indexMatch1, indexMatch2,
                                    Gamma1, Gamma2) {
  #Ev factor 1
  res1 <- primalSensAnalysisOneToK(data = data1,
                                exposure = exposure1,
                                outcome = outcome1,
                                indexMatch = indexMatch1,
                                Gamma = Gamma1)

  #Ev factor 2
  res2 <- primalSensAnalysisOneToOne(data = data2,
                                exposure = exposure2,
                                outcome = outcome2,
                                indexMatch = indexMatch2,
                                Gamma = Gamma2)
  #Joint test with Fisher's method:
  stat <- -2 * (log(res1$pvalue) + log(res2$pvalue) )
  stat

  pvalue <- 1 - pchisq(stat, df = 2 * 2)
  pvalue

  return(list(stat = stat,
         pvalue = pvalue))
}

#Make function accepting vectorial input
sensAnalysisEvFactorsVect <- Vectorize(FUN = sensAnalysisEvFactors,
                                       vectorize.args = c("Gamma1","Gamma2"))

#Fucntion to estimate the attributable effect
attributableEffect <- function(data, exposure, outcome, indexMatch, Gamma, alpha = 0.05) {

  tab2x2xS <- table(data[,exposure],
                    data[,outcome],
                    data[,indexMatch])

  nonExp_event <- apply(tab2x2xS, 3, FUN = "[", "0", "1")
  nonExp_nonEvent <- apply(tab2x2xS, 3, FUN = "[", "0", "0")
  exp_event <- apply(tab2x2xS, 3, FUN = "[", "1", "1")
  exp_nonEvent <- apply(tab2x2xS, 3, FUN = "[", "1", "0")

  S <- length(exp_event)

  #Assumed: only one treated subject per matched set
  if (!all((exp_event+exp_nonEvent) == 1)) stop("Each matched set must have ONE treated/exposed subject

  vect_m_s <- (nonExp_event + exp_event)
  vect_n_s <- (nonExp_event + nonExp_nonEvent + exp_event + exp_nonEvent)

  #Expectation of MH stat
```

```
  vect_lambdaBarBar_s <- vect_m_s * Gamma / (Gamma * vect_m_s + vect_n_s - vect_m_s)
  vect_lambdaBar_s <- (vect_m_s-1) * Gamma / (Gamma * (vect_m_s-1) + vect_n_s - (vect_m_s-1))
  #If m_s=0 (no events in the matched set), lambdaBar_s < 0 => force to NA
  vect_lambdaBar_s[vect_m_s == 0] <- NA
  #Variance of MH stat
  vect_omegaBarBar_s <- vect_lambdaBarBar_s * (1-vect_lambdaBarBar_s)
  vect_omegaBar_s <- vect_lambdaBar_s * (1-vect_lambdaBar_s)

  #In matched cohort studies, all the difference in the lambdas should be equal
  indexesOrder <- order((vect_lambdaBarBar_s - vect_lambdaBar_s),
                        (vect_omegaBarBar_s - vect_omegaBar_s), na.last = TRUE)

  tot_exp_event <- sum(exp_event==1)

  if(tot_exp_event==0) {

    #If tot_exp_event=0, there are no events among the exposed. The exposure cannot be the cause of any
    A_lowerCi <- NA
    pvalue <- NA

  } else {

    #a=0 (original test rejected or not)
    vect_p_s <- vect_lambdaBarBar_s

    stat <- (tot_exp_event - sum(vect_p_s))/sqrt(sum(vect_p_s*(1-vect_p_s)))
    pvalue <- 1 - pnorm(stat)

    if (pvalue < alpha) {

      for (a in 1:(tot_exp_event-1)) {

        vect_p_s <- c(vect_lambdaBar_s[indexesOrder][1:a], vect_lambdaBarBar_s[indexesOrder][(a+1):S])
        stat <- (tot_exp_event - a - sum(vect_p_s))/sqrt(sum(vect_p_s*(1-vect_p_s)))
        pvalue <- 1 - pnorm(stat)
        if(pvalue >= alpha) {break}
      }
      A_lowerCi <- a

    } else {

      A_lowerCi <- 0

    }

  }

  return(list(A_lowerCi = A_lowerCi,
              confLevel = 1-pvalue))
}
```

Evidence Factor Analysis

Two tests for two evidence factors:

1. Mantel-Haenszel statistic to compare NTC vs. TC in 1:2 matched sample.
2. McNemar's statistic to compare TCI vs. TCII in 1:1 matched sample.

We can use the Fisher's method to combine the p-values.

```
#MH test for evidence factor 1
test1 <- mantelhaen.test(table(matchedDat$NTC,
                               matchedDat$ED_death,
                               matchedDat$indexMatch))
test1
```

```
##
##  Mantel-Haenszel chi-squared test with continuity correction
##
## data:  table(matchedDat$NTC, matchedDat$ED_death, matchedDat$indexMatch)
## Mantel-Haenszel X-squared = 130.07, df = 1, p-value < 2.2e-16
## alternative hypothesis: true common odds ratio is not equal to 1
## 95 percent confidence interval:
##  2.246547 3.201424
## sample estimates:
## common odds ratio
##          2.681818
```

```
#McNemar's test for evidence factor 2
test2 <- mcnemar.test(tab)
test2
```

```
##
##  McNemar's Chi-squared test
##
## data:  tab
## McNemar's chi-squared = 0, df = 1, p-value = 1
```

```
#Fisher's method to combine p-values:
stat <- -2 * (log(test1$p.value) + log(test2$p.value) )
stat
```

```
## [1] 135.4039
```

```
pvalue <- 1 - pchisq(stat, df = 2 * 2)
pvalue
```

```
## [1] 0
```

These results are analogous to assuming $\Gamma_1 = \Gamma_2 = 1$ in the sensitivity analysis.

```
#Equivalent to MH test
resultMH <- primalSensAnalysisOneToK(data = matchedDat,
                           exposure = "NTC",
                           outcome = "ED_death",
                           indexMatch = "indexMatch",
                           Gamma = 1)
resultMH
```

```
## $stat
## [1] 11.45121
```

```
##
## $pvalue
## [1] 0
```

```
#Equivalent to McNemar's test
resultMN <- primalSensAnalysisOneToOne(data = matchedDat2,
                           exposure = "TCI",
                           outcome = "ED_death",
                           indexMatch = "indexMatch",
                           Gamma = 1)
resultMN
```

```
## $stat
## [1] 0
##
## $pvalue
## [1] 0.5
```

```
#Equivalent to Fisher's method for joint test
resultFisher <- sensAnalysisEvFactors(matchedDat, matchedDat2,
                       "NTC", "TCI",
                       "ED_death", "ED_death",
                       "indexMatch", "indexMatch",
                       Gamma1 = 1, Gamma2 = 1)
resultFisher
```

```
## $stat
## [1] Inf
##
## $pvalue
## [1] 0
```

```
#The value of the MH statistic is very large and the corresponding p-value is very small
#(almost zero). To compute log(p-value of MH test), which is required to compute the
#combined p-value with Fisher's method, we need to use the option log = T of pnorm
stat <- (- 2 * pnorm(resultMH$stat, lower.tail = F, log = T) - 2*log(resultMN$pvalue))
pvalue <- 1 - pchisq(stat, df = 2 * 2)

stat
```

```
## [1] 139.2456
```

```
pvalue
```

```
## [1] 0
```

Attributable effect:

```
attrEffMH <- attributableEffect(data = matchedDat,
                                exposure = "NTC",
                                outcome = "ED_death",
                                indexMatch = "indexMatch",
                                Gamma = 1, alpha = 0.05)
attrEffMH
```

```
## $A_lowerCi
## [1] 162
##
## $confLevel
```

```
## [1] 0.9465516
```

```
tab <- table(matchedDat$ED_death,matchedDat$NTC, dnn = c("ED mortality","NTC"))
tab
```

```
##              NTC
## ED mortality    0    1
##            0 6048 2839
##            1  268  319
```

```
#Risk in treatment groups
risks <- prop.table(tab, 2)["1",]
risks
```

```
##          0          1
## 0.04243192 0.10101330
```

```
#Risk difference
risks["1"]-risks["0"]
```

```
##          1
## 0.05858138
```

```
#Onesided 95% CI - Attributable risk
attrEffMH$A_lowerCi/length(unique(matchedDat$indexMatch))
```

```
## [1] 0.05129829
```

```
#The lower bound of the simultaneous two-sided CI for the RD was 0.041.
#What would be the corresponding lower bound with this method?
#alpha = 0.025 for the two-sided and /3 for Bonferroni correction in the 3 CIs
attrForComparison <- attributableEffect(data = matchedDat,
                                exposure = "NTC",
                                outcome = "ED_death",
                                indexMatch = "indexMatch",
                                Gamma = 1, alpha = 0.025/3)
attrForComparison$A_lowerCi/length(unique(matchedDat$indexMatch))
```

```
## [1] 0.04781507
```

Sensitivity Analysis

If treatment assignment cannot be assumed to be random in either of the steps, we can evaluate the robustness of the result with the sensitivity analyses. We can implement SA for the first step (with SA parameter Γ_1), for the second step (with SA parameter Γ_2) or for the combined p-value.

We can construct a table to show values of Γ_1 and Γ_2 corresponding to non-significant p-values.

```
#Table
vectorGamma1 <- c(1, 1.5, 2, 2.1, 2.2, 2.3, 2.4, 2.5, 20)
vectorGamma2 <- c(1, 1.25, 1.5, 20)

matrixResult <- matrix(NA,
                       nrow = length(vectorGamma1),
                       ncol = length(vectorGamma2))
rownames(matrixResult) <- paste0("G1=",vectorGamma1)
colnames(matrixResult) <- paste0("G2=",vectorGamma2)

for( i_G1 in 1:length(vectorGamma1)) {
  for( i_G2 in 1:length(vectorGamma2)) {
     resTemp <- sensAnalysisEvFactors(matchedDat, matchedDat2,
                                                "NTC", "TCI",
                                                "ED_death", "ED_death",
                                                "indexMatch", "indexMatch",
                                                Gamma1 = vectorGamma1[i_G1], Gamma2 = vectorGamma2[i_G2])$
    matrixResult[i_G1,i_G2] <- sprintf("%.3f",resTemp)
  }
}

matrixResult
```

```
##        G2=1    G2=1.25 G2=1.5  G2=20
## G1=1   "0.000" "0.000" "0.000" "0.000"
## G1=1.5 "0.000" "0.000" "0.000" "0.000"
## G1=2   "0.002" "0.003" "0.003" "0.003"
## G1=2.1 "0.009" "0.016" "0.016" "0.017"
## G1=2.2 "0.033" "0.056" "0.058" "0.058"
## G1=2.3 "0.087" "0.146" "0.151" "0.151"
## G1=2.4 "0.183" "0.295" "0.305" "0.305"
## G1=2.5 "0.314" "0.485" "0.498" "0.498"
## G1=20  "0.847" "0.999" "1.000" "1.000"
```

Plot to show threshold in significance at $\alpha = 0.05$ level in Γ_1 and Γ_2 space.

```
#Look for limits of plots:

#Smallest Gamma1 corresponding to alpha=0.05 at Gamma2 = Inf
sensAnalysisEvFactors(matchedDat, matchedDat2,
                      "NTC", "TCI",
                      "ED_death", "ED_death",
                      "indexMatch", "indexMatch",
                      Gamma1 = 2.1867434, Gamma2 = 25)
```

```
## $stat
## [1] 9.487728
##
## $pvalue
## [1] 0.05000002
```

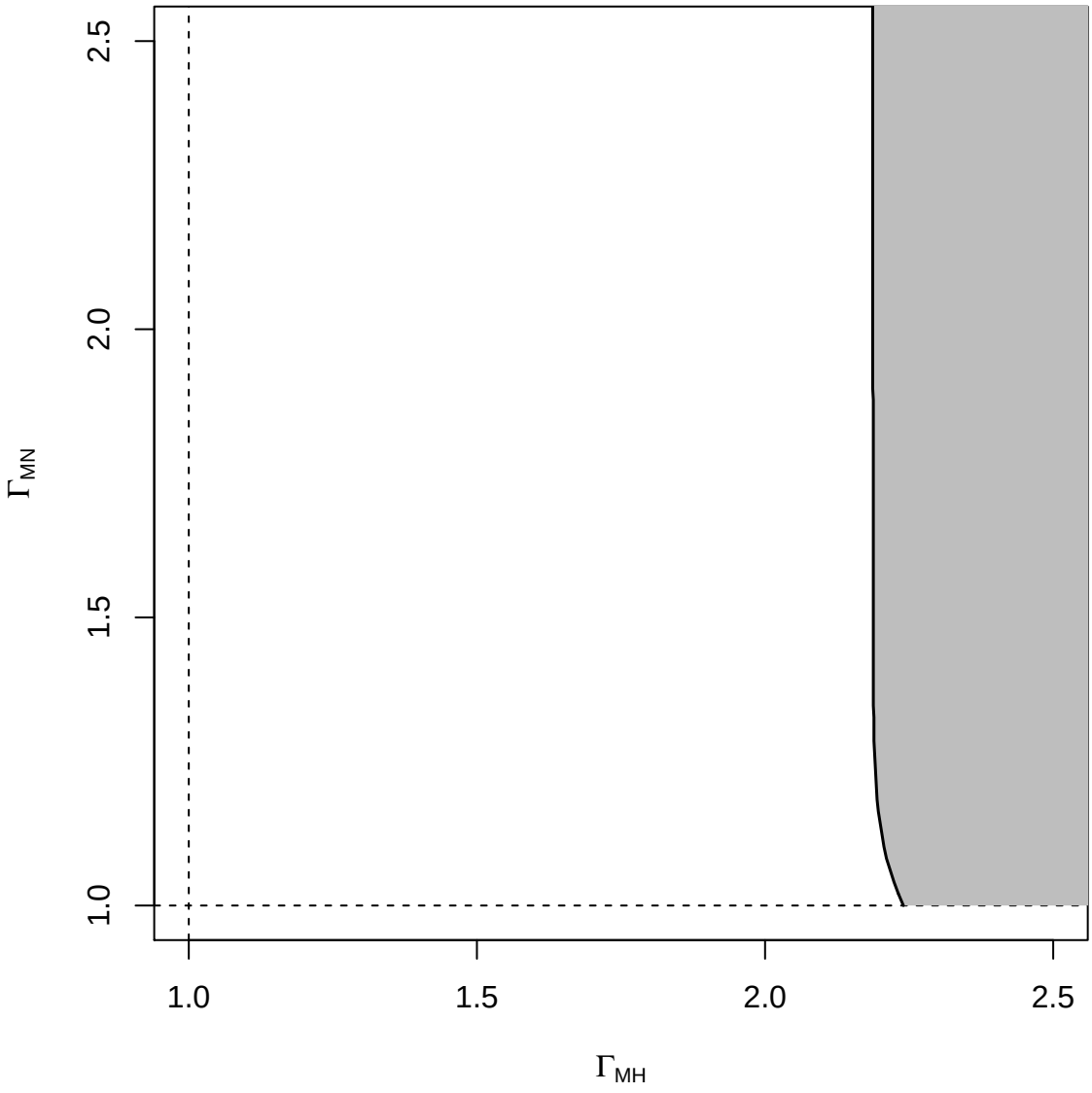
```
#Largest Gamma1 corresponding to alpha=0.05 at Gamma2 = 1
sensAnalysisEvFactors(matchedDat, matchedDat2,
                      "NTC", "TCI",
                      "ED_death", "ED_death",
```

```
                      "indexMatch", "indexMatch",
                      Gamma1 = 2.24002, Gamma2 = 1)
```

```
## $stat
## [1] 9.488023
##
## $pvalue
## [1] 0.04999393
```

```
vectorGamma1_1 <- seq(2.1867434, 2.24002, length = 50)
vectorGamma2_1 <- seq(1, 2, length = 50)
grid_1 <- expand.grid(Gamma1 = vectorGamma1_1,
                      Gamma2 = vectorGamma2_1)

grid <- rbind(grid_1)

grid$pvalue <- NA
grid$threshold <- 0

currentGamma2 <- grid$Gamma2[1]
thresholdFound <- FALSE

for( i in 1:nrow(grid)) {

  if(grid$Gamma2[i]!=currentGamma2) {
    currentGamma2 <- grid$Gamma2[i]
    thresholdFound <- FALSE
  }

  if(thresholdFound == T) { next; }

  resTemp <- sensAnalysisEvFactors(matchedDat, matchedDat2,
                        "NTC", "TCI",
                        "ED_death", "ED_death",
                        "indexMatch", "indexMatch",
                        Gamma1 = grid$Gamma1[i], Gamma2 = grid$Gamma2[i])
  grid$pvalue[i] <- resTemp$pvalue

  if(resTemp$pvalue>=.05 & thresholdFound==F ) {
     grid$threshold[i] <- 1
     thresholdFound <- T
  }
}


par(mar=c(4,4,1,1))
plot(NA,NA,
      xlim = c(1,2.5), ylim = c(1,2.5),
      xlab = expression(Gamma[MH]) ,
      ylab = expression(Gamma[MN]),
      main = "",
      lwd = 1.5, type = "l", asp = 1)

xMin <- min(grid$Gamma1[grid$threshold==1])
```

```
xMax <- max(grid$Gamma1[grid$threshold==1])
yMax <- max(grid$Gamma2[grid$threshold==1])
yMin <- min(grid$Gamma2[grid$threshold==1])
abline(h=1, v=1, lty = 2, col = "black")

polygon(c(2.24002,xMax,grid$Gamma1[grid$threshold==1],xMin,xMin,3,3),
        c(1,yMin,      grid$Gamma2[grid$threshold==1],yMax,3   ,3,1), col = "gray", border = NA)
lines(grid$Gamma1[grid$threshold==1],
      grid$Gamma2[grid$threshold==1], lwd = 1.5)
lines(c(xMin,xMin), c(yMax,3), lwd = 1.5)
lines(c(xMax,2.24002), c(yMin,1), lwd = 1.5)
```



Computation of Sensitivity Value With Our Method

Computation of sensitivity value of first evidence factor using our approach for MH test.

```r
#Fit conditional logistic regression model, with treatment as only
# independent variable and ED mortality as outcome
model <- clogit(ED_death ~ NTC + strata(indexMatch),
                data = matchedDat, method = "exact")
summary(model)
```

```
## Call:
## coxph(formula = Surv(rep(1, 9474L), ED_death) ~ NTC + strata(indexMatch),
##     data = matchedDat, method = "exact")
##
##   n= 9474, number of events= 587
##
##        coef exp(coef) se(coef)     z Pr(>|z|)
## NTC 0.99711   2.71043  0.09045 11.02   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##     exp(coef) exp(-coef) lower .95 upper .95
## NTC      2.71     0.3689      2.27     3.236
##
## Rsquare= 0.013   (max possible= 0.114 )
## Likelihood ratio test= 124  on 1 df,   p=0
## Wald test            = 121.5  on 1 df,   p=0
## Score (logrank) test = 131.1  on 1 df,   p=0
```

```r
#One-sided 95%-level CI of odds ratio, Wald method
ciWald <- exp(confint(model,level = .90))
orL_Wald <- ciWald[1]
orL_Wald
```

```
## [1] 2.33574
```

```r
#Verify: causal inference is robust to unobserved confounders characterized by
# Gamma_MH up to the value of the lower bound of the CI.
primalSensAnalysisOneToK(data = matchedDat,
                         exposure = "NTC",
                         outcome = "ED_death",
                         indexMatch = "indexMatch",
                         Gamma = 2)
```

```
## $stat
## [1] 3.372789
##
## $pvalue
## [1] 0.0003720553
```

```r
primalSensAnalysisOneToK(data = matchedDat,
                         exposure = "NTC",
                         outcome = "ED_death",
                         indexMatch = "indexMatch",
                         Gamma = 2.33)
```

```
## $stat
## [1] 1.673575
##
## $pvalue
```

```
## [1] 0.04710711
```

```
primalSensAnalysisOneToK(data = matchedDat,
                         exposure = "NTC",
                         outcome = "ED_death",
                         indexMatch = "indexMatch",
                         Gamma = 2.34)
```

```
## $stat
## [1] 1.626101
##
## $pvalue
## [1] 0.05196404
```

```
primalSensAnalysisOneToK(data = matchedDat,
                         exposure = "NTC",
                         outcome = "ED_death",
                         indexMatch = "indexMatch",
                         Gamma = 3)
```

```
## $stat
## [1] -1.122673
##
## $pvalue
## [1] 0.8692119
```