idem: Inference in Randomized Controlled Trials with Death and Missingness

Chenguang Wang

2016-07-04

Introduction

In randomized studies involving severely ill patients, functional outcomes are often unobserved due to missed clinic visits, premature withdrawal or death. It is well known that if these unobserved functional outcomes are not handled properly, biased treatment comparisons can be produced.

R package **idem** implement a procedure for comparing treatments that is based on the composite endpoint of both the functional outcome and survival. The procedure considers missing data imputation with a sensitivity analysis strategy to handle the unobserved functional outcomes not due to death.

Data accepted by idem

In dataset accepted by **idem**, each row should represent a subject with treatment assignment, baseline coveraites, baseline outcome, post-randomization outcomes and survival time.

The **idem** package provides dataset *abc* from *ABC* trial as an example data set.

```
head(abc);
```

```
##      AGE TRT SURV Y2 Y1
## 1 59.63   1  999 NA NA
## 2 66.89   0  999 52 49
## 3 59.70   1    1 NA NA
## 4 81.41   0   72 NA NA
## 5 66.52   1  999 45 51
## 6 40.27   0   65 NA NA
```

Basic setting

Parameters of most functions in **idem** are organized and passed by a list. These parameters include variable names in the dataset, endpoint specification, duration of the study, etc..

The function **chkPara** checks the specification in the list of parameters for errors:

```
err.lst.var <- list(trt="TRT", outcome=c("Y1","Y2"),
                y0=NULL, endfml="Y2", bounds=c(10,20),
                duration=365);

chkPara(err.lst.var, abc);
```

```
## Model specification in not valid. Please check the following:
## ----No survival time specified
## ----Endpoint does not involve outcome
## ----Upper bound is bigger than some observed outcomes
```

```
lst.var <- list(trt="TRT", surv="SURV", outcome=c("Y1","Y2"),
                y0=NULL, endp=c("Y2"),
                unitTime="days",
                trt.label = c("UC+SBT", "SAT+SBT"),
                cov=c("AGE"), endfml="Y2",
                duration=365, bounds=c(0,100));

chkPara(lst.var, abc);
```

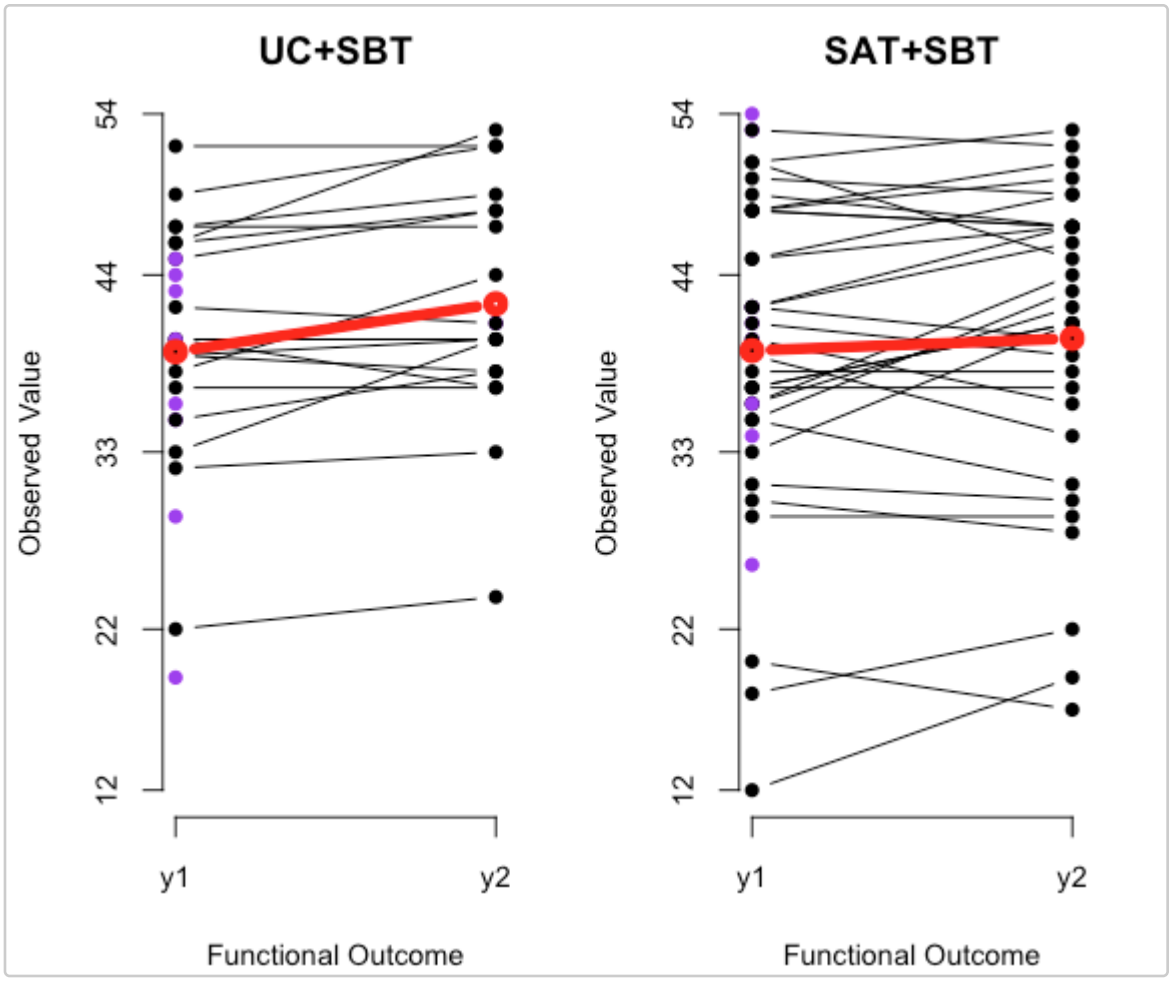
```
## No error found.
```

Data visualization

idem provides several functions for the visualization of the data.

Spaghetti plot for completers

```
plotCompleters(abc, lst.var);
```



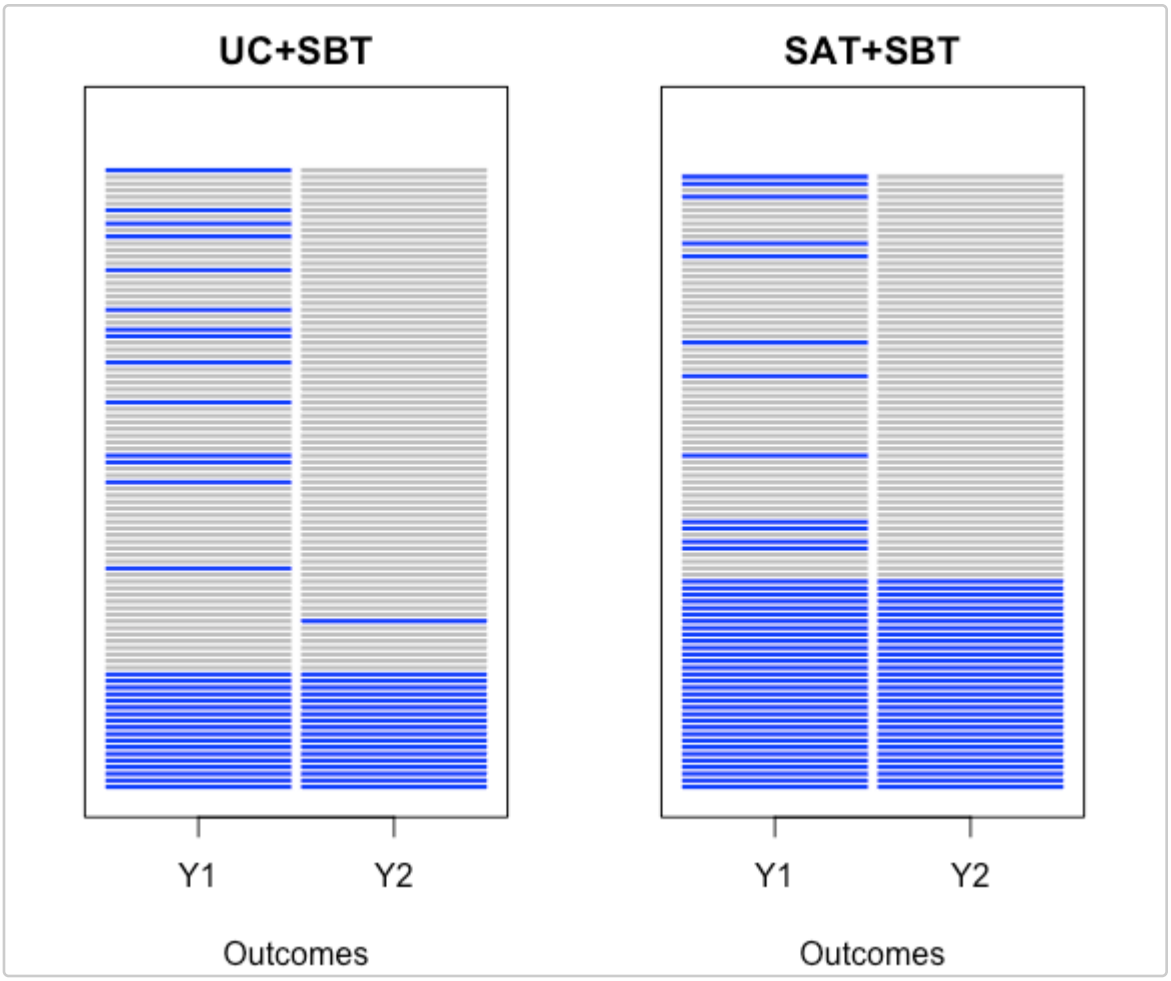
Missing pattern frequency table

```
get.mis.table(abc, lst.var);
```

```
##                 Y1         Y2         Control    Intervention
## Deaths on study ""         ""         "58 (62%)" "38 (41%)"
## S=1             "Observed" "Observed" "18 (19%)" "32 (34%)"
## S=2             "Observed" "Missing"  "8 (9%)"   "8 (9%)"
## S=3             "Missing"  "Observed" "1 (1%)"   "0 (0%)"
## S=4             "Missing"  "Missing"  "9 (10%)"  "15 (16%)"
## Total           ""         ""         "94"       "93"
```

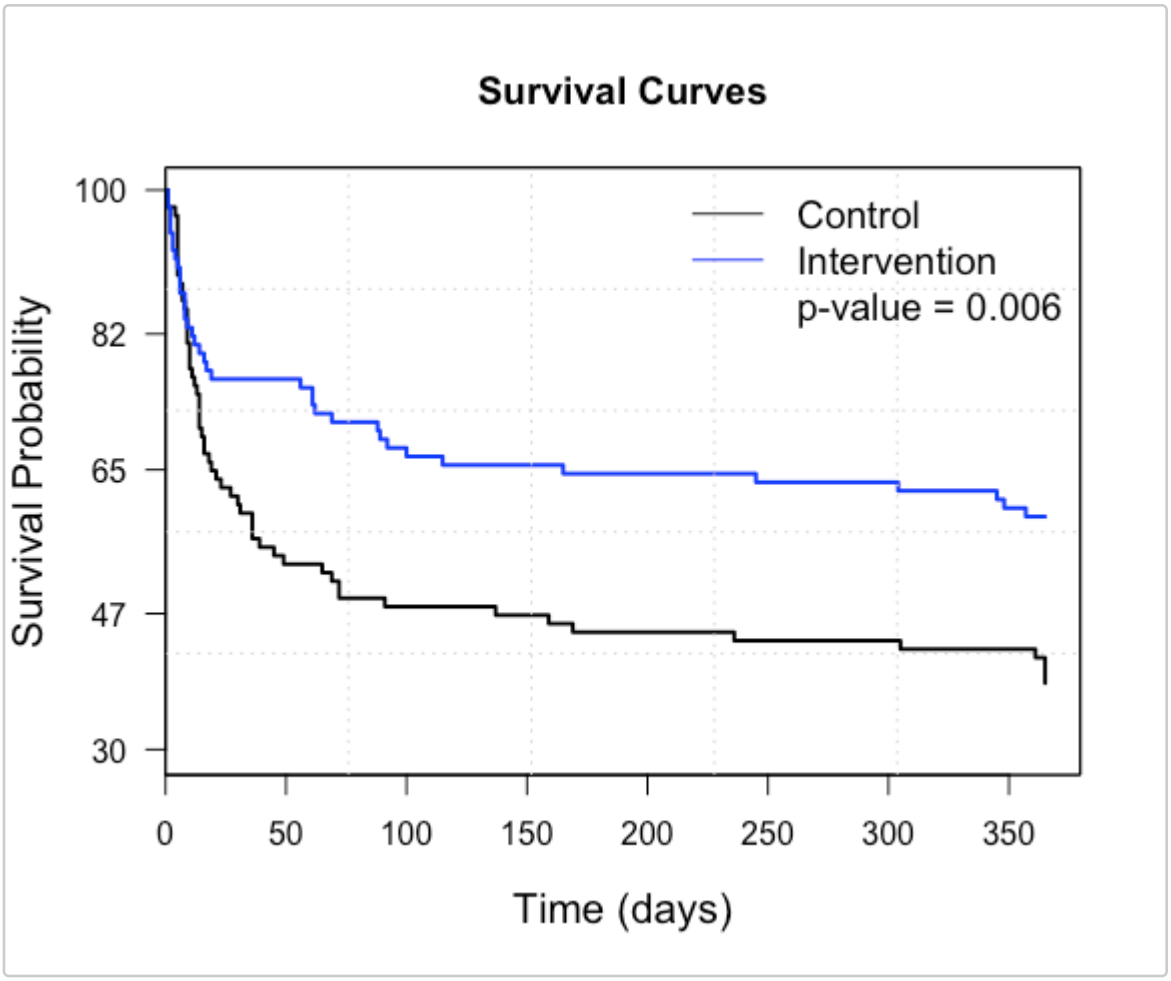
Missing pattern heatmap

```
plotMisPattern(abc, lst.var);
```



Kaplan-Meier curves

```
plotSurv(abc, lst.var);
```



Missing data imputation

Model fitting

The first step in missing data imputation is to fit the imputation model to data observed from the completers, i.e. the subjects who were alive at the end of the study without missing data. The result has class name **IDEM.FIT**, which will be passed to imputation functions.

```
rst.fit <- fit.model(abc, lst.var);
```

Check convergence

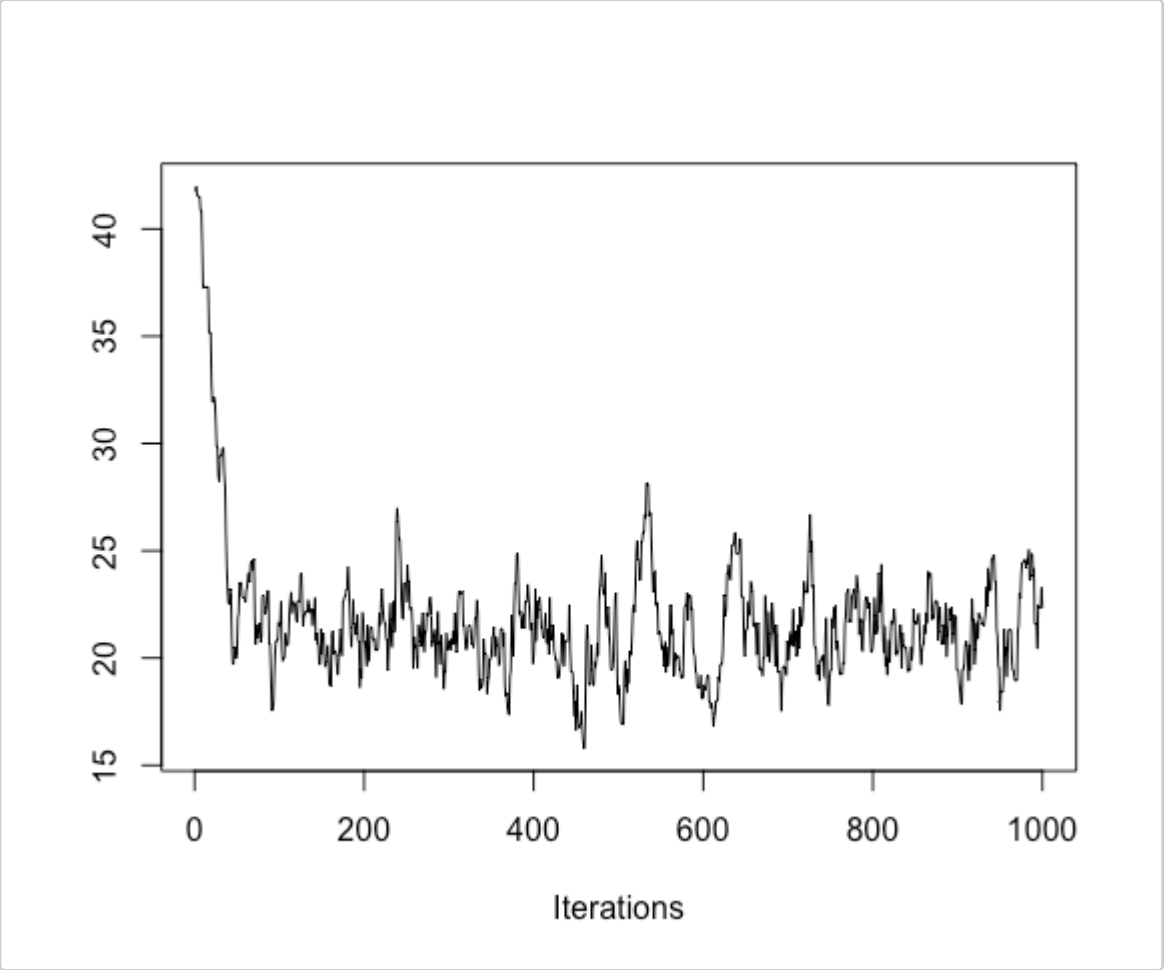
The **p.scale** factor adjusts the mixing of the MCMC sampling for the imputation of missing values. It is suggested that the convergence of the MCMC chains for at least a small subset of the subjects should be checked first. This can be done by the **get.imp.all** function by setting **trace.n** to be a non-zero number.

```
rst.mixing <- get.imp.all(abc, rst.fit, lst.var, deltas=0,
```

```
                        trace.n=1, normal=TRUE, iter=1000,
                        p.scale=10);
```

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

```
rst.chain <- list();
for (i in 1:ncol(rst.mixing$mcmc[[1]])) {
    rst.chain[[i]] <- rst.mixing$mcmc[[1]][,i,drop=FALSE];
}
coda::traceplot(rst.chain);
```



Imputation

The following code shows how to use **get.imp.all** to get the imputed complete datasets under benchmark assmption **delta=0** and for sensitivity analysis. We use **300** iterations to reduce the computation time.

```
rst.imp <- get.imp.all(abc, rst.fit, lst.var, deltas=c(-0.25,0,0.25),
                       normal=TRUE, iter=300, n.imp=2, thin=10,
```

```
                        p.scale=10);
```

The following results show the completed datasets:

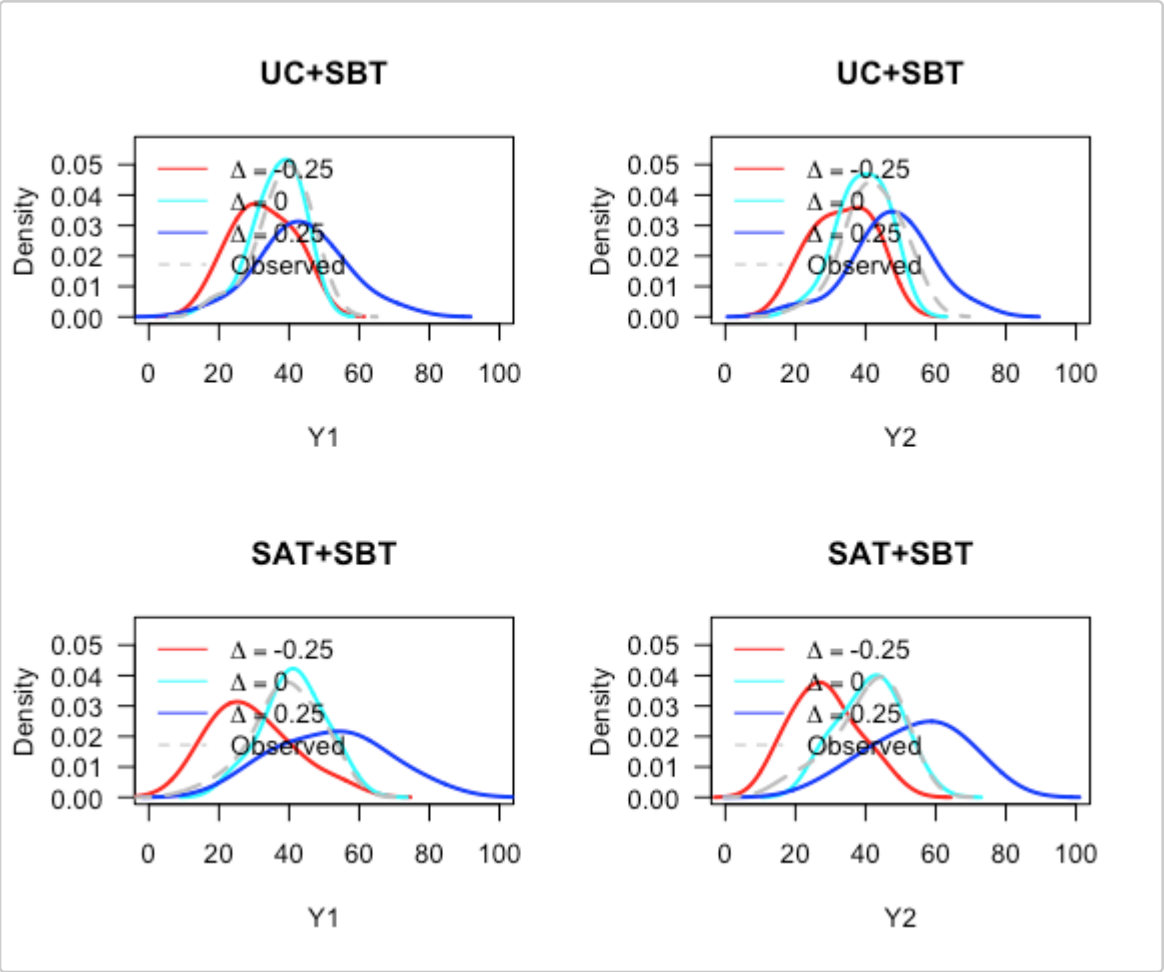
```
tail(rst.imp, n=10);
```

```
##        ID DELTA IMP   AGE TRT SURV       Y2 Y1 ORGY1 ORGY2     ENDP
## 1852 186  0.00   1 67.06   0  999 41.56661 36    36    NA 41.56661
## 1853 186  0.00   2 67.06   0  999 32.64299 36    36    NA 32.64299
## 1854 186  0.25   1 67.06   0  999 39.48108 36    36    NA 39.48108
## 1855 186  0.25   2 67.06   0  999 38.66370 36    36    NA 38.66370
## 187  187 -0.25   1 66.12   1  999 24.90369 26    26    NA 24.90369
## 1871 187 -0.25   2 66.12   1  999 25.19095 26    26    NA 25.19095
## 1872 187  0.00   1 66.12   1  999 29.77881 26    26    NA 29.77881
## 1873 187  0.00   2 66.12   1  999 29.12097 26    26    NA 29.12097
## 1874 187  0.25   1 66.12   1  999 31.90955 26    26    NA 31.90955
## 1875 187  0.25   2 66.12   1  999 24.47117 26    26    NA 24.47117
```

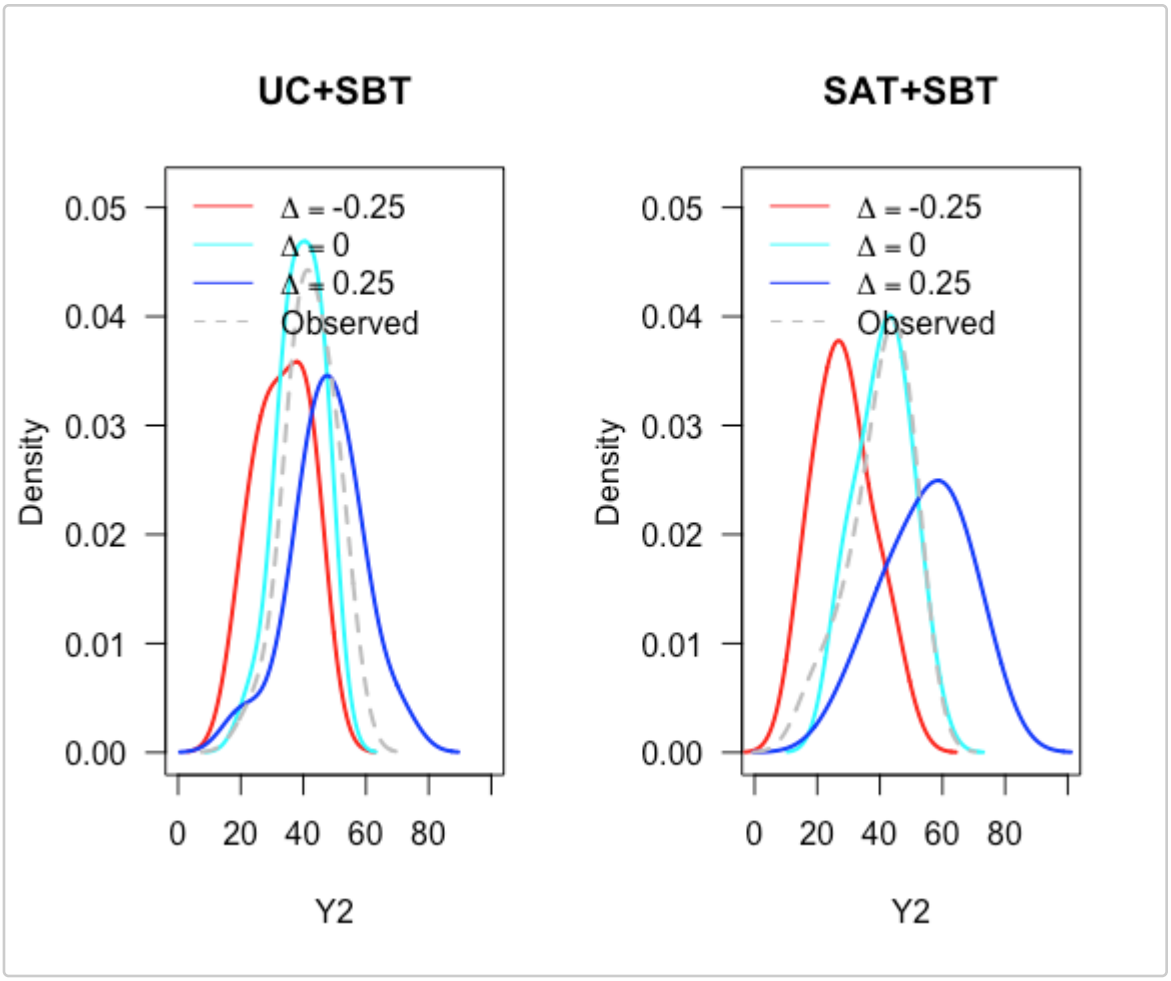
Plot denisity of the imputed data

Function **plotImputed** plots the densities of the imputed outcomes and the imputed functional endpoint.

```
plotImputed(rst.imp, lst.var, deltas=c(-0.25,0,0.25), xlim=c(0,100), endp=FALSE);
```



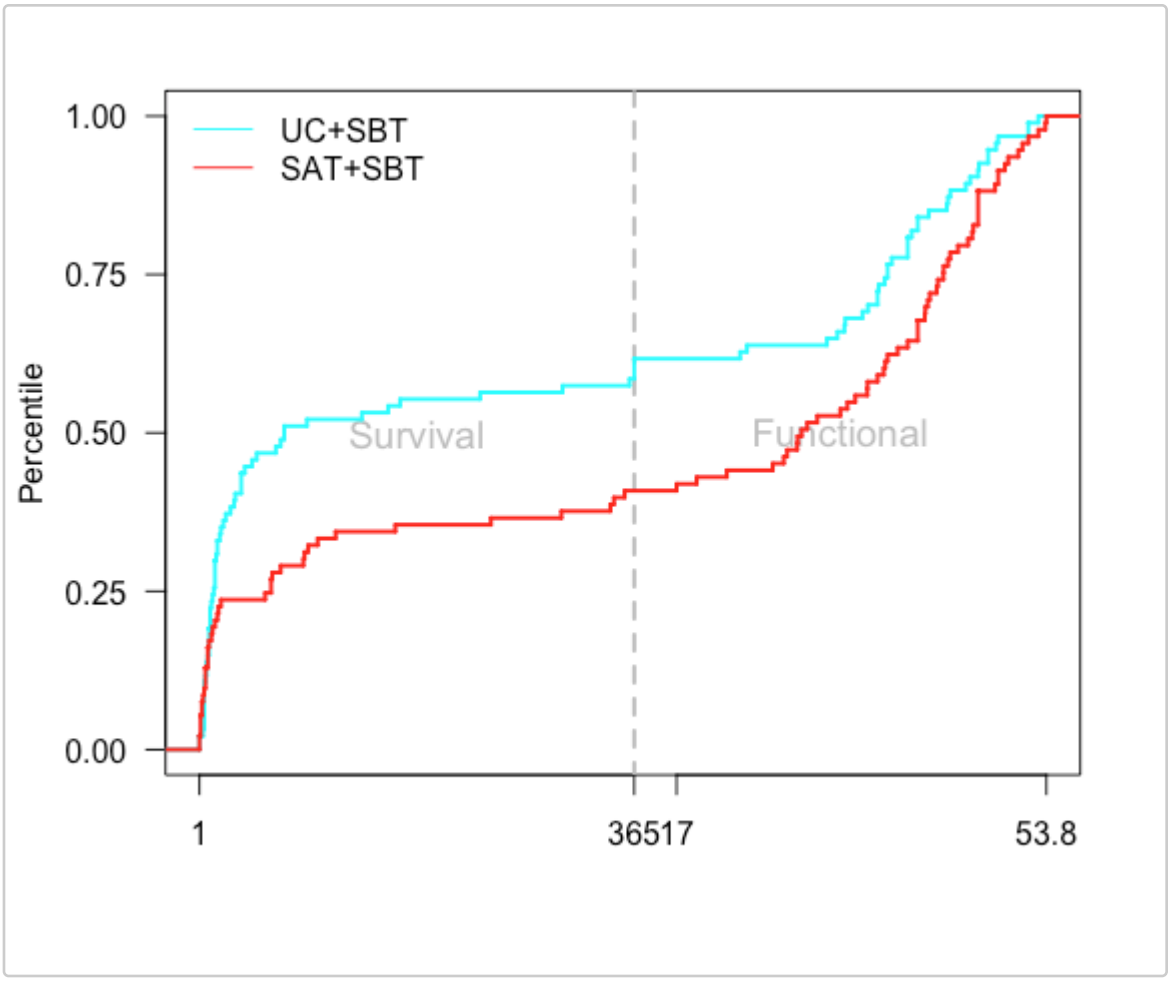

```
plotImputed(rst.imp, lst.var, deltas=c(-0.25,0,0.25), xlim=c(0,100), endp=TRUE);
```



Plot the cumulative distribution of the compositve endpoint

Function **plotComposite** generates the treatment-specific cumulative distribution functions of the composite endpoint, where the values of the composite endpoint are labeled according to the survival time and functional endpoint among survivors.

```
plotComposite(rst.imp, lst.var, delta=0);
```



Estimation of θ and quantiles

Given a complete dataset with imputed outcomes, **idem** estimates treatment effect and quantiles of the composite endpoint using function **get.theta.quantiles**:

```
rst.est <- get.theta.quantiles(rst.imp, lst.var, quantiles=c(0.25,0.5,0.75));
print(rst.est$theta);
```

```
##   Delta0 Delta1       Theta
## 1  -0.25  -0.25 -0.16815374
## 2   0.00  -0.25 -0.13532372
## 3   0.25  -0.25 -0.09460078
## 4  -0.25   0.00 -0.23713109
## 5   0.00   0.00 -0.20327156
## 6   0.25   0.00 -0.14607641
## 7  -0.25   0.25 -0.28506063
## 8   0.00   0.25 -0.25966598
## 9   0.25   0.25 -0.21150766
```

```
print(rst.est$quantiles);
```

```
##    Delta TRT    Q    Quant
## 1  -0.25   0 0.25 -1.00000
## 2  -0.25   0 0.50 -1.00000
## 3  -0.25   0 0.75 37.00000
## 4  -0.25   1 0.25 -1.00000
## 5  -0.25   1 0.50 22.01691
## 6  -0.25   1 0.75 39.89199
## 7   0.00   0 0.25 -1.00000
## 8   0.00   0 0.50 -1.00000
## 9   0.00   0 0.75 37.88017
## 10  0.00   1 0.25 -1.00000
## 11  0.00   1 0.50 29.56049
## 12  0.00   1 0.75 43.86520
## 13  0.25   0 0.25 -1.00000
## 14  0.25   0 0.50 -1.00000
## 15  0.25   0 0.75 40.50000
## 16  0.25   1 0.25 -1.00000
## 17  0.25   1 0.50 31.57008
## 18  0.25   1 0.75 47.00000
```

Bootstrap analysis

idem evaluates the uncertainty of the estimated θ and quantiles by bootstrap analysis.

For illustration, we run **10** bootstrap samples by the following code:

```
rst.boot <- get.bs.all(n.boot = 10,
                       n.cores = 5,
                       data.all = abc,
                       lst.var = lst.var,
                       deltas = c(-0.25, 0, 0.25),
                       quantiles = c(0.25,0.5,0.75),
                       normal=TRUE, iter=300,
                       n.imp=2, thin=10,
                       p.scale=10);
```

Hypothesis testing

Hypothesis testing results and the confidence intervals of and quantiles of the composite endpoint are obtained by summarizing the results from the bootstrap analysis and the analysis on the original dataset:

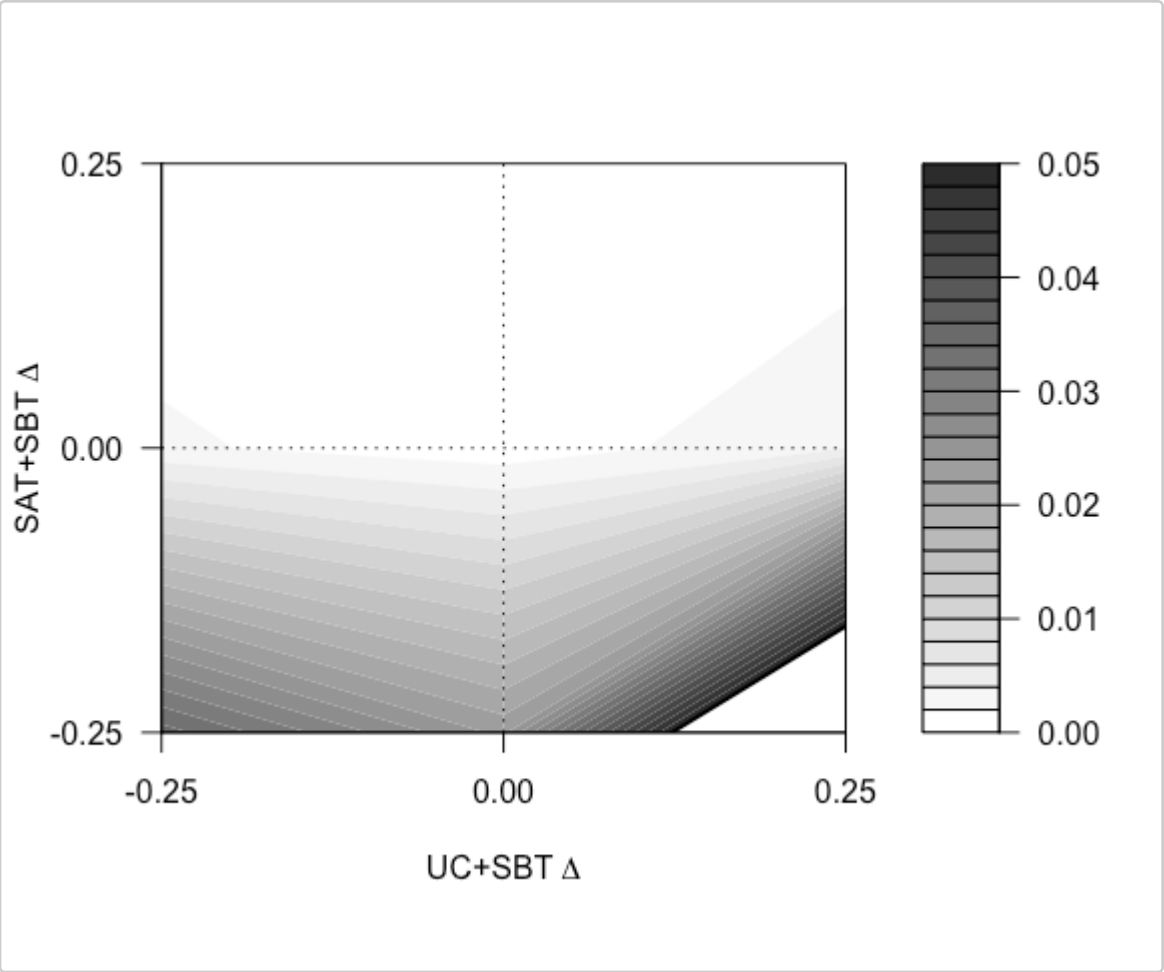
```
rst.final <- get.overall.rst(rst.est, rst.boot);
```

```
print(rst.final);
```

```
## $theta
##   Delta0 Delta1       Theta         SD       PValue
## 1  -0.25  -0.25 -0.16815374 0.07973151 3.494460e-02
## 2   0.00  -0.25 -0.13532372 0.05970093 2.340893e-02
## 3   0.25  -0.25 -0.09460078 0.05357365 7.742805e-02
## 4  -0.25   0.00 -0.23713109 0.07796869 2.355096e-03
## 5   0.00   0.00 -0.20327156 0.05951428 6.366296e-04
## 6   0.25   0.00 -0.14607641 0.05060692 3.895638e-03
## 7  -0.25   0.25 -0.28506063 0.07748273 2.341306e-04
## 8   0.00   0.25 -0.25966598 0.06251603 3.272986e-05
## 9   0.25   0.25 -0.21150766 0.05516525 1.260381e-04
##
## $quantiles
##    Delta TRT    Q    Quant         LB           UB
## 1  -0.25   0 0.25 -1.00000 -0.3090797 -0.072823725
## 2  -0.25   0 0.50 -1.00000 -0.2183082 -0.040265385
## 3  -0.25   0 0.75 37.00000 -0.1567862 -0.002144818
## 4  -0.25   1 0.25 -1.00000 -0.3950355 -0.158856669
## 5  -0.25   1 0.50 22.01691 -0.3032487 -0.117221460
## 6  -0.25   1 0.75 39.89199 -0.2183596 -0.070046900
## 7   0.00   0 0.25 -1.00000 -0.4376201 -0.207232327
## 8   0.00   0 0.50 -1.00000 -0.3711622 -0.189044269
## 9   0.00   0 0.75 37.88017 -0.3077070 -0.153085678
## 10  0.00   1 0.25 -1.00000 -0.3090797 -0.072823725
## 11  0.00   1 0.50 29.56049 -0.2183082 -0.040265385
## 12  0.00   1 0.75 43.86520 -0.1567862 -0.002144818
## 13  0.25   0 0.25 -1.00000 -0.3950355 -0.158856669
## 14  0.25   0 0.50 -1.00000 -0.3032487 -0.117221460
## 15  0.25   0 0.75 40.50000 -0.2183596 -0.070046900
## 16  0.25   1 0.25 -1.00000 -0.4376201 -0.207232327
## 17  0.25   1 0.50 31.57008 -0.3711622 -0.189044269
## 18  0.25   1 0.75 47.00000 -0.3077070 -0.153085678
##
## attr(,"class")
## [1] "IDEM.TEST"
```

A contour plot of p-values in the sensitivity analysis results can be generated by **plotContour**:

```
plotContour(rst.final, lst.var, nlevels = 30,
            con.v=0.05, zlim=c(0, 0.05), main="");
```




```
##             [,1]         [,2]         [,3]
## [1,] 0.03494460 0.0023550963 2.341306e-04
## [2,] 0.02340893 0.0006366296 3.272986e-05
## [3,] 0.07742805 0.0038956375 1.260381e-04
```