Numeric Example

Import precomputed data

Set the data's directory preferably in the variable 'filename'.

```
filename <- "import/precomp_testdata.dat"
data <- importNew(filename)
```

```
# Create inequality members
ineqmembers <- CineqmembersNew(data$mate)
```

```
#Create Data Array
dataArray <- CdataArrayNew(data$distanceMatrices, ineqmembers)
```

Maximization

Differential Evolution Method

The default DifferentialEvolution parameters:

```
# Objective function
coefficient1 <- 1
numFreeAttrs <- data$noAttr - 1
bounds <- makeBounds(data$noAttr, 100)
optimParams <- list(lower=bounds$lower, upper=bounds$upper, NP=50, F=0.6, CR=0.5,
                    itermax=100, trace=FALSE, reltol=1e-3)
permuteInvariant <- TRUE
# Set seed once before the rest.
randomSeed <- 42
set.seed(randomSeed)
optimizeScoreArgs <- list(dataArray = dataArray,
                          coefficient1 = coefficient1,
                          optimParams = optimParams,
                          getIneqSat = TRUE,
                          permuteInvariant = permuteInvariant)
optResult <- do.call(optimizeScoreFunction, optimizeScoreArgs)
print(optResult)
```

option name	default value	
lower,upper	-10,10	two vectors specifying scalar real lower and upper bounds on each parameter to be optimiz
CR	0.5	crossover probability from interval [0,1]
trace	FALSE	Positive integer or logical value indicating whether printing of progress occurs at each itera
itermax	100	the maximum iteration (population generation) allowed
F	0.6	differential weighting factor from interval [0,2]
NP	50	number of population members. Defaults to NA; if the user does not change the value of N
reltol	0.001	relative convergence tolerance. The algorithm stops if it is unable to reduce the value by a
RandomSeed	0	Random Seed to be used for result reproducibility

```
## $optVal
## [1] 96
##
## $optArg
##      par4      par1      par2      par3
##   1.81745  -2.75489  87.03322 -17.54139
##
## $ineqSat
##   [1] 1 1 1 1 1 1 1 1 0 0 1 1 0 1 1 1 1 1 1 1 0 1 1 0 0 1 1 1 0 0 1 1 1 1 1 0 0
##  [38] 1 1 1 1 0 1 1 1 1 0 0 1 1 1 0 0 1 1 1 0 0 1 0 0 1 1 1 1 1 0 1 0 0 1 1 0 0
##  [75] 0 1 1 0 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 0 1 1 0
## [112] 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 1
```

```
print(calcPerMarketStats(optResult$ineqSat, makeGroupIDs(ineqmembers)))
```

```
##      Market ID Total inequalities Satisfied inequalities Satisfied/Total ratio
## [1,]         1                 45                     34             0.7555556
## [2,]         2                 45                     28             0.6222222
## [3,]         3                 45                     34             0.7555556
```

Confidence Intervals

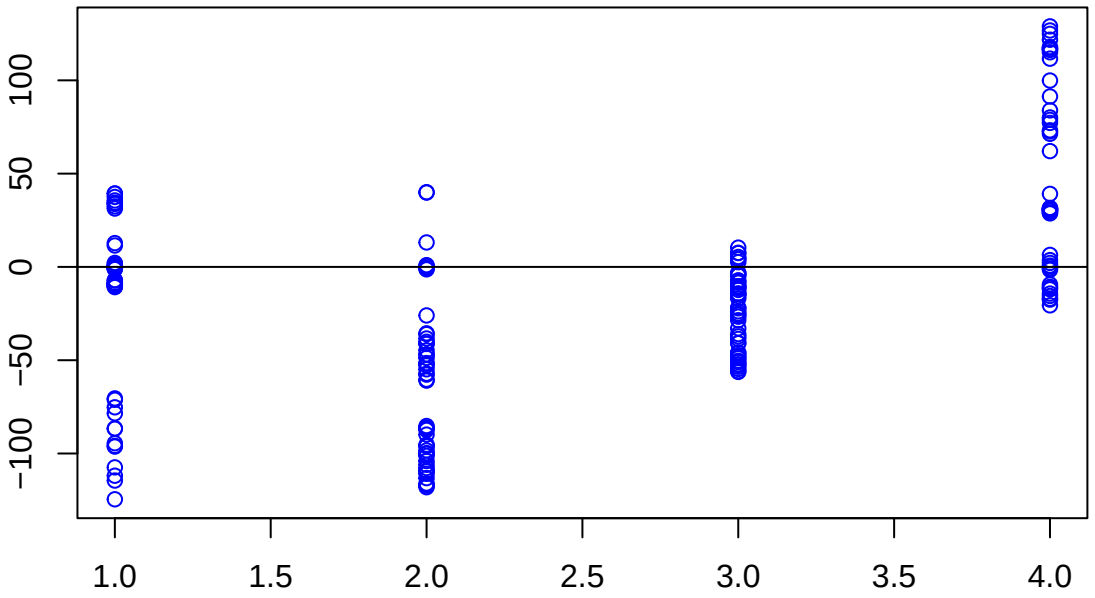
#Generate random subsample

```
#Create groupIDs
groupIDs <- makeGroupIDs(ineqmembers)
ssSize <- 2
optionsCR <- list(progressUpdate=1, confidenceLevel=0.95, asymptotics="nests")
numSubsamples <- 50
pointEstimate <- as.numeric(optResult$optArg)
optimizeScoreArgs$dataArray <- NULL
optimizeScoreArgs$getIneqSat <- FALSE
cr <- pointIdentifiedCRNew(ssSize, numSubsamples, pointEstimate, numFreeAttrs,
                           groupIDs, dataArray,
                           optimizeScoreArgs = optimizeScoreArgs,
                           options = optionsCR)
```

```
## [pointIdentifiedCRNew] Iterations completed: 1
## [pointIdentifiedCRNew] Iterations completed: 2
## [pointIdentifiedCRNew] Iterations completed: 3
## [pointIdentifiedCRNew] Iterations completed: 4
## [pointIdentifiedCRNew] Iterations completed: 5
## [pointIdentifiedCRNew] Iterations completed: 6
## [pointIdentifiedCRNew] Iterations completed: 7
## [pointIdentifiedCRNew] Iterations completed: 8
## [pointIdentifiedCRNew] Iterations completed: 9
## [pointIdentifiedCRNew] Iterations completed: 10
## [pointIdentifiedCRNew] Iterations completed: 11
## [pointIdentifiedCRNew] Iterations completed: 12
## [pointIdentifiedCRNew] Iterations completed: 13
## [pointIdentifiedCRNew] Iterations completed: 14
## [pointIdentifiedCRNew] Iterations completed: 15
## [pointIdentifiedCRNew] Iterations completed: 16
## [pointIdentifiedCRNew] Iterations completed: 17
## [pointIdentifiedCRNew] Iterations completed: 18
```

```
## [pointIdentifiedCRNew] Iterations completed: 19
## [pointIdentifiedCRNew] Iterations completed: 20
## [pointIdentifiedCRNew] Iterations completed: 21
## [pointIdentifiedCRNew] Iterations completed: 22
## [pointIdentifiedCRNew] Iterations completed: 23
## [pointIdentifiedCRNew] Iterations completed: 24
## [pointIdentifiedCRNew] Iterations completed: 25
## [pointIdentifiedCRNew] Iterations completed: 26
## [pointIdentifiedCRNew] Iterations completed: 27
## [pointIdentifiedCRNew] Iterations completed: 28
## [pointIdentifiedCRNew] Iterations completed: 29
## [pointIdentifiedCRNew] Iterations completed: 30
## [pointIdentifiedCRNew] Iterations completed: 31
## [pointIdentifiedCRNew] Iterations completed: 32
## [pointIdentifiedCRNew] Iterations completed: 33
## [pointIdentifiedCRNew] Iterations completed: 34
## [pointIdentifiedCRNew] Iterations completed: 35
## [pointIdentifiedCRNew] Iterations completed: 36
## [pointIdentifiedCRNew] Iterations completed: 37
## [pointIdentifiedCRNew] Iterations completed: 38
## [pointIdentifiedCRNew] Iterations completed: 39
## [pointIdentifiedCRNew] Iterations completed: 40
## [pointIdentifiedCRNew] Iterations completed: 41
## [pointIdentifiedCRNew] Iterations completed: 42
## [pointIdentifiedCRNew] Iterations completed: 43
## [pointIdentifiedCRNew] Iterations completed: 44
## [pointIdentifiedCRNew] Iterations completed: 45
## [pointIdentifiedCRNew] Iterations completed: 46
## [pointIdentifiedCRNew] Iterations completed: 47
## [pointIdentifiedCRNew] Iterations completed: 48
## [pointIdentifiedCRNew] Iterations completed: 49
## [pointIdentifiedCRNew] Iterations completed: 50
```

```
plotCR(cr$estimates)
```



```
print(t(cr$estimates))
```

```
##              par4         par1       par2        par3
##  [1,]   39.2827607  -57.7209441   4.128056 -17.0788114
##  [2,]   37.4958970  -57.1897221  -7.163990 -17.5168830
##  [3,]    1.4169944 -118.0609784   2.537456  31.1599214
##  [4,]   -0.1620064  -98.3623381 -14.342381  30.4106967
##  [5,]  -78.4608344  -38.3484611 -50.301103  72.6189129
##  [6,]    0.1063852  -95.2354152 -16.943216  30.1877987
##  [7,]   34.4053377  -48.7751783 -10.486638 -11.2492795
##  [8,]   11.4866311  -86.0760699 -45.853694  30.4264831
##  [9,]   35.6269639  -51.8350712  -8.064260 -15.5754253
## [10,]  -75.2391736   40.0465659  -3.370745   6.4434296
## [11,]  -71.1995612  -26.0098859 -52.215896  62.0722797
## [12,]   -1.1837278  -96.1778742 -15.618621  31.0166724
## [13,]    0.1763746  -87.3699763 -24.995440  29.2241120
## [14,] -114.4962674  -54.8799196 -22.162121  91.3442059
## [15,]   -0.6012599  -95.8355925 -14.446308  30.4348254
## [16,]   -0.7306043    0.5303944 -12.328869   3.6480825
## [17,]   -0.1494662    0.9428587   4.366348  -0.8424391
## [18,]   31.2896274  -47.0523792 -26.683925  -9.3703232
## [19,]   -0.7635858  -86.6320174 -25.699315  29.0089699
## [20,]    2.1461953   -1.2999370   7.456060   0.1038773
## [21,]   -7.3460572 -104.0313071 -48.451083 117.5541358
## [22,] -124.5334057  -60.4169313 -14.317907  99.9007657
## [23,]  -94.5024793  -41.0765257 -40.875873  77.1835677
## [24,]   34.1361588  -51.5461193  -8.474311 -12.0205848
## [25,]   -8.5292685 -110.8808163 -53.482948 111.6089142
## [26,]  -10.1068396 -117.2361897 -54.626896 117.7990277
## [27,] -107.4264430  -44.6545216 -23.794854  83.8883055
## [28,]  -86.6809091  -36.0282505 -40.857943  71.3353227
## [29,]   32.4568669  -46.8522829 -11.410889 -10.6524599
## [30,] -111.8976640  -41.2843540 -21.742894  80.1063518
## [31,]   -0.1172587   -0.2151784   7.523375  -1.5740995
## [32,]   33.7228496  -52.7991591 -10.567206 -14.2809678
## [33,]   -9.1156347 -107.7523368 -52.455398 121.7630159
## [34,]   -9.1530476 -109.5938879 -51.077353 124.9015654
## [35,]   -7.0280109 -116.7779678 -48.921419 116.7098232
## [36,]   39.3844058  -60.9388271   5.344349 -20.5940712
## [37,]  -70.4725782   39.8578055  -4.580924   2.1373590
## [38,]   12.7694651  -89.8755251 -47.284912  31.6301518
## [39,]   -9.1584201 -100.1854634 -56.272541 116.4417373
## [40,]  -96.2890982   13.1194357  10.338697  39.1378902
## [41,]  -10.1051709 -106.0489269 -46.102463 115.0460891
## [42,]  -10.8283821 -113.1897316 -37.231362 126.7190914
## [43,]  -86.5770877  -35.5887053 -38.600695  73.2053461
## [44,]    0.2394880  -99.7871265 -10.429859  30.7311822
## [45,]  -95.9132525  -40.1069140 -32.912449  78.8038670
## [46,]   -8.8976850 -101.2681884 -55.800161 116.3418968
## [47,]    0.5838945 -109.0812815  -4.461282  31.4355885
## [48,]   -0.2717981   -0.8892662  -3.135809   0.6717187
## [49,]  -10.4265259 -116.0835793 -35.995836 128.9194454
## [50,]   -0.2420447  -85.2630313 -28.581899  28.6961980
```