Technical details and explanation of R code and output for nbreg

Here we provide the technical details and explanation for the example R programs (tested in R v4.0.4) using function `nbreg`. The simulated datasets and sample R programs can also be downloaded from https://github.com/shuaichencode/nbreg.

Description of `nbreg`

Estimation of net benefit regression for possibly censored cost-effectiveness data from randomized or observational studies.

Usage

```
nbreg(Followup, delta, group, Cost=NULL, Eff=NULL,
      Patition.times=NULL, Z=NULL, PS.Z=NULL, interaction=NULL,
      Method=c('SW','PT','CC','AL'), Sep.K=TRUE, PS.trim=0.01,
      Doubly.Robust=FALSE, DR.Reg.SE=TRUE, Eff.only=FALSE,
      Cost.only=FALSE, lambda=NULL, L)
```

Arguments

`Followup`	vector containing continuous positive follow-up time
`Delta`	vector containing binary indicator of event, 1 - complete, 0 - censored
`group`	vector containing binary treatment indicator, 1 - treatment, 0 - control
`Cost`	vector, matrix, or dataframe containing observed total or grouped costs, Cost[i,j] is observed cost of the ith people accumulated in the jth interval
`Eff`	vector, matrix, or dataframe containing observed total or grouped effectiveness, assume effectiveness is survival if not provided
`Patition.times`	vector containing end time points of each time interval, must be monotonically increasing, required if using PT method without Eff provided, also required to truncate grouped costs and effectiveness within time limit L if the cost/effect history is provided.
`Z`	vector, matrix, or dataframe containing covariates for net benefit regression, if not provided, will do unadjusted analysis using simple regression
`PS.Z`	vector, matrix, or dataframe containing covariates matrix for propensity score model using logistic regression, used in doubly

	robust method, if not provided, will fit an unadjusted logistic regression (e.g., for randomized studies)
`interaction`	vector containing covariate names to be included in interactions with treatment, must be a subset of variable names in Z, otherwise will be ignored
`Method`	method for estimation. 'SW' - simple weighted, 'PT' - partitioned, 'CC' - naive complete case, 'AL' - naive all data, doubly robust method requires either 'SW' or 'PT'
`Sep.K`	logical, if TRUE, estimate K (survival function of censoring time, used in inverse probability of censoring weighting) using Kaplan-Meier estimator within each treatment separately, default=TRUE
`PS.trim`	value between (0, 0.5) to trim extreme propensity scores outside the range of (PS.trim, 1-PS.trim), used in doubly robust method, default=0.05
`Doubly.Robust`	logical, if TRUE, perform doubly robust method, default=FALSE
`DR.Reg.SE`	logical, if TRUE, report SE (and p-value) for the regression part of doubly robust method, otherwise only coefficient is reported, default=TRUE
`Eff.only`	logical, if TRUE, fit a regression with dependent variable as effect, default=FALSE
`Cost.only`	logical, if TRUE, fit a regression with dependent variable as cost, default=FALSE
`lambda`	vector or scalar containing willingness-to-pay (WTP) values
`L`	time limit horizon, used to truncate event time, costs and effectiveness if they are outside this time limit, assuming cost and effectiveness are evenly spread within each time interval in truncation

Value

The fitted object is a list, each for a value of WTP lambda. For example, if the fitted object is `fit`, then `fit[[1]]` is a list containing results for the 1st value of WTP. Similarly, `fit[[2]]` is a list containing results for the 2nd value of WTP. If `Eff.only=TRUE` and `Cost.only=TRUE`, the last 2 elements are for effectiveness-only and cost-only regressions, respectively. The following describes the results saved in each element (e.g., `fit[[1]]`) for each WTP value.

`Method`	method for estimation.

`lambda`	value of WTP
`Reg.type`	type of the regression model (`NBR`, `Effect`, or `Cost`)
`est`	vector or scalar containing estimates for coefficients or causal average INB
`se`	vector or scalar containing standard error estimates for coefficients or causal average INB
`covariance`	covariance matrix for coefficient estimates, provided when Doubly.Robust=FALSE
`coef.table`	dataframe for the table of coefficients or causal average INB (estimate, standard error, Wald, and p-value)
`CEAC`	CEAC value for the given WTP value, provided for doubly robust method and net benefit regressions without interaction, not provided for net benefit regressions with interaction due to heterogeneous cost-effectiveness across subgroups
`int.name`	vector containing covariate names in the interactions, only provided for non-doubly robust method
`covar1st`	an example dataframe containing covariates for the 1st patient, only provided for non-doubly robust method
`Regmodel`	coefficient table for the part of net benefit regression, only provided for doubly robust method
`PSmodel`	coefficient table for the part of propensity score model using logistic regression, only provided for doubly robust method
`PS`	estimated propensity scores, only provided for doubly robust method
`group`	vector containing treatment group indicator, only provided for doubly robust method

Examples

Preparation.

```
> require(geepack)
> require(dplyr)
> source("nbreg.r")
> data=read.csv("Censored_CEdata.csv")
> data=data %>% mutate(across(c(Age65, LBBB, Female), as.factor))
> data[,8:23]=data[,8:23]/1000
> lambda=seq(0,6,0.5)
```

The 5th line clarifies that *Age65*, *LBBB* and *Female* are categorical variables. The function `nbreg` works with both continuous and categorical covariates, but categorical

covariates need to be changed to `factor` in advance, which is done with the `as.factor`.

Fit a covariate-adjusted net benefit regression without interaction using SW method with history of costs and effectiveness of QALY.

```
> fit1<-nbreg(Followup=data$survival, delta=data$dead,
      group=data$Trt, Cost=data[,8:22], Eff=data[,24:38],
      Patition.times=1:15, Method='SW', Z=data[,5:7],Eff.only=TRUE,
      Cost.only=TRUE, lambda=lambda, L=10)
```

If `Z=` option is not provided, simple (unadjusted) regressions will be fitted. Although detailed results are saved in `fit1` in R, important results were printed out for each WTP value. The coefficient estimates for group are interpreted as the estimated covariate-adjusted INBs for a given WTP (or covariate-adjusted extra mean QALY/costs for effectiveness-/cost-only regressions) (bold in following output). The output is:

```
Time limit horizon L = 10
Censoring rate = 48.5 %
Method: Simple Weighted

All n = 2000 , Used n = 2000
WTP (lambda) = 0
                Estimate   Std.err         Wald            p
(Intercept) -18.11894523 0.3250253 3.107646e+03 0.000000e+00
group        -2.54654804 0.3901034 4.261321e+01 6.670731e-11
Age651       -0.09676961 0.4146777 5.445737e-02 8.154811e-01
LBBB1         3.22440371 0.4175194 5.964107e+01 1.132427e-14
Female1      -0.40563647 0.3771137 1.156990e+00 2.820907e-01

<snip>

All n = 2000 , Used n = 2000
WTP (lambda) = NA (for Effect)
             Estimate   Std.err       Wald            p
(Intercept)  2.9483057 0.1302542 512.343915 0.000000e+00
group        0.9482411 0.1462802  42.021024 9.029733e-11
Age651      -0.5796059 0.1465873  15.634098 7.685613e-05
LBBB1        1.1995345 0.1460079  67.495209 2.220446e-16
Female1      0.1384287 0.1377967   1.009194 3.150960e-01

All n = 2000 , Used n = 2000
WTP (lambda) = NA (for Cost)
               Estimate   Std.err         Wald            p
(Intercept) 18.11894523 0.3250253 3.107646e+03 0.000000e+00
group        2.54654804 0.3901034 4.261321e+01 6.670731e-11
Age651       0.09676961 0.4146777 5.445737e-02 8.154811e-01
LBBB1       -3.22440371 0.4175194 5.964107e+01 1.132427e-14
```

```
Female1      0.40563647 0.3771137 1.156990e+00 2.820907e-01
```

From the results, we can see that cost-only regression is equivalent to setting WTP lambda=0 and then switching the signs of all coefficient estimates. There is "1" after the covariate name (e.g., LBBB1) which means that LBBB=0 is the reference group and here the coefficient estimate is for LBBB=1. The reference group can be redefined using `relevel` function if needed.

If there are missing values (e.g., in covariates, treatment, or follow-up time), the patients with missing values will be excluded, and hence the `Used n = 2000` in the output will decrease due to discarding observations. We could also use other imputation packages together with `nbreg` (e.g., `mice` R package to perform multiple imputation) to handle the missing data.

To quickly calculate adjusted ICER, we can simply fit cost-only and effect-only regressions without providing `lambda` (or let `lambda=NULL`):

```
fit1_0<-nbreg(Followup=data$survival, delta=data$dead,
      group=data$Trt, Cost=data[,8:22], Eff=data[,24:38],
      Patition.times=1:15, Method='SW', Z=data[,5:7],
      Eff.only=TRUE, Cost.only=TRUE, lambda=NULL, L=10)
```

Fit an unadjusted net benefit regression.

Covariates are not necessary sometimes (e.g., for randomized studies), and we can fit the unadjusted regressions without covariates by removing the option `Z=`.

```
fit1_1<-nbreg(Followup=data$survival, delta=data$dead,
      group=data$Trt, Cost=data[,8:22], Eff=data[,24:38],
      Patition.times=1:15, Method='SW',
      Eff.only=TRUE, Cost.only=TRUE, L=10, lambda=lambda)
```

Fit a net benefit regression for uncensored data.

As a special case, when there is no censoring, all 4 methods (`Method='CC', 'AL', 'SW',` or `'PT'`) are equivalent to ordinary least squares (OLS). The following code demonstrates that `nbreg` also works for the uncensored data (with death indicator as 1 for all patients).

```
data.uncensored=read.csv("True_CEdata.csv")
```

```
data.uncensored=data.uncensored %>%
      mutate(across(c(Age65,LBBB,Female), as.factor))
data.uncensored[,8:23]=data.uncensored[,8:23]/1000

fit9<-nbreg(Followup=data.uncensored$survival,
      delta=data.uncensored$dead, group=data.uncensored$Trt,
      Cost=data.uncensored[,8:22], Eff=data.uncensored[,24:38],
      Patition.times=1:15, Method='CC', Z=data.uncensored[,5:7],
      Eff.only=TRUE, lambda=lambda, L=10)
```

Fit a covariate-adjusted net benefit regression without interaction using SW method using total cost and effectiveness only.

If only a vector of total costs and a vector of total effectiveness were provided, `Patition.times` is not required. However, it is recommended to provide cost and effectiveness history to help better truncate them within *L*. The following code will simply prorate the total costs and effectiveness accumulated in the entire follow-up time into *L*=10 years. For example, if a patient was followed by 14 years with total observed costs of $25,000, the `nbreg` function will calculate the 10-year observed costs by $25,000/14×10=$17,857, assuming costs were spread evenly over time (since cost history is not provided). Two ways can be adopted to improve the estimation for this example: (1) provide yearly cost (and effectiveness) history so that `nbreg` function can calculate 10-year costs more accurately, which might be more convenient than (2); (2) calculate the total observed costs (and effectiveness) within 10 years before fitting the models and then provide them for `nbreg` function. Note that, for (2), if a patient's follow-up time is longer than 10 years, you also need to re-define his/her follow-up time to be 10 years (so that `nbreg` function will know that the total costs occur within the 10 years and will not prorate them), and then introduce this new follow-up time to `nbreg`.

```
> fit2<-nbreg(Followup=data$survival, delta=data$dead,
      group=data$Trt, Cost=data$tot.cost, Eff=data$tot.QALY,
      Method='SW', Z=data[,5:7], Eff.only=TRUE, lambda=lambda, L=10)
```

Fit a covariate-adjusted net benefit regression without interaction using SW method using life years (LY) as effectiveness.

We can also fit a net benefit regression using life years as effectiveness, for which the option `Eff` is not needed, and the follow-up time will be used to calculate effectiveness directly:

```
> fit3<-nbreg(Followup=data$survival, delta=data$dead,
      group=data$Trt, Cost = data[,8:22], Eff=NULL,
      Patition.times=1:15, Method='SW', Z=data[,5:7],
      Eff.only = TRUE, lambda=lambda, L=10)
```

Do not choose too large an *L*.

The `nbreg` function will produce an error message if *L* is larger than the longest follow-up time. Although the `nbreg` function will produce estimates if we choose an *L* slightly smaller than longest follow-up time, the estimates might be instable with large standard errors. Therefore, it is recommended to choose an *L* such that a "reasonable" number of subjects are still being observed at that time (e.g., choose *L* as the upper quartile of follow-up times).

```
> fit3_bigL<-nbreg(Followup=data$survival, delta=data$dead,
group=data$Trt, Cost=data[,8:22], Patition.times=1:15,
Method='SW', Z=data[,5:7], Eff.only=TRUE, lambda=lambda, L=15)

Error in nbreg(Followup = data$survival, delta = data$dead, group =
data$Trt,  :
  Time limit L is greater than longest follow-up time. Choose a
smaller L.
```

Fit a covariate-adjusted net benefit regression without interaction using PT method with history of costs and effectiveness of QALY.

```
> fit4<-nbreg(Followup=data$survival, delta=data$dead,
      group=data$Trt, Cost=data[,8:22], Eff=data[,24:38],
      Patition.times=1:15, Method='PT', Z=data[,5:7], Eff.only=TRUE,
      lambda=lambda, L=10)
```

Fit a net benefit regression for dataset with unequal time intervals.

The `nbreg` function also works for dataset with unequal time intervals. Assuming that the time intervals in dataset do not have the same length (although not true for this dataset), here is example code for such dataset with unequal time intervals. Assume `cost.1` (and `QALY.1`) is the cost (and QALY) accumulated in the first 2 years, `cost.2` (and `QALY.2`) and `cost.3` (and `QALY.3`) are cost (and QALY) accumulated in the following 6 months. Other time intervals keep the same. Thus, the time intervals are [0,2], (2,2.5], (2.5,3], (3,4], ..., (14,15] for cost and QALY histories, which can be introduced to `nbreg` by setting `Patition.times`:

```
> fit4_unequal<-nbreg(Followup=data$survival,delta=data$dead,
```

```
      group=data$Trt,Cost=data[,8:22],Eff=data[,24:38],
      Patition.times=c(2,2.5,3:15), Method='PT', Z=data[,5:7],
      Eff.only=TRUE, lambda=lambda, L=10)
```

Possible issue in variable name when only one covariate is provided.

When provide one covariate as a vector to option `Z`, if the name is not correctly kept, R may automatically drop the covariate name and show the default name Z. The following code provides the covariate stored in the 6th column in `data` without its name:

```
> fit5_1<-nbreg(Followup=data$survival, delta=data$dead,
      group=data$Trt, Cost=data[,8:22], Eff=data[,24:38],
      Patition.times=1:15, Method='PT', Z=data[,5],
      Eff.only=TRUE, lambda=lambda, L=10)

All n = 2000 , Used n = 2000
WTP (lambda) = 0
               Estimate   Std.err        Wald            p
(Intercept) -17.4835554 0.2667715 4295.172294 0.000000e+00
group        -1.3430141 0.3418485   15.434540 8.541279e-05
Z1            0.4236033 0.3733751    1.287146 2.565746e-01

<snip>
```

To fix this issue, we may re-assign the name:

```
> fit5_2<-nbreg(Followup=data$survival, delta=data$dead,
      group=data$Trt, Cost=data[,8:22], Eff=data[,24:38],
      Patition.times=1:15, Method='PT', Z=data.frame(Age65=data[,5]),
      Eff.only=TRUE, lambda=lambda,L=10)

All n = 2000 , Used n = 2000
WTP (lambda) = 0
               Estimate   Std.err        Wald            p
(Intercept) -17.4835554 0.2667715 4295.172294 0.000000e+00
group        -1.3430141 0.3418485   15.434540 8.541279e-05
Age651        0.4236033 0.3733751    1.287146 2.565746e-01

<snip>
```

Alternatively, we can use `drop=FALSE` option to prevent R dropping the name:

```
> fit5_3<-nbreg(Followup=data$survival, delta=data$dead,
      group=data$Trt, Cost=data[,8:22], Eff=data[,24:38],
      Patition.times=1:15, Method='PT', Z=data[,5,drop=FALSE],
      Eff.only=TRUE, lambda=lambda,L=10)

All n = 2000 , Used n = 2000
WTP (lambda) = 0
               Estimate   Std.err        Wald            p
(Intercept) -17.4835554 0.2667715 4295.172294 0.000000e+00
```

```
group        -1.3430141 0.3418485   15.434540 8.541279e-05
Age651        0.4236033 0.3733751    1.287146 2.565746e-01

<snip>
```

Fit a covariate-adjusted net benefit regression with interaction using PT method with history of costs and effectiveness of QALY.

```
> fit6<-nbreg(Followup=data$survival, delta=data$dead,
      group=data$Trt, Cost=data[,8:22], Eff=data[,24:38],
      Patition.times=1:15, Method='PT', Z=data[,5:7],
      interaction=c("LBBB"), Eff.only=TRUE, lambda=lambda, L=10)

All n = 2000 , Used n = 2000
WTP (lambda) = 0
               Estimate   Std.err        Wald            p
(Intercept) -17.8853239 0.3254077 3020.911785 0.000000e+00
group        -3.2362059 0.5033860   41.330437 1.285516e-10
Age651       -0.1705199 0.3892581    0.191900 6.613399e-01
LBBB1         2.4484747 0.5599085   19.123051 1.225557e-05
Female1      -0.3688707 0.3502176    1.109360 2.922212e-01
group:LBBB1   1.5203064 0.7239393    4.410197 3.572472e-02

<snip>

All n = 2000 , Used n = 2000
WTP (lambda) = NA (for Effect)
              Estimate   Std.err       Wald            p
(Intercept)  3.1549636 0.1046052 909.666296 0.000000e+00
group        0.2702074 0.1612147   2.809216 9.372418e-02
Age651      -0.4470651 0.1139687  15.387580 8.756194e-05
LBBB1        0.3520552 0.1660203   4.496753 3.395929e-02
Female1      0.1143624 0.1069441   1.143542 2.849050e-01
group:LBBB1  1.4725388 0.2282211  41.631476 1.102045e-10
```

The option `interaction=` specifies the names of covariates which further have interactions with treatment, where the variable names must be a subset of those provided through the option `Z`. More than one interaction can be included, for example, `interaction = c("Age65", "LBBB")` to include two interactions with treatment, or `interaction = names(data[,5:7])` to include all three possible interactions with treatment.

Perform doubly robust method combining covariate-adjusted net benefit regressions with interaction and propensity scores.

```
> fit7<-nbreg(Followup=data$survival, delta=data$dead,
      group=data$Trt, Cost=data[,8:22], Eff=data[,24:38],
      Patition.times=1:15, Method='PT', Z=data[,5:7],
      interaction=c("LBBB"), PS.Z=data[,5:7], Doubly.Robust=TRUE,
      Eff.only=TRUE, lambda=lambda, L=10)
```

If `PS.Z=` option is not provided, unadjusted logistic regression will be fitted, which assumes that all patients receive the new treatment with the same probability (like a randomized study). Output:

```
Time limit horizon L = 10
Censoring rate = 48.5 %
Method: Doubly Robust Partitioned (estimate is causal average INB)

All n = 2000 , Used n = 2000
WTP (lambda) = 0
       Estimate   Std.err    Wald            p
group -2.425465 0.3736994 42.1255 8.559953e-11

<snip>

All n = 2000 , Used n = 2000
WTP (lambda) = NA (for Effect)
      Estimate   Std.err     Wald p
group 1.029909 0.1176782 76.59588 0
```

Details of the fitted propensity score model part and net benefit regression are also saved in `fit7`. For example, the following code prints out the saved results with the 1st value of WTP (i.e., $\lambda = 0$) for causal average INB, net benefit regression part, and propensity score part, respectively. Similarly, `fit7[[2]]` stores the results with the 2nd value of WTP.

```
> fit7[[1]]$lambda
[1] 0

> fit7[[1]]$Reg.type
[1] "NBR"

> fit7[[1]]$coef.table
       Estimate   Std.err    Wald            p
group -2.425465 0.3736994 42.1255 8.559953e-11

> fit7[[1]]$Regmodel
               Estimate   Std.err  Estimate.1            p
(Intercept) -17.8853239 0.3254077 3020.911785 0.000000e+00
group        -3.2362059 0.5033860   41.330437 1.285516e-10
Age651       -0.1705199 0.3892581    0.191900 6.613399e-01
```

```
LBBB1         2.4484747 0.5599085   19.123051 1.225557e-05
Female1      -0.3688707 0.3502176    1.109360 2.922212e-01
group:LBBB1   1.5203064 0.7239393    4.410197 3.572472e-02

> fit7[[1]]$PSmodel
               Estimate Std. Error    z value     Pr(>|z|)
(Intercept) -0.48458949 0.08068323 -6.0060746 1.900687e-09
Age651      -0.66068101 0.10887725 -6.0681271 1.294105e-09
LBBB1        1.66027659 0.10260678 16.1809641 6.871008e-59
Female1     -0.05843057 0.10005462 -0.5839867 5.592292e-01
```

Examine estimated propensity scores from doubly robust method.

The following code examines the distribution and creates histograms of the estimated propensity scores saved in `fit7`:

```
> summary(fit7[[1]]$PS)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.2308  0.3675  0.6122  0.5245  0.7535  0.7642

> require(ggplot2)
> fit7[[1]]$group <- factor(fit7[[1]]$group, levels = c("0", "1"),
       labels = c("Trt=0", "Trt=1"))      #add labels for group
> ggplot(data,aes(x=fit7[[1]]$PS))+geom_histogram(binwidth =0.1)+
      facet_grid(rows = fit7[[1]]$group)+
      xlab("Estimated propensity score")+ylab("Frequency")+
      xlim(c(0,1))
```

By default, the propensity scores are trimmed at 0.05 to prevent extreme values, that is, propensity scores smaller than 0.05 or larger than 0.95 will be trimmed to 0.05 or 0.95, respectively. However, it is easy to change the trimming value from 0.05 to other value, for example, use option `PS.trim=0.1` in `nbreg` to change the trimming range to (0.1, 0.9).

Construct CEAC plot based on the fitted models.

The following R code creates four CEACs based on the fitted net benefit regression models:

```
> plot(fit4,ylab="Probability new treatment is cost-effective",
      xlab="WTP (in $1000) for one additional QALY", lwd=2,
      pch=19, cex=1.2)

> plot(fit6, subgroup=list(LBBB=0), add=TRUE, col="gray50", lwd=2,
      lty=2, pch=15, cex=1.2)

> plot(fit6, subgroup=list(LBBB=1), add=TRUE, col="gray50", lwd=2,
```

```
      lty=3, pch=17,cex=1.2)

> plot(fit7, add=TRUE, lty=4, col="gray70", lwd=2, pch=0, cex=1.2)

> legend('right', c("Adjusted", "Subgroup:non-LBBB", "Subgroup:LBBB",
      "Doubly Robust"), lty=c(1,2,3,4), lwd=c(2,2,2,2),
      col=c(1, "gray50", "gray50", "gray70"), pch=c(19,15,17,0),
      cex=c(1.2,1.2,1.2,1.2), seg.len =2.8)
```

The first line creates CEAC based on `fit4`, where the first option provides the fitted model from `nbreg`, and other options are parameters for `plot` to customize the curve. The 2nd and 3rd line create CEACs for non-LBBB and LBBB subgroups (adjusted for age and gender), respectively, based on `fit6`, where the option `add=TRUE` adds this new curve to the existing plot instead of creating a new figure. Since `fit6` is from a net benefit regression with interaction between LBBB and covariates, the option `subgroup=list(LBBB=0)` is required to specify the subgroup LBBB=0, where the subgroup is defined by the interaction term(s). If there are two interactions (e.g., interactions between treatment and LBBB and Age65) in the fitted model, we need to specify the values of both LBBB and Age65 to determine a subgroup, e.g., `subgroup=list(LBBB=0, Age65=1)`. The 4th line adds the CEAC based on the doubly robust method (`fit7`), and the last line adds a legend. Details about other parameter options are in the help files of `plot` and `legend` for R.