R package for "Semiparametric Bayesian Analysis of Censored Linear Regression with Errors-in-Covariates"

Description of the package: The purpose of the package is to do a semiparametric Bayesian analysis of the AFT model with covariate measurement error. This package returns the estimates of the regression parameters using a semiparametric Bayesian method when a covariate is subject to measurement error. Along with the semiparametric Bayes estimates, it also produces estimates of the naive and regression calibrartion method. In the later methods we use Buckley-James method to estimate the model parameters. The model is

$$\log(T) = Z^T\beta_1 + X\beta_2 + e,$$

where the distribution of e is completely unspecified. The only constraint we have is that e has a finite variance. Here Z denotes the covariates measured without any error, X is not observed in the data, rather multiple replications of W is observed for each subject, where

$$W = X + U$$

with U being the measurement error. In our proposed semiparametric Bayesian analysis except mean zero and finite variance, we do not make any other assumption on U. Moreover, we treat the distribution of X and e nonparametrically. For the naive and regression calibration method we apply the Buckley-James estimating equations, and in particular we use `bj` function of the R-package `rms`. Here is the function.

```
semipbayes(time=v, status=delta, w, covariate=NULL,
mean.beta=NULL, sigma2.beta=NULL, nmcmc=nmcmc, numb.burn=NULL, numb.thin=NULL, capb=200)
```

Here is the list of input arguments.

- time: It is a positive valued numeric vector of length n.
- status: It is a vector of indicators of length n. For a right censored subject it is 0 otherwise it is 1.
- w: This is an $n \times m$ matrix containting unbiased surrogate for unobserved x. Here m must be greater or equal to 2.
- covariate: This is an nxp matrix of p error-free covariates. If we want to use a single categorical variable with 3 nominal categories, then this will be an $n \times 2$ matrix, where the two columns are the two dummy variables. The default set up is there there is no covariate, that means $p = 0$.
- mean.beta: This is a vector of prior mean for the regression parameters. Note that the length of this vector must be $(p + 1)$. The default values are the regression calibration estimates.

- sigma2.beta: This is a vector of prior variance for the regression parameters. Note that the length of this vector must be $(p + 1)$. The default values are the 10 times variance of the regression calibrartion estimators.
- nmcmc: This is the number of Markov chain Monte Carlo iterations.
- numb.burn: This is the number of burn-in samples. The default value is 1000.
- numb.thin: This is number of observations to be used for thinning. The default value is 10.
- capb: This is the number of bootstrap samples used for estimating the standard error of the regression calibration estimators.

Here is the list of output arguments,

- naive: Here the naive estimates and the corresponding standard errors are produced.
- reg.calib: Here the regression calibration estimates and the corresponding standard errors are produced. The standard errors are produced by the bootstrap resampling method.
- semi.Bayes: Under this semiparametric Bayesian method posterior mean, median and 95 percent credible intervals are produced for each of the regression parameters.

Here is one example.

```
# Simulating a data set
myn <- 1000# sample size
nmcmc <-5000# Number of MCMC iterations
#
mycovz1 <- rnorm(myn,0, 1) # Generating the first covariate
mycovz2<-rbinom(myn, 1, 0.4) # Generating the second covariate
# Generating X
newn=as.integer(myn/3)
myx=0.2*mycovz1+c(rnorm(newn, 0, 0.7), rnorm((myn-newn), 2, 0.3))
#
#
#
e0=log(rgamma(myn, 1, 1))
log.t =0.5+ mycovz1 + 0.5*mycovz2+2*myx+ e0
cns=0.5*myx^2+runif(myn, 0, 500)
myd <- as.numeric(exp(log.t) <= cns)
myv <- apply(cbind(exp(log.t), cns), 1, min)
##
sig.u <-1 # measurement error variance
mym=2
```

```
myu <- matrix(0,nrow=myn, ncol=mym)
ntm=myn*mym
myu=matrix( ((rgamma(ntm, 1, 1)-1)*sig.u), ncol=mym, nrow=myn) # measurement errors
##
myw = myx + myu;
### Note that in the analysis we DO NOT use x (here it is myx)
ut=proc.time()
myout=semipbayes(time=myv, status=myd, myw, covariate=cbind(mycovz1, mycovz2),
mean.beta=NULL, sigma2.beta=NULL, nmcmc=nmcmc, numb.burn=NULL, numb.thin=NULL, capb=200)
vt=proc.time()-ut
> vt
   user  system elapsed
100.339   0.018 100.419
# The elapsed time indicates the computational time in second.
> myout$naive # This is the naive estimates and SE
                Est         SE
covariate 1.1789573 0.06728271
covariate 0.5661054 0.12896123
x         1.3977866 0.04805941

> myout$reg.calib # This is the naive estimates and SE
                Est         SE
covariate 0.9822038 0.07469968
covariate 0.5908344 0.13333264
x         2.0424278 0.09008166
> myout$semi.Bayes # This returns the results of the semiparametric Bayesian method
          Posterior.mean Posterior.median      2.5%     97.5%
covariate      0.9717590        0.9733750 0.8747335 1.0677519
covariate      0.5478233        0.5435816 0.3792630 0.7329126
x              1.9561508        1.9551830 1.8709867 2.0483837
>
```

Here is the second example.

```
# Simulating a data set
myn <- 200# sample size
nmcmc <-5000# Number of MCMC iterations
#
# Generating X
newn=as.integer(myn/3)
myx=c(runif(newn, -1, 1), rnorm((myn-newn), 1.5, 0.2))
#
#
#
e0=log(rgamma(myn, 1, 1))
log.t =0.5+ myx+ e0
cns=0.5*myx^2+runif(myn, 0, 500)
myd <- as.numeric(exp(log.t) <= cns)
myv <- apply(cbind(exp(log.t), cns), 1, min)
```

```
##
sig.u <-1 # measurement error variance
mym=2
myu <- matrix(0,nrow=myn, ncol=mym)
ntm=myn*mym
myu=matrix( ((rnorm(ntm, 0, 1))*sig.u), ncol=mym, nrow=myn) # measurement errors
##
myw = myx + myu;
### Note that in the analysis we DO NOT use x (here it is myx)
ut=proc.time()
myout=semipbayes(time=myv, status=myd, myw,
mean.beta=NULL, sigma2.beta=NULL, nmcmc=nmcmc, numb.burn=NULL, numb.thin=NULL, capb=200)
vt=proc.time()-ut
> vt
   user  system elapsed
 11.052   0.001  11.066
# Elapsed time indicates total compuation time in second.
> myout$naive
       Est       SE
x 0.423189 0.104662
> myout$reg.calib
        Est      SE
x 0.8838029 0.29592
> myout$semi.Bayes
  Posterior.mean Posterior.median      2.5%    97.5%
x        0.87934        0.8602616 0.5306814 1.297967
>
```

References

Buckley J, James I. Linear regression with censored data. *Biometrika* 1979; **66**:429–436.

Sinha, S. and Wang, S. Semiparametric Bayesian analysis of censored linear regression with errors-in-covariates. *Statistical Methods in Medical Research* 2015.