Package ‘aftgee’

June 16, 2020

Title Accelerated Failure Time Model with Generalized Estimating Equations

Version 1.1.5

Description A collection of methods for both the rank-based estimates and least-square estimates to the Accelerated Failure Time (AFT) model. For rank-based estimation, it provides approaches that include the computationally efficient Gehan's weight and the general's weight such as the logrank weight. Details of the rank-based estimation can be found in Chiou et al. (2014) <doi:10.1007/s11222-013-9388-2> and Chiou et al. (2015) <doi:10.1002/sim.6415>. For the least-square estimation, the estimating equation is solved with generalized estimating equations (GEE). Moreover, in multivariate cases, the dependence working correlation structure can be specified in GEE's setting. Details on the least-squares estimation can be found in Chiou et al. (2014) <doi:10.1007/s10985-014-9292-x>.

Depends R (>= 3.4.0)

License GPL (>= 3)

URL http://github.com/stc04003/aftgee

BugReports http://github.com/stc04003/aftgee/issues

Encoding UTF-8

LazyData true

LazyLoad yes

Imports methods, parallel, geepack, survival, BB, MASS

RoxygenNote 6.1.1

NeedsCompilation yes

Author Sy Han Chiou [aut, cre], Sangwook Kang [aut], Jun Yan [aut]

Maintainer Sy Han Chiou <schiou@utdallas.edu>

Repository CRAN

Date/Publication 2020-06-16 16:30:29 UTC
R topics documented:

- aftgee-package ......................................................... 2
- aftgee ................................................................. 3
- aftgee.control ......................................................... 5
- aftsr ................................................................... 6
- export_Surv ............................................................. 9
- QIC ................................................................. 9

Index 11

| aftgee-package | aftgee: Accelerated Failure Time with Generalized Estimating Equation |

Description

A package that uses Generalized Estimating Equations (GEE) to estimate Multivariate Accelerated Failure Time Model (AFT). This package implements recently developed inference procedures for AFT models with both the rank-based approach and the least squares approach. For the rank-based approach, the package allows various weight choices and uses an induced smoothing procedure that leads to much more efficient computation than the linear programming method. With the rank-based estimator as an initial value, the generalized estimating equation approach is used as an extension of the least squares approach to the multivariate case. Additional sampling weights are incorporated to handle missing data needed as in case-cohort studies or general sampling schemes.

Author(s)

Maintainer: Sy Han Chiou <schiou@utdallas.edu>

Authors:
- Sangwook Kang
- Jun Yan

References


**See Also**

Useful links:

- [http://github.com/stc04003/aftgee](http://github.com/stc04003/aftgee)

---

**aftgee**

*Least-Squares Approach for Accelerated Failure Time with Generalized Estimating Equation*

---

**Description**

Fits a semiparametric accelerated failure time (AFT) model with least-squares approach. Generalized estimating equation is generalized to multivariate AFT modeling to account for multivariate dependence through working correlation structures to improve efficiency.

**Usage**

```r
aftgee(formula, data, subset, id = NULL, contrasts = NULL, weights = NULL, margin = NULL, corstr = "independence", binit = "srrgehan", B = 100, control = aftgee.control())
```

**Arguments**

- `formula` a formula expression, of the form `response ~ predictors`. The response is a `Surv` object with right censoring. In the case of no censoring, `aftgee` will return an ordinary least estimate when `corstr = "independence"`. See the documentation of `lm`, `coxph` and `formula` for details.
- `data` an optional data.frame in which to interpret the variables occurring in the `formula`.
- `subset` an optional vector specifying a subset of observations to be used in the fitting process.
- `id` an optional vector used to identify the clusters. If missing, then each individual row of data is presumed to represent a distinct subject. The length of `id` should be the same as the number of observations.
- `contrasts` an optional list.
- `weights` an optional vector of observation weights.
- `margin` a `sformula` vector; default at 1.
corstr a character string specifying the correlation structure. The following are permitted:

- independence
- exchangeable
- ar1
- unstructured
- userdefined
- fixed

binit an optional vector can be either a numeric vector or a character string specifying the initial slope estimator.

- When binit is a vector, its length should be the same as the dimension of covariates.
- When binit is a character string, it should be either lm for simple linear regression, or srrgehan for smoothed Gehan weight estimator.

The default value is "srrgehan".

B a numeric value specifies the resampling number. When B = 0, only the beta estimate will be displayed.

control controls maxiter and tolerance.

Value

An object of class "aftgee" representing the fit. The aftgee object is a list containing at least the following components:

- coefficients a vector of initial value and a vector of point estimates
- coef.res a vector of point estimates
- var.res estimated covariance matrix
- coef.init a vector of initial value
- var.init.mat estimated initial covariance matrix
- binit a character string specifying the initial estimator.
- conv An integer code indicating type of convergence after GEE iteration. 0 indicates successful convergence; 1 indicates that the iteration limit maxit has been reached
- ini.conv An integer code indicating type of convergence for initial value. 0 indicates successful convergence; 1 indicates that the iteration limit maxit has been reached
- conv.step An integer code indicating the step until convergence

References


aftgee.control

Examples

datgen <- function(n = 100, tau = 0.3, cen = 100, dim = 2) {
  id <- rep(1:n, rep(dim, n))
  x1 <- rbinom(dim * n, 1, 0.5)
  x2 <- rnorm(dim * n)
  e <- c(t(exp(MASS::mvrnorm(n = n, mu = rep(0, dim), Sigma = tau + (1 - tau) * diag(dim)))))
  T <- exp(2 + x1 + x2 + e)
  cstime <- runif(n, 0, cen)
  delta <- (T < cstime) * 1
  Y <- pmin(T, cstime)
  out <- data.frame(T = T, Y = Y, delta = delta, x1 = x1, x2 = x2, id = rep(1:n, each = dim))
  out
}
set.seed(1)
mydata <- datgen(n = 50, dim = 2)
summary(aftgee(Surv(Y, delta) ~ x1 + x2, data = mydata, id = id, corstr = "ind", B = 8))
summary(aftgee(Surv(Y, delta) ~ x1 + x2, data = mydata, id = id, corstr = "ex", B = 8))

aftgee.control

Auxiliary for Controlling AFTGEE Fitting

Description

Auxiliary function as user interface for aftgee and aftsrr fitting.

Usage

aftgee.control(maxiter = 50, reltol = 0.001, trace = FALSE,
               seIni = FALSE, parallel = FALSE, parCl = parallel::detectCores()/2,
               gp.pwr = -999)

Arguments

maxiter max number of iteration.
reltol relative error tolerance.
trace a binary variable, determine whether to display output for each iteration.
seIni a logical value indicating whether a new rank-based initial value is computed
       for each resampling sample in variance estimation.
parallel an logical value indicating whether parallel computing is used for resampling
          and bootstrap.
parCl an integer value indicating the number of CPU cores used when parallel = TRUE.
gp.pwr an numerical value indicating the GP parameter when rankWeights = GP. The
default value is half the CPU cores on the current host.
Details

When `trace` is TRUE, output for each iteration is printed to the screen.

Value

A list with the arguments as components.

See Also

`aftgee`

---

**aftsrr**

*Accelerated Failure Time with Smooth Rank Regression*

---

**Description**

Fits a semiparametric accelerated failure time (AFT) model with rank-based approach. General
weights, additional sampling weights and fast sandwich variance estimations are also incorporated.
Estimating equations are solved with Barzilar-Borwein spectral method implemented as `BBsolve`
in package `BB`.

**Usage**

```r
aftsrr(formula, data, subset, id = NULL, contrasts = NULL,
weights = NULL, B = 100, rankWeights = c("gehan", "logrank", "PW",
"GP", "userdefined"), eqType = c("is", "ns", "mis", "mns"),
se = c("NULL", "bootstrap", "MB", "ZLCE", "ZLMB", "sHCF", "sHMB",
"ISCF", "ISMB"), control = list())
```

**Arguments**

- `formula`: a formula expression, of the form `response ~ predictors`. The `response` is
  a `Surv` object object with right censoring. See the documentation of `lm`, `coxph`
  and `formula` for details.
- `data`: an optional data frame in which to interpret the variables occurring in the `formula`.
- `subset`: an optional vector specifying a subset of observations to be used in the fitting
  process.
- `id`: an optional vector used to identify the clusters. If missing, then each individual
  row of data is presumed to represent a distinct subject. The length of `id` should
  be the same as the number of observation.
- `contrasts`: an optional list.
- `weights`: an optional vector of observation weights.
- `B`: a numeric value specifies the resampling number. When `B = 0` or `se = NULL`, only
  the beta estimate will be displayed.
rankWeights a character string specifying the type of general weights. The following are permitted:
logrank logrank weight
dehan Gehan’s weight
PW Prentice-Wilcoxon weight
GP GP class weight
userdefined a user defined weight provided as a vector with length equal to the number of subject. This argument is still under-development.
eqType a character string specifying the type of the estimating equation used to obtain the regression parameters. The following are permitted:
is Regression parameters are estimated by directly solving the induced-smoothing estimating equations. This is the default and recommended method.
nis Regression parameters are estimated by directly solving the nonsmooth estimating equations.
mis Regression parameters are estimated by iterating the monotonic smoothed Gehan-based estimating equations. This is typical when rankWeights = "PW" and rankWeights = "GP".
mns Regression parameters are estimated by iterating the monotonic non-smoothed Gehan-based estimating equations. This is typical when rankWeights = "PW" and rankWeights = "GP".
se a character string specifying the estimating method for the variance-covariance matrix. The following are permitted:
NULL if se is specified as NULL, the variance-covariance matrix will not be computed.
bootstrap nonparametric bootstrap.
MB multiplier resampling.
ZLCF Zeng and Lin’s approach with closed form V, see Details.
ZLMB Zeng and Lin’s approach with empirical V, see Details.
shCF Huang’s approach with closed form V, see Details.
shMB Huang’s approach with empirical V, see Details.
ISCF Johnson and Strawderman’s sandwich variance estimates with closed form V, see Details.
ISMB Johnson and Strawderman’s sandwich variance estimates with empirical V, see Details.
control controls equation solver, maxiter, tolerance, and resampling variance estimation. The available equation solvers are BBSolve and dfsane of the BB package. The default algorithm control parameters are used when these functions are called. However, the monotonicity parameter, M, can be specified by users via the control list. When M is specified, the merit parameter, noimp, is set at

10 * M

. The readers are refered to the BB package for details. Instead of searching for the zero crossing, options including BBoptim and optim will return solution from maximizing the corresponding objective function. When se =
"bootstrap" or se = "MB", an additional argument parallel = TRUE can be specified to enable parallel computation. The number of CPU cores can be specified with parC1, the default number of CPU cores is the integer value of detectCores() / 2.

Details

When se = "bootstrap" or se = "MB", the variance-covariance matrix is estimated through a bootstrap fashion. Bootstrap samples that failed to converge are removed when computing the empirical variance matrix. When bootstrap is not called, we assume the variance-covariance matrix has a sandwich form

$$\Sigma = A^{-1}V(A^{-1})^T,$$

where $V$ is the asymptotic variance of the estimating function and $A$ is the slope matrix. In this package, we provide several methods to estimate the variance-covariance matrix via this sandwich form, depending on how $V$ and $A$ are estimated. Specifically, the asymptotic variance, $V$, can be estimated by either a closed-form formulation (CF) or through bootstrap the estimating equations (MB). On the other hand, the methods to estimate the slope matrix $A$ are the inducing smoothing approach (IS), Zeng and Lin’s approach (ZL), and the smoothed Huang’s approach (sH).

Value

aftsr returns an object of class "aftsr" representing the fit. An object of class "aftsr" is a list containing at least the following components:

- beta  
  A vector of beta estimates

- covmat  
  A list of covariance estimates

- convergence  
  An integer code indicating type of convergence.
  - 0 indicates successful convergence.
  - 1 indicates that the iteration limit maxit has been reached.
  - 2 indicates failure due to stagnation.
  - 3 indicates error in function evaluation.
  - 4 is failure due to exceeding 100 step length reductions in line-search.
  - 5 indicates lack of improvement in objective function.

- bhist  
  When variance = "MB", bhist gives the bootstrap samples.

References


Examples

data(kidney, package = "survival")
fit <- aftsrr(Surv(time, status) ~ age + sex, id = id, 
  data = kidney, se = c("ISMB", "ZLMB"), B = 10)
fit

data(nwtco, package = "survival")
subinx <- sample(1:nrow(nwtco), 668, replace = FALSE)
nwtco$subcohort <- 0
nwtco$subcohort[subinx] <- 1
pn <- table(nwtco$subcohort)[[2]] / sum(table(nwtco$subcohort))
nwtco$hi <- nwtco$rel + (1 - nwtco$rel) * nwtco$subcohort / pn
nwtco$age12 <- nwtco$age / 12
nwtco$study <- nwtco$study - 3
nwtco$histol = nwtco$histol - 1
sub <- nwtco[subinx,]
fit <- aftsrr(Surv(edrel, rel) ~ histol + age12 + study, id = seqno, 
  weights = hi, data = sub, B = 10, se = c("ISMB", "ZLMB"), 
  subset = stage == 4)
summary(fit)

Description

This function is imported from the survival package. See Surv.

QIC

*Quasi Information Criterion*

Description

Implementation based on MES::QIC.geeglm

Usage

QIC(object)
Arguments

object is a aftgee fit

Examples

datgen <- function(n = 100, tau = 0.3, cen = 100, dim = 2) {
  id <- rep(1:n, rep(dim, n))
  x1 <- rbinom(dim * n, 1, 0.5)
  x2 <- rnorm(dim * n)
  e <- c(t(exp(MASS::mvrnorm(n = n, mu = rep(0, dim), Sigma = tau + (1 - tau) * diag(dim)))))
  T <- exp(2 + x1 + x2 + e)
  cstime <- runif(n, 0, cen)
  delta <- (T < cstime) * 1
  Y <- pmin(T, cstime)
  out <- data.frame(T = T, Y = Y, delta = delta, x1 = x1, x2 = x2, id = rep(1:n, each = dim))
  return(out)
}

set.seed(1)
mydata <- datgen(n = 50, dim = 2)
fit1 <- aftgee(Surv(Y, delta) ~ x1 + x2, data = mydata, id = id, corstr = "ind", B = 0)
fit2 <- aftgee(Surv(Y, delta) ~ x1 + x2, data = mydata, id = id, corstr = "ex", B = 0)

QIC(fit1)
QIC(fit2)
Index

*Topic aftgee
  aftgee, 3
  _PACKAGE (aftgee-package), 2

aftgee, 3, 6
aftgee-package, 2
aftgee-packages (aftgee-package), 2
aftgee.control, 5
aftsrr, 6

export_Surv, 9

QIC, 9

Surv, 9
Surv (export_Surv), 9