Package ‘dynfrail’

October 30, 2017

Type Package

Title Fitting Dynamic Frailty Models with the EM Algorithm

Version 0.5.2

Author Theodor Adrian Balan

Maintainer Theodor Adrian Balan `<t.a.balan@lumc.nl>`

Description Fits semiparametric dynamic frailty models according to the methodology of Putter and van Houwelingen (2015) <doi:10.1093/biostatistics/kxv002>. Intermediate models, where the frailty is piecewise constant on prespecified intervals, are also supported. The frailty process is taken to have a specific auto-correlation structure, and the supported distributions include gamma, inverse Gaussian, power variance family (PVF) and positive stable.

License GPL (>= 2)

Depends R (>= 3.3.0), survival

Imports Rcpp (>= 0.12.8), magrittr, tibble, dplyr, tidyr

Suggests frailtyEM, knitr, parfm, ggplot2

LinkingTo Rcpp, RcppArmadillo

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

Collate 'RcppExports.R' 'dynfrail_fit.R' 'dynfrail_aux.R' 'dynfrail.R' 'dynfrail_arguments.R' 'dynfrail_prep.R' 'print.dynfrail.R'

VignetteBuilder knitr

NeedsCompilation yes

Repository CRAN

Date/Publication 2017-10-30 10:11:49 UTC

R topics documented:

dynfrail 2
dynfrail_control 4

dynfrail_dist 5
dynfrail_fit 6
dynfrail_prep 7

Index **9**

`dynfrail` *Fitting dynamic frailty models with the EM algorithm*

Description

Fitting dynamic frailty models with the EM algorithm

Usage

```
dynfrail(formula, data, distribution = dynfrail_dist(),
  control = dynfrail_control(), ...)
```

Arguments

`formula`	A formula that contains on the left hand side an object of the type `Surv` and on the right hand side a `+cluster(id)` statement.
`data`	A data frame in which the formula argument can be evaluated
`distribution`	An object as created by `dynfrail_dist`
`control`	An object as created by `dynfrail_control`
`...`	Other arguments, currently used to warn about deprecated argument names

Details

This function fits dynamic frailty models where the intensity of the process is described by

$$\lambda(t) = Z(t)\exp(\beta^\top x)\lambda_0(t).$$

As in regular frailty models, the random effect is shared by observations from a cluster, or by recurrent event episodes within an individual. This implementation generally follows the lines of Putter & van Houwelingen (2015). The maximum likelihood estimates are obtained with an exact E step.

$Z(t)$ has two parameters: θ plays the role of the spread of the frailty distribution. For the frailty distributons with finite variance (all except the positive stable) this is the inverse of the variance, so that 0 corresponds to infinite variance and infinity to variance 0. The second parameter λ determines how much variation in time is in $Z(t)$, so that

$$cor(Z(t_1), Z(t_2)) = exp(-\lambda(t_2 - t_1)).$$

Note that this heavily depends on the time scale, so the starting value in the `distribution` should reflect that.

By default, the program must calculate $Z(t)$ for each cluster and for each event time point in the data. This is computationally challenging. An option is to use the `nints` argument in the `control` argument. This considers $Z(t)$ to be piecewise constant over `nints + 1` intervals. These intervals are determined automatically so that there are roughly an equal number of observations for each interval. Using `nints = 0` is equivalent to fitting a shared frailty model with the `frailtyEM` package.

It is recommended that the user starts with `nints = 0` and then slowly increase the number of intervals. Other options for performance may be set within the `control` argument. Also, this could be tried out first on a subset of the data.

For computational reasons, the standard errors of θ and λ are not calculated, and the standard errors of the regression coefficients are obtained under the assumption that the frailty distribution is fixed.

Value

A `dynfrail` object that contains the following fields:

`coefficients` A named vector of the estimated regression coefficients

`hazard` The breslow estimate of the baseline hazard at each event time point, in chronological order

`imat` Fisher's information matrix corresponding to the coefficients and hazard, assuming θ, λ constant

`logtheta` The point estimate of the logarithm of the frailty parameter θ. See details.

`loglambda` The point estimate of the logarithm of the autocorrelation parameter λ. See details.

`frail` A `data.frame` containing the variables: `id` (cluster id), `interval` (for piecewise constant frailty, the label of the interval on which the frailty is constant), `Y` (a `Surv` object which determines a starting and a stopping time for each row), `frail` (the empirical Bayes estimates of the piecewise constant frailty corresponding to that specific cluster and that specific time period)

`tev` The time points of the events in the data set, this is the same length as hazard

`loglik` A vector of length two with the log-likelihood of the starting Cox model and the maximized log-likelihood

`formula` The original formula argument

`distribution` The original distribution argument

`control` The original control argument

References

Putter, H., & Van Houwelingen, H. C. (2015). Dynamic frailty models based on compound birth–death processes. Biostatistics, 16(3), 550-564.

Examples

```
# 5 piecewise constant intervals
m2 <- dynfrail(Surv(time, status) ~ rx + sex + cluster(litter),
data = rats,
distribution = dynfrail_dist(n_ints = 4))
```

```
## Not run:
#' # essentially a gamma frailty fit
m1 <- dynfrail(Surv(time, status) ~ rx + sex + cluster(litter),
data = rats,
distribution = dynfrail_dist(n_ints = 0))

# completely semiparametric gamma frailty
m2 <- dynfrail(Surv(time, status) ~ rx + sex + cluster(litter),
data = rats)

## End(Not run)
```

dynfrail_control *Control parameters for dynfrail*

Description

Control parameters for dynfrail

Usage

```
dynfrail_control(nlm_control = list(stepmax = 1), inner_control = list(eps =
  1e-04, maxit = 100, verbose = FALSE, lik_tol = 1))
```

Arguments

`nlm_control`	A list of named arguments to be sent to `nlm` for the outer optimization.
`inner_control`	A list of parameters for the inner optimization. See details.

Details

The `nlm_control` argument should not overalp with `hessian`, `f` or `p`.

The `inner_control` argument should be a list with the following items:

- `eps` A criterion for convergence of the EM algorithm (difference between two consecutive values of the log-likelihood)
- `maxit` The maximum number of iterations between the E step and the M step
- `verbose` Logical, whether details of the optimization should be printed
- `lik_tol` For values higher than this, the algorithm returns a warning when the log-likelihood decreases between EM steps. Technically, this should not happen, but if the parameter θ is somewhere really far from the maximum, numerical problems might lead in very small likelihood decreases.

The starting value of the outer optimization may be set in the `dynfrail_dist()` argument.

Value

An object of the type `dynfrail_control`.

See Also

`dynfrail`, `dynfrail_dist`

Examples

```
dynfrail_control()
# this stops each EM (inner maximization) after 10 iterations, event if it did not
# reach the maximum.
dynfrail_control(inner_control = list(maxit = 10))
```

`dynfrail_dist` *Distribution parameters for dynfrail*

Description

Distribution parameters for dynfrail

Usage

```
dynfrail_dist(dist = "gamma", theta = 2, pvfm = -1/2, lambda = 0.1,
  n_ints = NULL, times = NULL)
```

Arguments

`dist` One of 'gamma', 'stable' or 'pvf'.

`theta` Frailty distribution parameter. Must be >0.

`pvfm` Only relevant if `dist = 'pvf'` is used. It determines which PVF distribution should be used. Must be larger than -1 and not equal to 0.

`lambda` Frailty autocorrelation parameter. Must be >0.

`n_ints` For piece-wise constant frailty, the number of intervals. With `n_ints = 0`, the classical shared frailty scenario is obtained.

`times` A vector of time points which determine the piecewise-constant interval for the frailty. Overrides `n_ints`.

Details

The `theta` and `lambda` arguments must be positive. In the case of gamma or PVF, `theta` is the inverse of the frailty variance, i.e. the larger the `theta` is, the closer the model is to a Cox model. When `dist = "pvf"` and `pvfm = -0.5`, the inverse Gaussian distribution is obtained. For the positive stable distribution, the γ parameter of the Laplace transform is $\theta/(1+\theta)$, with the $alpha$ parameter fixed to 1.

Value

An object of the type `dynfrail_dist`, which is mostly used to denote the supported frailty distributions in a consistent way.

See Also

dynfrail, dynfrail_control

Examples

```
dynfrail_dist()
# Compound Poisson distribution:
dynfrail_dist(dist = 'pvf', theta = 1.5, pvfm = 0.5)
# Inverse Gaussian distribution:
dynfrail_dist(dist = 'pvf')
```

dynfrail_fit *Inner maximization of the log-likelihood*

Description

Inner maximization of the log-likelihood

Usage

```
dynfrail_fit(logfrailtypar, dist, pvfm, Y, Xmat, atrisk, basehaz_line,
  mcox = list(), c_vecs, inner_control, return_loglik = TRUE)
```

Arguments

logfrailtypar A vector containing the natural logarithm of the two parameters (theta - for the distribution, lambda - for the autocorrelation)

dist Argument of dynfrail_dist

pvfm Argument of dynfrail_dist

Y A Surv object obtained by splitting the original data at all the time points where the frailty process takes new values

Xmat A model matrix obtained by splitting the original data at all the time points where the frailty process takes new values

atrisk A list of various calculations that are used in the maximization process.

basehaz_line A vector with the baseline hazard estimate at each right hand side time point from Y (can be 0 for the others)

mcox An initial Cox model

c_vecs A list of the length equal to the number of clusters; each element contains a vector of the length of different values that $Z(t)$ takes in that cluster. Each element of this vector contains the sum of the cumulative hazards associated with that value of the frailty.

inner_control Argument of dynfrail_control

return_loglik Logical. If TRUE, then this just returns the log-likelihood, otherwise it returns also the estimates and information matrix

Details

This is an internal function that is used by `dynfrail`. It is not recommended to use this directly unless you know exactly what you are doing. On the other hand, this might be useful if someone wants, for example, to use different maximizers, or to calculate the log-likelihood at specific values of `theta, lambda`. Most of the input can be produced by `dynfrail_prep`.

Value

The log-likelihood if `return_loglik = TRUE` or a list with the log-likelihood and estimates if `return_loglik = FALSE`.

Examples

```
arglist1 <- dynfrail_prep(Surv(time, status) ~ rx + sex + cluster(litter),
data = rats)

# using list() inside is because of the way that R converts lists and vectors
mod1 <- do.call(dynfrail_fit, c(logfrailtypar = list(log(c(0.5, 0.1))), arglist1))
```

`dynfrail_prep` *Preparation of the input for* `dynfrail_fit`

Description

Preparation of the input for `dynfrail_fit`

Usage

```
dynfrail_prep(formula, data, distribution = dynfrail_dist(),
  control = dynfrail_control(), ...)
```

Arguments

`formula` A formula that contains on the left hand side an object of the type `Surv` and on the right hand side a `+cluster(id)` statement.

`data` A data frame in which the formula argument can be evaluated

`distribution` An object as created by `dynfrail_dist`

`control` An object as created by `dynfrail_control`

`...` Other arguments, currently used to warn about deprecated argument names

Details

This is an internal function of `dynfrail` thath actually does before going to the inner maximization, except for the starting values. The input is identical to that from `dynfrail` A scenario where this would be useful would be to make these calculations and then things would be passed on to `dynfrail_fit`.

Value

A list with what is needed to be used with dynfrail_fit

See Also

dynfrail, dynfrail_fit

Examples

```
arglist1 <- dynfrail_prep(Surv(time, status) ~ rx + sex + cluster(litter),
data = rats)
```

Index

dynfrail, 2, *5–8*
dynfrail_control, *2*, 4, *6*, *7*
dynfrail_dist, *2*, *5*, 5, *6*, *7*
dynfrail_fit, 6, *7*, *8*
dynfrail_prep, *7*, 7