Package ‘phmm’

March 8, 2019

Version 0.7-11
Date 2019-03-05
Title Proportional Hazards Mixed-Effects Model
Maintainer Michael Donohue <mdonohue@usc.edu>
Depends survival, lattice, Matrix
VignetteBuilder knitr
URL https://github.com/mcdonohue/phmm
Suggests knitr, lme4 (>= 1.0)
Description Fits proportional hazards model incorporating random effects using
License GPL-3
Collate 'phmm-package.R' 'formula.R' 'linear.predictors.R' 'phmm.R'
 'pseudoPoisPHMM.R' 'traceHat.R'
RoxygenNote 6.0.1
NeedsCompilation yes
Author Michael Donohue [aut, cre],
Ronghui Xu [aut]
Repository CRAN
Date/Publication 2019-03-08 13:40:07 UTC

R topics documented:

phmm-package .................................................. 2
AIC.phmm .................................................... 3
cAIC.phmm .................................................. 5
linear.predictors ............................................. 7
loglik.cond ................................................... 8
phmm .......................................................... 8
phmm.cond.loglik ........................................... 11
pseudoPoisPHMM ............................................ 11
traceHat ....................................................... 13
Description

Fits proportional hazards model incorporating random effects. The function implements an EM algorithm using Markov Chain Monte Carlo at the E-step as described in Vaida and Xu (2000).

Details

<table>
<thead>
<tr>
<th>Package</th>
<th>phmm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>0.2</td>
</tr>
<tr>
<td>Date</td>
<td>2008-01-15</td>
</tr>
<tr>
<td>Depends</td>
<td>survival</td>
</tr>
<tr>
<td>Suggests</td>
<td>lme4</td>
</tr>
<tr>
<td>License</td>
<td>GPL-2</td>
</tr>
<tr>
<td>Packaged</td>
<td>Fri Jul 11 10:33:57 2008; mdonohue</td>
</tr>
<tr>
<td>Built</td>
<td>R 2.8.0; universal-apple-darwin8.11.1; 2008-11-29 12:05:00; unix</td>
</tr>
</tbody>
</table>

Author(s)

Ronghui Xu, Michael Donohue
Maintainer: Michael Donohue <mdonohue@ucsd.edu>

References


Examples

n <- 50       # total sample size
clust <- 5     # number of clusters
clusters <- rep(1:nclust,each=n/nclust)
beta0 <- c(1,2)
set.seed(13)
# generate phmm data set
Z <- cbind(Z1=sample(0:1,n,replace=TRUE),
          Z2=sample(0:1,n,replace=TRUE),
          Z3=sample(0:1,n,replace=TRUE))
b <- cbind(rep(rnorm(nclust),each=n/nclust),rep(rnorm(nclust),each=n/nclust))
wb <- matrix(0,n,2)
for(j in 1:2) wb[,j] <- Z[,j]*b[,j]
wb <- apply(wb,1,sum)
T <- -log(runif(n,0,1))*exp(-Z[,c('Z1','Z2')]%*%beta0-wb)
C <- runif(n,0,1)
time <- ifelse(T<C,T,C)
event <- ifelse(T<=C,1,0)
mean(event)
phmmd <- data.frame(Z)
phmmd$cluster <- clusters
phmmd$time <- time
phmmd$event <- event

fit.phmm <- phmm(Surv(time, event) ~ Z1 + Z2 + (-1 + Z1 + Z2 | cluster),
                 phmmd, Gbs = 100, Gbsvar = 10000, VARSTART = 1,
                 NINIT = 10, MAXSTEP = 100, CONVERG=90)
summary(fit.phmm)

---

Description

Function calculating the Akaike information criterion for PHMM fitted model objects, according to
the formula $-2 \times \log \text{–likelihood} + k \times \rho$, where $npar$ represents the number of parameters in
the fitted model. The function returns a list of AIC calculations corresponding different likelihood
estimations: conditional and marginal likelihoods calculated by Laplace approximation, reciprocal
importance sampling, and bridge sampling (only implemented for nreff < 3). The default $k = 2$, is
for the usual AIC.

Usage

# S3 method for class 'phmm'
AIC(object, ..., k = 2)
Arguments

object an object of class \texttt{phmm}.

\ldots optionally more fitted model objects.

k numeric, the penalty per parameter to be used; the default \(k = 2\) is the classical AIC.

Value

Returns a list of AIC values corresponding to all available log-likelihood values from the fit. See \texttt{phmm} for details of the log-likelihood values.

References


See Also

\texttt{phmm}, \texttt{AIC}

Examples

\begin{verbatim}
N <- 50  # total sample size
nclust <- 5  # number of clusters
clusters <- rep(1:nclust,each=n/nclust)
beta0 <- c(1,2)
set.seed(13)
# generate phmm data set
Z <- cbind(Z1=sample(0:1,n,replace=TRUE),
Z2=sample(0:1,n,replace=TRUE),
Z3=sample(0:1,n,replace=TRUE))
b <- cbind(rep(rnorm(nclust),each=n/nclust),rep(rnorm(nclust),each=n/nclust))
wb <- matrix(rnorm(n,0,2),nrow=2)
for( j in 1:2 ) wb[,j] <- Z[,j]*b[,j]
wb <- apply(wb,1,sum)
T <- -log(runif(n,0,1))*exp(-Z[,c('Z1','Z2')]*beta0-wb)
C <- runif(n,0,1)
time <- ifelse(T<C,T,C)
event <- ifelse(T<=C,1,0)
mean(event)
phmmx <- data.frame(Z)
phmmx$cluster <- clusters
phmmx$time <- time
phmmx$event <- event

fit.phmm <- phmm(Surv(time, event) ~ Z1 + Z2 + (-1 + Z1 + Z2 | cluster),
phmmx, Gbs = 100, Gbsvar = 1000, VARSTART = 1,
NINIT = 10, MAXSTEP = 100, CONVERG=90)

# Same data can be fit with glmer,
\end{verbatim}
cAIC.phmm

# though the correlation structures are different.
poisphmmmd <- pseudoPoisPHMM(fit.phmm)

library(lme4)
fit.lmer <- glmer(m~-1+as.factor(time)+z1+z2+ (-1+w1+w2|cluster)+offset(log(N)),
as.data.frame(as(poisphmmmd, "matrix")), family=poisson, nAGQ=0)

fixef(fit.lmer)[c("z1","z2")]
fit.phmm$coef

VarCorr(fit.lmer)$cluster
fit.phmm$Sigma

logLik(fit.lmer)
fit.phmm$loglik

traceHat(fit.phmm)
summary(fit.lmer)

### S3 method for class 'phmm'
cAIC(object, method = "direct", ..., k = 2)

#### Argument

- **object**: A fitted PHMM model object of class phmm.
- **method**: Passed to traceHat. Options include "direct", "pseudoPois", or "HaLee". The methods "direct" and "HaLee" are algebraically equivalent.
- **...**: Optionally more fitted model objects.
- **k**: numeric, the penalty per parameter to be used; the default k = 2 conforms with the classical AIC.

---

**Description**

Function calculating the conditional Akaike information criterion (Vaida and Blanchard 2005) for PHMM fitted model objects, according to the formula $-2 \times \log \text{-likelihood} + k \times \rho$, where $\rho$ represents the "effective degrees of freedom" in the sense of Hodges and Sargent (2001). The function uses the log-likelihood conditional on the estimated random effects; and trace of the "hat matrix", using the generalized linear mixed model formulation of PHMM, to estimate $\rho$. The default $k = 2$, conforms with the usual AIC.
Value

Returns a numeric value of the cAIC corresponding to the PHMM fit.

References


See Also

phmm, AIC

Examples

```r
## Not run:
n <- 50    # total sample size
nclust <- 5 # number of clusters
clusters <- rep(1:nclust, each=n/nclust)
beta0 <- c(1,2)
set.seed(13)
#generate phmm data set
Z <- cbind(Z1=sample(0:1,n,replace=TRUE),
Z2=sample(0:1,n,replace=TRUE),
Z3=sample(0:1,n,replace=TRUE))
b <- cbind(rep(rnorm(nclust),each=n/nclust), rep(rnorm(nclust),each=n/nclust))
wb <- matrix(rnorm(p*n,0,2)
for( j in 1:2) wb[,j] <- Z[,j]+b[,j]
wb <- apply(wb,1,sum)
T <- -log(runif(n,0,1)) * exp(-Z[,c('Z1','Z2')]%*%beta0-wb)
C <- runif(n,0,1)
time <- ifelse(T<C,T,C)
event <- ifelse(T<=C,1,0)
mean(event)
phmmd <- data.frame(Z)
phmmd$cluster <- clusters
phmmd$time <- time
phmmd$event <- event

fit.phmm <- phmm(Surv(time, event) ~ Z1 + Z2 + (-1 + Z1 + Z2 | cluster),
phmmd, Gb = 100, Gbvar = 1000, VARSTART = 1,
```
linear.predictors

Description

Internal function for extracting the linear predictors of the PHMM model from an object of class phmm returned by phmm.

Usage

linear.predictors(x)

Arguments

x an object of class phmm.

Value

A vector of estimates from call to phmm of $\beta'x_{ij} + w'_{ij} b_i$.

See Also

phmm
loglik.cond  \hspace{2cm} _{PHMM \ text{conditional log-likelihood}}

**Description**

Function for computing log-likelihood conditional on the estimated random effects from an object of class `phmm` returned by `phmm`.

**Usage**

```r
loglik.cond(x)
```

```r
## S3 method for class 'phmm'
loglik.cond(x)
```

**Arguments**

- `x` an object of class `phmm`.

**Value**

The PHMM log-likelihood conditional on the estimated random effects.

**See Also**

`phmm`, `phmm.cond.loglik`

---

`phmm` \hspace{2cm} _{Proportional \ Hazards \ Model \ with \ Mixed \ Effects}  

**Description**

Fits a proportional hazards regression model incorporating random effects. The function implements an EM algorithm using Markov Chain Monte Carlo (MCMC) at the E-step as described in Vaida and Xu (2000).

**Usage**

```r
phmm(formula, data, subset, na.action = na.fail, Sigma = "identity",
      varcov = "diagonal", NINIT = 10, VARSTART = 1, MAXSTEP = 100,
      CONVERG = 90, Gbs = 100, Gbsvar = 1000, verbose = FALSE,
      maxtime = 120, random)
```
Arguments

- **formula**: model formula for the fixed and random components of the model (as in `lmer`). An intercept is implicitly included in the model by estimation of the error distribution. As a consequence `-1` in the model formula does not have any effect. The left-hand side of the formula must be a `Surv` object.

- **data**: optional data frame in which to interpret the variables occurring in the formulas.

- **subset**: subset of the observations to be used in the fit.

- **na.action**: function to be used to handle any NAs in the data. The user is discouraged to change a default value `na.fail`.

- **Sigma**: initial covariance matrix for the random effects. Defaults to "identity".

- **varcov**: constraint on `Sigma`. Currently only "diagonal" is supported.

- **NINIT**: number of starting values supplied to Adaptive Rejection Metropolis Sampling (ARMS) algorithm.

- **VARSTART**: starting value of the variances of the random effects.

- **MAXSTEP**: number of EM iterations.

- **CONVERG**: iteration after which Gibbs sampling size changes from Gbs to Gbsvar.

- **Gbs**: initial Gibbs sampling size (until CONVERG iterations).

- **Gbsvar**: Gibbs sampling size after CONVERG iterations.

- **verbose**: Set to `TRUE` to print EM steps.

- **maxtime**: maximum time in seconds, before aborting EM iterations. Defaults to 120 seconds.

- **random**: The argument `random` is no longer used. Random components are are expressed in formula.

Details

The proportional hazards model with mixed effects is equipped to handle clustered survival data. The model generalizes the usual frailty model by allowing log-linear multivariate random effects. The software can only handle random effects from a multivariate normal distribution. Maximum likelihood estimates of the regression parameters and variance components is gotten by EM algorithm, with Markov chain Monte Carlo (MCMC) used in the E-step.

Care must be taken to ensure the MCMC-EM algorithm has converged, as the algorithm stops after MAXSTEP iterations. No convergence criteria is implemented. It is advised to plot the estimates at each iteration using the `plot` method. For more on MCMC-EM convergence see Booth and Hobart (1999).

Value

The function produces an object of class "phmm" consisting of:
phmm

References


See Also
survfit, Surv.

Examples

n <- 50       # total sample size
clust <- 5    # number of clusters
clusters <- rep(1:nclust,each=n/clust)
beta0 <- c(1,2)
set.seed(13)
#generate phmm data set
Z <- cbind(Z1=sample(0:1,n,replace=TRUE),
            Z2=sample(0:1,n,replace=TRUE),
            Z3=sample(0:1,n,replace=TRUE))
b <- cbind(rep(rnorm(nclust),each=n/clust),rep(rnorm(nclust),each=n/clust))
wb <- matrix(0,n,2)
for( j in 1:2 ) wb[,j] <- Z[,j]*b[,j]
wb <- apply(wb,1,sum)
T <- log(runif(n,0,1))/exp(-Z[,c('Z1','Z2')]*beta0-wb)
C <- runif(n,0,1)
time <- ifelse(T<C,T,C)
event <- ifelse(T<=C,1,0)
mean(event)
phmmd <- data.frame(Z)
phmmd$cluster <- clusters
phmmd$time <- time
phmmd$event <- event

fit.phmm <- phmm(Surv(time, event) ~ Z1 + Z2 + (-1 + Z1 + Z2 | cluster),
                 phmmd, Gbs = 100, Gbsvar = 1000, VARSTART = 1,
                 NINIT = 10, MAXSTEP = 100, CONVERG=90)
**phmm.cond.loglik**

*PHMM conditional log-likelihood*

**Description**

Function for computing log-likelihood conditional on the estimated random effects from the data and specified parameter estimates of a PHMM.

**Usage**

```r
phmm.cond.loglik(time, delta, z, beta, w, b)
```

**Arguments**

- `time`: Follow-up time (right censored data).
- `delta`: The status indicator (0=alive, 1=dead; or TRUE=dead, FALSE=alive).
- `z`: Numeric matrix (\(n \times n_{\text{fixed}}\)) of covariates for fixed effects.
- `beta`: Fitted fixed effects coefficients (\(p\)-vector).
- `w`: Numeric matrix (\(n \times n_{\text{random}}\)) of covariates for random effects.
- `b`: Numeric matrix (\(n \times n_{\text{random}}\)) of random effects estimates.

**Value**

The PHMM log-likelihood conditional on the estimated random effects.

**See Also**

- `phmm`, `loglik.cond`

---

**pseudoPoisPHMM**

*Pseudo poisson data for fitting PHMM via GLMM*

**Description**

Function for generating a pseudo Poisson data set which can be used to fit a PHMM using GLMM software. This follows the mixed-model extension Whitehead (1980), who described how to fit Cox (fixed effects) models with GLM software.

**Usage**

```r
pseudoPoisPHMM(x)
```
pseudoPoisPHMM

Arguments

x an object of class phmm.

Value

A data.frame with columns:

References


See Also

phmm, tracehat

Examples

## Not run:

n <- 50  # total sample size
nclust <- 5  # number of clusters
clusters <- rep(1:nclust,each=n/nclust)
beta0 <- c(1,2)
set.seed(13)
# generate phmm data set
Z <- cbind(Z1=sample(0:1,n,replace=TRUE),
Z2=sample(0:1,n,replace=TRUE),
Z3=sample(0:1,n,replace=TRUE))
b <- cbind(rep(rnorm(nclust),each=n/nclust),rep(rnorm(nclust),each=n/nclust))
wb <- matrix(0,n,2)
for( j in 1:2) wb[,j] <- Z[,j]*b[,j]
wb <- apply(wb,1,sum)
T <- -log(runif(n,0,1))*exp(-Z[,c('Z1','Z2')]*beta0-wb)
C <- runif(n,0,1)
time <- ifelse(T<C,T,C)
event <- ifelse(T<=C,1,0)
mean(event)
phmmd <- data.frame(Z)
phmmd$cluster <- clusters
phmmd$time <- time
phmmd$event <- event

fit.phmm <- phmm(Surv(time, event) ~ Z1 + Z2 + (-1 + Z1 + Z2 | cluster),
phmmd, Gbs = 100, Gbsvar = 1000, VARSTART = 1,
NINIT = 10, MAXSTEP = 100, CONVERG=90)

# Same data can be fit with glmer,
# though the correlation structures are different.
poisphmmd <- pseudoPoisPHMM(fit.phmm)

library(lme4)
traceHat <- glmer(m~1+as.factor(time)+z1+z2+ (-1+w1+w2|cluster)+offset(log(N)),
 as.data.frame(as(poishmmmd, "matrix")), family=poisson, nAGQ=0)

fixef(fit.lmer)[c("z1","z2")]
fit.phmm$coeff
VarCorr(fit.lmer)$cluster
fit.phmm$Sigma
logLik(fit.lmer)
fit.phmm$loglik

traceHat(fit.phmm)

## End(Not run)

---

**traceHat**

*Trace of the "hat" matrix from PHMM-MCEM fit*

**Description**

Compute trace of the “hat” matrix from PHMM-MCEM fit using a direct approximation method (Donohue, et al, submitted), an approximation via hierarchical likelihoods (Ha et al, 2007), or an approximation via a generalized linear mixed-effects model (GLMM) (Donohue, et al, submitted).

**Usage**

```r
traceHat(x, method = "direct")
```

**Arguments**

- `x`: an object of class `phmm`
- `method`: acceptable values are "direct", "pseudoPois", or "HaLee".

**Value**

The trace of the "hat" matrix which can be used as a measure of complexity of the model.

**References**


See Also

phmm, AIC.phmm

Examples

```r
## Not run:
n <- 50       # total sample size
nclust <- 5   # number of clusters
clusters <- rep(1:nclust,each=n/nclust)
beta0 <- c(1,2)
set.seed(13)
#generate phmm data set
Z <- cbind(Z1=sample(0:1,n,replace=TRUE),
           Z2=sample(0:1,n,replace=TRUE),
           Z3=sample(0:1,n,replace=TRUE))
b <- cbind(rep(rnorm(nclust),each=n/nclust),rep(rnorm(nclust),each=n/nclust))
Wb <- matrix(0,n,2)
for( j in 1:2) Wb[,j] <- Z[,j]*b[,j]
Wb <- apply(Wb,1,sum)
T <- -log(runif(n,0,1))*exp(-Z[,c('Z1','Z2')]%*%beta0-Wb)
C <- runif(n,0,1)
time <- ifelse(T<=C,T,C)
event <- ifelse(T<=C,1,0)
mean(event)
phmd <- data.frame(Z)
phmd$cluster <- clusters
phmd$time <- time
phmd$event <- event

fit.phmm <- phmm(Surv(time, event) ~ Z1 + Z2 + (-1 + Z1 + Z2 | cluster),
                 phmd, Gbs = 100, Gbsvar = 1000, VARSTART = 1,
                 NINIT = 10, MAXSTEP = 100, CONVERG=90)

# Same data can be fit with glmer,
# though the correlation structures are different.
poisphmd <- pseudoPoisPHMM(fit.phmm)

library(lme4)
fit.lmer <- glmer(m=-1+as.factor(time)+Z1+Z2+ 
                 (-1+Z1+Z2) | cluster)+offset(log(N)),
                 as.data.frame(as(poisphmd, "matrix")), family=poisson, nAGQ=0)

fixef(fit.lmer)[c("Z1","Z2")]
fit.phmm$coef

VarCorr(fit.lmer)$cluster
fit.phmm$Sigma
```
\texttt{traceHat}

\begin{verbatim}
logLik(fit.lmer)
fit.phmm$loglik

traceHat(fit.phmm)

## End(Not run)
\end{verbatim}
Index

*Topic package
  phmm-package, 2

*Topic survival
  AIC.phmm, 3
  cAIC.phmm, 5
  linear.predictors, 7
  loglik.cond, 8
  phmm, 8
  phmm.cond.loglik, 11
  pseudoPoisPHMM, 11
  traceHat, 13

AIC, 4–6
AIC.phmm, 3, 14

cAIC (cAIC.phmm), 5
  cAIC.phmm, 5

linear.predictors, 7
lmer, 9
loglik.cond, 8, 11

phmm, 4, 6–8, 8, 11, 12, 14
phmm-package, 2
phmm.cond.loglik, 8, 11
pseudoPoisPHMM, 11

Surv, 9, 10
survfit, 10

traceHat, 5, 12, 13