Package 'pencal'

June 5, 2025

Title Penalized Regression Calibration (PRC) for the Dynamic Prediction of Survival

Version 2.3.0

Description Computes penalized regression calibration (PRC), a statistical method for the dynamic prediction of survival when many longitudinal predictors are available. See Signorelli (2024) <doi:10.32614/RJ-2024-014> and Signorelli et al. (2021) <doi:10.1002/sim.9178> for details.

License GPL (>= 3)

URL https://mirkosignorelli.github.io/r

Depends R (>= 4.2.0)

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.2

Imports doParallel, dplyr, foreach, glmnet, lcmm, magic, MASS, Matrix, methods, nlme, purrr, riskRegression, stats, survcomp, survival, survivalROC

Suggests knitr, ptmixed, rmarkdown, survminer

NeedsCompilation no

Author Mirko Signorelli [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-8102-3356>), Pietro Spitali [ctb], Roula Tsonaka [ctb], Barbara Vreede [ctb]

Maintainer Mirko Signorelli <msignorelli.rpackages@gmail.com>

Repository CRAN

Date/Publication 2025-06-05 10:10:02 UTC

Contents

`fitted_prclmm` *A fitted PRC LMM*

Description

This list contains a fitted PRC LMM, where the CBOCP is computed using 50 cluster bootstrap samples. It is used to reduce the computing time in the example of the function `performance_prc`. The simulated dataset on which the model was fitted was landmarked at t = 2.

Usage

```
data(fitted_prclmm)
```

Format

A list comprising step 2 and step 3 as obtained during the estimation of a PRC LMM

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

See Also

performance_prc

Examples

```
data(fitted_prclmm)
ls(fitted_prclmm)
```

---

fitted_prcmlpmm *A fitted PRC MLPMM*

Description

This list contains a fitted PRC MLPMM. It is used to reduce the computing time in the example of the function survpred_prcmlpmm. The simulated dataset on which the model was fitted was landmarked at t = 2.

Usage

```
data(fitted_prclmm)
```

Format

A list comprising step 2 and step 3 as obtained during the estimation of a PRC MLPMM

Author(s)

Mirko Signorelli

References

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

survpred_prcmlpmm

Examples

```
data(fitted_prcmlpmm)
ls(fitted_prcmlpmm)
```

`fit_lmms` *Step 1 of PRC-LMM (estimation of the linear mixed models)*

Description

This function performs the first step for the estimation of the PRC-LMM model (see references for details)

Usage

```
fit_lmms(y.names, fixefs, ranefs, long.data, surv.data, t.from.base,
  n.boots = 0, n.cores = 1, max.ymissing = 0.2, verbose = TRUE,
  seed = 123, control = list(opt = "optim", niterEM = 500, maxIter = 500))
```

Arguments

`y.names`	character vector with the names of the response variables which the LMMs have to be fitted to
`fixefs`	fixed effects formula for the model, example: `~ time`
`ranefs`	random effects formula for the model, specified using the representation of random effect structures of the `R` package `nlme`
`long.data`	a data frame with the longitudinal predictors, comprehensive of a variable called `id` with the subject ids
`surv.data`	a data frame with the survival data and (if relevant) additional baseline covariates. `surv.data` should at least contain a subject id (called `id`), the time to event outcome (`time`), and binary event variable (`event`)
`t.from.base`	name of the variable containing time from baseline in `long.data`
`n.boots`	number of bootstrap samples to be used in the cluster bootstrap optimism correction procedure (CBOCP). If 0, no bootstrapping is performed
`n.cores`	number of cores to use to parallelize part of the computations. If `ncores = 1` (default), no parallelization is done. Pro tip: you can use `parallel::detectCores()` to check how many cores are available on your computer
`max.ymissing`	maximum proportion of subjects allowed to not have any measurement of a longitudinal response variable. Default is 0.2
`verbose`	if `TRUE` (default and recommended value), information on the ongoing computations is printed in the console
`seed`	random seed used for the bootstrap sampling. Default is `seed = 123`
`control`	a list of control values to be passed to `lme` when fitting the linear mixed models. By default, we set `opt = 'optim'`, `niterEM = 500`, `maxIter = 500`. See `?nlme::lmeControl` for all possible arguments and values

Value

A list containing the following objects:

- `call.info`: a list containing the following function call information: `call`, `y.names`, `fixefs`, `ranefs`;
- `lmm.fits.orig`: a list with the LMMs fitted on the original dataset (it should comprise as many LMMs as the elements of `y.names` are);
- `df.sanitized`: a sanitized version of the supplied `long.data` dataframe, without the longitudinal measurements that are taken after the event or after censoring;
- `n.boots`: number of bootstrap samples;
- `boot.ids`: a list with the ids of bootstrapped subjects (when `n.boots` > `0`);
- `lmms.fits.boot`: a list of lists, which contains the LMMs fitted on each bootstrapped datasets (when `n.boots` > `0`).

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

`simulate_prclmm_data`, `summarize_lmms` (step 2), `fit_prclmm` (step 3), `performance_prc`

Examples

```
# generate example data
set.seed(1234)
p = 4 # number of longitudinal predictors
simdata = simulate_prclmm_data(n = 100, p = p, p.relev = 2,
             seed = 123, t.values = c(0, 0.2, 0.5, 1, 1.5, 2))

# specify options for cluster bootstrap optimism correction
# procedure and for parallel computing
do.bootstrap = FALSE
# IMPORTANT: set do.bootstrap = TRUE to compute the optimism correction!
n.boots = ifelse(do.bootstrap, 100, 0)
more.cores = FALSE
# IMPORTANT: set more.cores = TRUE to parallelize and speed computations up!
if (!more.cores) n.cores = 1
if (more.cores) {
   # identify number of available cores on your machine
   n.cores = parallel::detectCores()
```

```
   if (is.na(n.cores)) n.cores = 8
}

# step 1 of PRC-LMM: estimate the LMMs
y.names = paste('marker', 1:p, sep = '')
step1 = fit_lmms(y.names = y.names,
                 fixefs = ~ age, ranefs = ~ age | id,
                 long.data = simdata$long.data,
                 surv.data = simdata$surv.data,
                 t.from.base = t.from.base,
                 n.boots = n.boots, n.cores = n.cores)
# estimated betas and variances for the 3rd marker:
summary(step1, 'marker3', 'betas')
summary(step1, 'marker3', 'variances')
# usual T table:
summary(step1, 'marker3', 'tTable')
```

`fit_mlpmms` *Step 1 of PRC-MLPMM (estimation of the linear mixed models)*

Description

This function performs the first step for the estimation of the PRC-MLPMM model proposed in Signorelli et al. (2021)

Usage

```
fit_mlpmms(y.names, fixefs, ranef.time, randint.items = TRUE, long.data,
  surv.data, t.from.base, n.boots = 0, n.cores = 1, verbose = TRUE,
  seed = 123, maxiter = 100, conv = rep(0.001, 3),
  lcmm.warnings = FALSE)
```

Arguments

`y.names`	a list with the names of the response variables which the MLPMMs have to be fitted to. Each element in the list contains all the items used to reconstruct a latent biological process of interest
`fixefs`	a fixed effects formula for the model, where the time variable (specified also in `ranef.time`) is included as first element and within the function `contrast()`. Examples: `~ contrast(age)`, `~ contrast(age) + group + treatment`
`ranef.time`	a character with the name of the time variable for which to include a shared random slope
`randint.items`	logical: should item-specific random intercepts be included in the MLCMMs? Default is `TRUE`. It can also be a vector, with different values for different elements of `y.names`
`long.data`	a data frame with the longitudinal predictors, comprehensive of a variable called `id` with the subject ids

surv.data	a data frame with the survival data and (if relevant) additional baseline covariates. surv.data should at least contain a subject id (called id), the time to event outcome (time), and binary event variable (event)
t.from.base	name of the variable containing time from baseline in long.data
n.boots	number of bootstrap samples to be used in the cluster bootstrap optimism correction procedure (CBOCP). If 0, no bootstrapping is performed
n.cores	number of cores to use to parallelize part of the computations. If ncores = 1 (default), no parallelization is done. Pro tip: you can use parallel::detectCores() to check how many cores are available on your computer
verbose	if TRUE (default and recommended value), information on the ongoing computations is printed in the console
seed	random seed used for the bootstrap sampling. Default is seed = 123
maxiter	maximum number of iterations to use when calling the function multlcmm. Default is 100
conv	a vector containing the three convergence criteria (convB, convL and convG) to use when calling the function multlcmm. Default is c(1e-3, 1e-3, 1e-3)
lcmm.warnings	logical. If TRUE, a warning is printed every time the (strict) convergence criteria of the multlcmm function are not met. Default is FALSE

Details

This function is essentially a wrapper of the multlcmm that is meant to simplify the estimation of several MLPMMs. In general, ensuring convergence of the algorithm implemented in multlcmm is sometimes difficult, and it is hard to write a function that can automatically solve all possible convergence problems. fit_mplmms returns a warning when estimation did not converge for one or more MLPMMs. If this happens, try to change the convergence criteria in conv or the relevant randint.items value. If doing this doesn't solve the problem, it is recommended to re-estimate the specific MLPMMs for which estimation didn't converge directly with multlcmm, trying to manually solve the convergence issues

Value

A list containing the following objects:

- call.info: a list containing the following function call information: call, y.names, fixefs, ranef.time, randint.items;
- mlpmm.fits.orig: a list with the MLPMMs fitted on the original dataset (it should comprise as many MLPMMs as the elements of y.names are);
- df.sanitized: a sanitized version of the supplied long.data dataframe, without the longitudinal measurements that are taken after the event or after censoring;
- n.boots: number of bootstrap samples;
- boot.ids: a list with the ids of bootstrapped subjects (when n.boots > 0);
- mlpmm.fits.boot: a list of lists, which contains the MLPMMs fitted on each bootstrapped datasets (when n.boots > 0).

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. To appear in: The R Journal. Preprint: arXiv:2309.15600

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196. DOI: 10.1002/sim.9178

See Also

simulate_prcmlpmm_data, summarize_mlpmms (step 2), fit_prcmlpmm (step 3), performance_prc

Examples

```
# generate example data
set.seed(123)
n.items = c(4,2,2,3,4,2)
simdata = simulate_prcmlpmm_data(n = 100, p = length(n.items),
             p.relev = 3, n.items = n.items,
             type = 'u+b', seed = 1)

# specify options for cluster bootstrap optimism correction
# procedure and for parallel computing
do.bootstrap = FALSE
# IMPORTANT: set do.bootstrap = TRUE to compute the optimism correction!
n.boots = ifelse(do.bootstrap, 100, 0)
more.cores = FALSE
# IMPORTANT: set more.cores = TRUE to speed computations up!
if (!more.cores) n.cores = 2
if (more.cores) {
   # identify number of available cores on your machine
   n.cores = parallel::detectCores()
   if (is.na(n.cores)) n.cores = 2
}

# step 1 of PRC-MLPMM: estimate the MLPMMs
y.names = vector('list', length(n.items))
for (i in 1:length(n.items)) {
  y.names[[i]] = paste('marker', i, '_', 1:n.items[i], sep = '')
}

step1 = fit_mlpmms(y.names, fixefs = ~ contrast(age),
                 ranef.time = age, randint.items = TRUE,
                 long.data = simdata$long.data,
                 surv.data = simdata$surv.data,
                 t.from.base = t.from.base,
                 n.boots = n.boots, n.cores = n.cores)
```

```
# print MLPMM summary for marker 5 (all items involved in that MLPMM):
summary(step1, 'marker5_2')
```

`fit_prclmm` *Step 3 of PRC-LMM (estimation of the penalized Cox model(s))*

Description

This function performs the third step for the estimation of the PRC-LMM model (see references for methodological details)

Usage

```
fit_prclmm(object, surv.data, baseline.covs = NULL, penalty = "ridge",
  standardize = TRUE, pfac.base.covs = 0, cv.seed = 19920207,
  n.alpha.elnet = 11, n.folds.elnet = 5, n.cores = 1, verbose = TRUE)
```

Arguments

`object`	the output of step 2 of the PRC-LMM procedure, as produced by the `summarize_lmms` function
`surv.data`	a data frame with the survival data and (if relevant) additional baseline covariates. `surv.data` should at least contain a subject id (called `id`), the time to event outcome (`time`), and binary event variable (`event`)
`baseline.covs`	a formula specifying the variables (e.g., baseline age) in `surv.data` that should be included as baseline covariates in the penalized Cox model. Example: `baseline.covs = '~ baseline.age'`. Default is `NULL`
`penalty`	the type of penalty function used for regularization. Default is `'ridge'`, other possible values are `'elasticnet'` and `'lasso'`
`standardize`	logical argument: should the predictors (both baseline covariates and predicted random effects) be standardized when included as covariates in the penalized Cox model? Default is `TRUE`
`pfac.base.covs`	a single value, or a vector of values, indicating whether the baseline covariates (if any) should be penalized (1) or not (0). Default is `pfac.base.covs = 0` (no penalization of all baseline covariates)
`cv.seed`	value of the random seed to use for the cross-validation done to select the optimal value of the tuning parameter
`n.alpha.elnet`	number of alpha values for the two-dimensional grid of tuning parameteres in elasticnet. Only relevant if `penalty = 'elasticnet'`. Default is 11, so that the resulting alpha grid is c(1, 0.9, 0.8, ..., 0.1, 0)
`n.folds.elnet`	number of folds to be used for the selection of the tuning parameter in elasticnet. Only relevant if `penalty = 'elasticnet'`. Default is 5

n.cores	number of cores to use to parallelize part of the computations. If ncores = 1 (default), no parallelization is done. Pro tip: you can use parallel::detectCores() to check how many cores are available on your computer
verbose	if TRUE (default and recommended value), information on the ongoing computations is printed in the console

Value

A list containing the following objects:

- call: the function call
- pcox.orig: the penalized Cox model fitted on the original dataset;
- tuning: the values of the tuning parameter(s) selected through cross-validation
- surv.data: the supplied survival data (ordered by subject id)
- n.boots: number of bootstrap samples;
- boot.ids: a list with the ids of bootstrapped subjects (when n.boots > 0);
- pcox.boot: a list where each element is a fitted penalized Cox model for a given bootstrap sample (when n.boots > 0).

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

fit_lmms (step 1), summarize_lmms (step 2), performance_prc

Examples

```
# generate example data
set.seed(1234)
p = 4 # number of longitudinal predictors
simdata = simulate_prclmm_data(n = 100, p = p, p.relev = 2,
             seed = 123, t.values = c(0, 0.2, 0.5, 1, 1.5, 2))

# specify options for cluster bootstrap optimism correction
# procedure and for parallel computing
do.bootstrap = FALSE
# IMPORTANT: set do.bootstrap = TRUE to compute the optimism correction!
n.boots = ifelse(do.bootstrap, 100, 0)
```

```
more.cores = FALSE
# IMPORTANT: set more.cores = TRUE to parallelize and speed computations up!
if (!more.cores) n.cores = 1
if (more.cores) {
   # identify number of available cores on your machine
   n.cores = parallel::detectCores()
   if (is.na(n.cores)) n.cores = 8
}

# step 1 of PRC-LMM: estimate the LMMs
y.names = paste('marker', 1:p, sep = '')
step1 = fit_lmms(y.names = y.names,
                 fixefs = ~ age, ranefs = ~ age | id,
                 long.data = simdata$long.data,
                 surv.data = simdata$surv.data,
                 t.from.base = t.from.base,
                 n.boots = n.boots, n.cores = n.cores)

# step 2 of PRC-LMM: compute the summaries
# of the longitudinal outcomes
step2 = summarize_lmms(object = step1, n.cores = n.cores)

# step 3 of PRC-LMM: fit the penalized Cox models
step3 = fit_prclmm(object = step2, surv.data = simdata$surv.data,
                   baseline.covs = ~ baseline.age,
                   penalty = 'ridge', n.cores = n.cores)
summary(step3)
```

`fit_prcmlpmm` *Step 3 of PRC-MLPMM (estimation of the penalized Cox model(s))*

Description

This function performs the third step for the estimation of the PRC-MLPMM model proposed in Signorelli et al. (2021)

Usage

```
fit_prcmlpmm(object, surv.data, baseline.covs = NULL, include.b0s = TRUE,
  penalty = "ridge", standardize = TRUE, pfac.base.covs = 0,
  cv.seed = 19920207, n.alpha.elnet = 11, n.folds.elnet = 5,
  n.cores = 1, verbose = TRUE)
```

Arguments

`object`	the output of step 2 of the PRC-MLPMM procedure, as produced by the `summarize_mlpmms` function
`surv.data`	a data frame with the survival data and (if relevant) additional baseline covariates. `surv.data` should at least contain a subject id (called `id`), the time to event outcome (`time`), and binary event variable (`event`)

baseline.covs	a formula specifying the variables (e.g., baseline age) in surv.data that should be included as baseline covariates in the penalized Cox model. Example: baseline.covs = '~ baseline.age'. Default is NULL
include.b0s	logical. If TRUE, the PRC-MLPMM(U+B) model is estimated; if FALSE, the PRC-MLPMM(U) model is estimated. See Signorelli et al. (2021) for details
penalty	the type of penalty function used for regularization. Default is 'ridge', other possible values are 'elasticnet' and 'lasso'
standardize	logical argument: should the predicted random effects be standardized when included in the penalized Cox model? Default is TRUE
pfac.base.covs	a single value, or a vector of values, indicating whether the baseline covariates (if any) should be penalized (1) or not (0). Default is pfac.base.covs = 0 (no penalization of all baseline covariates)
cv.seed	value of the random seed to use for the cross-validation done to select the optimal value of the tuning parameter
n.alpha.elnet	number of alpha values for the two-dimensional grid of tuning parameteres in elasticnet. Only relevant if penalty = 'elasticnet'. Default is 11, so that the resulting alpha grid is c(1, 0.9, 0.8, ..., 0.1, 0)
n.folds.elnet	number of folds to be used for the selection of the tuning parameter in elasticnet. Only relevant if penalty = 'elasticnet'. Default is 5
n.cores	number of cores to use to parallelize part of the computations. If ncores = 1 (default), no parallelization is done. Pro tip: you can use parallel::detectCores() to check how many cores are available on your computer
verbose	if TRUE (default and recommended value), information on the ongoing computations is printed in the console

Value

A list containing the following objects:

- call: the function call
- pcox.orig: the penalized Cox model fitted on the original dataset;
- tuning: the values of the tuning parameter(s) selected through cross-validation
- surv.data: the supplied survival data (ordered by subject id)
- n.boots: number of bootstrap samples;
- boot.ids: a list with the ids of bootstrapped subjects (when n.boots > 0);
- pcox.boot: a list where each element is a fitted penalized Cox model for a given bootstrap sample (when n.boots > 0).

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

fit_mlpmms (step 1), summarize_mlpmms (step 2), performance_prc

Examples

```
# generate example data
set.seed(123)
n.items = c(4,2,2,3,4,2)
simdata = simulate_prcmlpmm_data(n = 100, p = length(n.items),
             p.relev = 3, n.items = n.items,
             type = 'u+b', seed = 1)

# specify options for cluster bootstrap optimism correction
# procedure and for parallel computing
do.bootstrap = FALSE
# IMPORTANT: set do.bootstrap = TRUE to compute the optimism correction!
n.boots = ifelse(do.bootstrap, 100, 0)
more.cores = FALSE
# IMPORTANT: set more.cores = TRUE to speed computations up!
if (!more.cores) n.cores = 2
if (more.cores) {
   # identify number of available cores on your machine
   n.cores = parallel::detectCores()
   if (is.na(n.cores)) n.cores = 2
}

# step 1 of PRC-MLPMM: estimate the MLPMMs
y.names = vector('list', length(n.items))
for (i in 1:length(n.items)) {
  y.names[[i]] = paste('marker', i, '_', 1:n.items[i], sep = '')
}

step1 = fit_mlpmms(y.names, fixefs = ~ contrast(age),
                ranef.time = age, randint.items = TRUE,
                long.data = simdata$long.data,
                surv.data = simdata$surv.data,
                t.from.base = t.from.base,
                n.boots = n.boots, n.cores = n.cores)

# step 2 of PRC-MLPMM: compute the summaries
step2 = summarize_mlpmms(object = step1, n.cores = n.cores)

# step 3 of PRC-LMM: fit the penalized Cox models
```

```
step3 = fit_prcmlpmm(object = step2, surv.data = simdata$surv.data,
                     baseline.covs = ~ baseline.age,
                     include.b0s = TRUE,
                     penalty = 'ridge', n.cores = n.cores)
summary(step3)
```

---

pbc2data *pbc2 dataset*

Description

This list contains data from the Mayo Clinic primary biliary cirrhosis (PBC) study (1974-1984). It comprises two datasets, one with the survival and baseline covariates and the other with the longitudinal measurements. The datasets are a rearrangement of the ‘pbc2‘ dataframe from the ‘joineRML‘ package that makes them more suitable for analysis within ‘pencal‘

Usage

```
data(pbc2data)
```

Format

The list contains two data frames:

- `baselineInfo` contains the subject indicator ‘id‘, information about the survival outcome (‘time‘ and ‘event‘) and the covariates ‘baselineAge‘, ‘sex‘ and ‘treatment‘;
- `longitudinalInfo` contains the subject ‘id‘ and the repeated measurement data: ‘age‘ is the age of the individual at each visit, ‘fuptime‘ the follow-up time (time on study), and ‘serBilir‘, ‘serChol‘, ‘albumin‘, ‘alkaline‘, ‘SGOT‘, ‘platelets‘ and ‘prothrombin‘ contain the value of each covariate at the corresponding visit

Author(s)

Mirko Signorelli

Examples

```
data(pbc2data)
head(pbc2data$baselineInfo)
head(pbc2data$longitudinalInfo)
```

pencox *Estimation of a penalized Cox model with time-independent covariates*

Description

This function estimates a penalized Cox model where only time-independent covariates are included as predictors, and then computes a bootstrap optimism correction procedure that is used to validate the predictive performance of the model

Usage

```
pencox(data, formula, penalty = "ridge", standardize = TRUE,
  penalty.factor = 1, n.alpha.elnet = 11, n.folds.elnet = 5,
  n.boots = 0, n.cores = 1, verbose = TRUE)
```

Arguments

`data`	a data frame with one row for each subject.It should at least contain a subject id (called `id`), the time to event outcome (`time`), and the binary censoring indicator (`event`), plus at least one covariate to be included in the linear predictor
`formula`	a formula specifying the variables in `data` to include as predictors in the penalized Cox model
`penalty`	the type of penalty function used for regularization. Default is `'ridge'`, other possible values are `'elasticnet'` and `'lasso'`
`standardize`	logical argument: should the covariates be standardized when included in the penalized Cox model? Default is `TRUE`
`penalty.factor`	a single value, or a vector of values, indicating whether the covariates (if any) should be penalized (1) or not (0). Default is `penalty.factor = 1`
`n.alpha.elnet`	number of alpha values for the two-dimensional grid of tuning parameteres in elasticnet. Only relevant if `penalty = 'elasticnet'`. Default is 11, so that the resulting alpha grid is c(1, 0.9, 0.8, ..., 0.1, 0)
`n.folds.elnet`	number of folds to be used for the selection of the tuning parameter in elasticnet. Only relevant if `penalty = 'elasticnet'`. Default is 5
`n.boots`	number of bootstrap samples to be used in the bootstrap optimism correction procedure. If 0, no bootstrapping is performed
`n.cores`	number of cores to use to parallelize the computation of the CBOCP. If `ncores = 1` (default), no parallelization is done. Pro tip: you can use `parallel::detectCores()` to check how many cores are available on your computer
`verbose`	if `TRUE` (default and recommended value), information on the ongoing computations is printed in the console

Value

A list containing the following objects:

- `call`: the function call
- `pcox.orig`: the penalized Cox model fitted on the original dataset;
- `surv.data`: a data frame with the survival data
- `X.orig`: a data frame with the design matrix used to estimate the Cox model
- `n.boots`: number of bootstrap samples;
- `boot.ids`: a list with the ids of bootstrapped subjects (when `n.boots > 0`);
- `pcox.boot`: a list where each element is a fitted penalized Cox model for a given bootstrap sample (when `n.boots > 0`).

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

`fit_prclmm`, `fit_prcmlpmm`

Examples

```
# generate example data
set.seed(1234)
p = 4 # number of longitudinal predictors
simdata = simulate_prclmm_data(n = 100, p = p, p.relev = 2,
             seed = 123, t.values = c(0, 0.2, 0.5, 1, 1.5, 2))
#create dataframe with baseline measurements only
baseline.visits = simdata$long.data[which(!duplicated(simdata$long.data$id)),]
df = merge(simdata$surv.data, baseline.visits, by = 'id')
df = df[ , -c(5:6)]

do.bootstrap = FALSE
# IMPORTANT: set do.bootstrap = TRUE to compute the optimism correction!
n.boots = ifelse(do.bootstrap, 100, 0)
more.cores = FALSE
# IMPORTANT: set more.cores = TRUE to speed computations up!
if (!more.cores) n.cores = 2
if (more.cores) {
   # identify number of available cores on your machine
   n.cores = parallel::detectCores()
```

```
   if (is.na(n.cores)) n.cores = 2
}

form = as.formula(~ baseline.age + marker1 + marker2
                    + marker3 + marker4)
base.pcox = pencox(data = df,
              formula = form,
              n.boots = n.boots, n.cores = n.cores)
ls(base.pcox)
```

`performance_pencox` *Predictive performance of the penalized Cox model with time-independent covariates*

Description

This function computes the naive and optimism-corrected measures of performance (C index, time-dependent AUC and time-dependent Brier score) for a penalized Cox model with time-independent covariates. The optimism correction is computed based on a cluster bootstrap optimism correction procedure (CBOCP, Signorelli et al., 2021)

Usage

```
performance_pencox(fitted_pencox, metric = c("tdauc", "c", "brier"),
  times = c(2, 3), n.cores = 1, verbose = TRUE)
```

Arguments

`fitted_pencox`	the output of pencox
`metric`	the desired performance measure(s). Options include: 'tdauc', 'c' and 'brier'
`times`	numeric vector with the time points at which to estimate the time-dependent AUC and time-dependent Brier score
`n.cores`	number of cores to use to parallelize part of the computations. If `ncores = 1` (default), no parallelization is done. Pro tip: you can use `parallel::detectCores()` to check how many cores are available on your computer
`verbose`	if `TRUE` (default and recommended value), information on the ongoing computations is printed in the console

Value

A list containing the following objects:

- `call`: the function call;
- `concordance`: a data frame with the naive and optimism-corrected estimates of the concordance (C) index;
- `tdAUC`: a data frame with the naive and optimism-corrected estimates of the time-dependent AUC at the desired time points.

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

pencox

Examples

```
# generate example data
set.seed(1234)
p = 4 # number of longitudinal predictors
simdata = simulate_prclmm_data(n = 100, p = p, p.relev = 2,
             seed = 123, t.values = c(0, 0.5, 1, 1.5, 2))
# create dataframe with baseline measurements only
baseline.visits = simdata$long.data[which(!duplicated(simdata$long.data$id)),]
df = merge(simdata$surv.data, baseline.visits, by = 'id')
df = df[ , -c(5:6)]

do.bootstrap = FALSE
# IMPORTANT: set do.bootstrap = TRUE to compute the optimism correction!
n.boots = ifelse(do.bootstrap, 100, 0)
more.cores = FALSE
# IMPORTANT: set more.cores = TRUE to speed computations up!
if (!more.cores) n.cores = 2
if (more.cores) {
   # identify number of available cores on your machine
   n.cores = parallel::detectCores()
   if (is.na(n.cores)) n.cores = 2
}

form = as.formula(~ baseline.age + marker1 + marker2
                    + marker3 + marker4)
base.pcox = pencox(data = df,
              formula = form,
              n.boots = n.boots, n.cores = n.cores)
ls(base.pcox)

# compute the performance measures
perf = performance_pencox(fitted_pencox = base.pcox,
          metric = 'tdauc', times = 3:5, n.cores = n.cores)
 # use metric = 'brier' for the Brier score and metric = 'c' for the
 # concordance index
```

```
# time-dependent AUC estimates:
ls(perf)
perf$tdAUC
```

`performance_prc` *Predictive performance of the PRC-LMM and PRC-MLPMM models*

Description

This function computes the naive and optimism-corrected measures of performance (C index, time-dependent AUC and time-dependent Brier score) for the PRC models proposed in Signorelli et al. (2021). The optimism correction is computed based on a cluster bootstrap optimism correction procedure (CBOCP)

Usage

```
performance_prc(step2, step3, metric = c("tdauc", "c", "brier"),
  times = c(2, 3), n.cores = 1, verbose = TRUE)
```

Arguments

`step2`	the output of either `summarize_lmms` or `summarize_mlpmms` (step 2 of the estimation of PRC)
`step3`	the output of `fit_prclmm` or `fit_prcmlpmm` (step 3 of PRC)
`metric`	the desired performance measure(s). Options include: 'tdauc', 'c' and 'brier'
`times`	numeric vector with the time points at which to estimate the time-dependent AUC and time-dependent Brier score
`n.cores`	number of cores to use to parallelize part of the computations. If `ncores = 1` (default), no parallelization is done. Pro tip: you can use `parallel::detectCores()` to check how many cores are available on your computer
`verbose`	if `TRUE` (default and recommended value), information on the ongoing computations is printed in the console

Value

A list containing the following objects:

- `call`: the function call;
- `concordance`: a data frame with the naive and optimism-corrected estimates of the concordance (C) index;
- `tdAUC`: a data frame with the naive and optimism-corrected estimates of the time-dependent AUC at the desired time points;
- `Brier`: a data frame with the naive and optimism-corrected estimates of the time-dependent Brier score at the desired time points;

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

for the PRC-LMM model: `fit_lmms` (step 1), `summarize_lmms` (step 2) and `fit_prclmm` (step 3); for the PRC-MLPMM model: `fit_mlpmms` (step 1), `summarize_mlpmms` (step 2) and `fit_prcmlpmm` (step 3).

Examples

```
data(fitted_prclmm)

more.cores = FALSE
# IMPORTANT: set more.cores = TRUE to speed computations up!
if (!more.cores) n.cores = 2
if (more.cores) {
   # identify number of available cores on your machine
   n.cores = parallel::detectCores()
   if (is.na(n.cores)) n.cores = 2
}

# compute the time-dependent AUC
perf = performance_prc(fitted_prclmm$step2, fitted_prclmm$step3,
             metric = 'tdauc', times = c(3, 3.5, 4), n.cores = n.cores)
 # use metric = 'brier' for the Brier score and metric = 'c' for the
 # concordance index

# time-dependent AUC estimates:
ls(perf)
perf$tdAUC
```

`print.prclmm` *Print method for PRC-LMM model fits*

Description

Print method for PRC-LMM model fits

Usage

```
## S3 method for class 'prclmm'
print(x, digits = 4, ...)
```

Arguments

x	an object of class prclmm
digits	number of digits at which the printed estimated regression coefficients should be rounded (default is 4)
...	additional arguments

Value

Summary information about the fitted PRC-LMM model

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

fit_prclmm, summary.prclmm

---

print.prcmlpmm *Print method for PRC-MLPMM model fits*

Description

Print method for PRC-MLPMM model fits

Usage

```
## S3 method for class 'prcmlpmm'
print(x, digits = 4, ...)
```

Arguments

x	an object of class prcmlpmm
digits	number of digits at which the printed estimated regression coefficients should be rounded (default is 4)
...	additional arguments

Value

Summary information about the fitted PRC-MLPMM model

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

fit_prcmlpmm, summary.prcmlpmm

---

simulate_prclmm_data *Simulate data that can be used to fit the PRC-LMM model*

Description

This function allows to simulate a survival outcome from longitudinal predictors following the PRC LMM model (see references for details). Specifically, the longitudinal predictors are simulated from linear mixed models (LMMs), and the survival outcome from a Weibull model where the time to event depends linearly on the baseline age and on the random effects from the LMMs.

Usage

```
simulate_prclmm_data(n = 100, p = 10, p.relev = 4, t.values = c(0, 0.5,
  1, 2), landmark = max(t.values), seed = 1, lambda = 0.2, nu = 2,
  cens.range = c(landmark, 10), base.age.range = c(3, 5), tau.age = 0.2)
```

Arguments

n	sample size
p	number of longitudinal outcomes
p.relev	number of longitudinal outcomes that are associated with the survival outcome (min: 1, max: p)
t.values	vector specifying the time points at which longitudinal measurements are collected (NB: for simplicity, this function assumes a balanced designed; however, pencal is designed to work both with balanced and with unbalanced designs!)
landmark	the landmark time up until which all individuals survived. Default is equal to max(t.values)
seed	random seed (defaults to 1)
lambda	Weibull location parameter, positive
nu	Weibull scale parameter, positive
cens.range	range for censoring times. By default, the minimum of this range is equal to the landmark time
base.age.range	range for age at baseline (set it equal to c(0, 0) if you want all subjects to enter the study at the same age)
tau.age	the coefficient that multiplies baseline age in the linear predictor (like in formula (6) from Signorelli et al. (2021))

Value

A list containing the following elements:

- a dataframe long.data with data on the longitudinal predictors, comprehensive of a subject id (id), baseline age (base.age), time from baseline (t.from.base) and the longitudinal biomarkers;
- a dataframe surv.data with the survival data: a subject id (id), baseline age (baseline.age), the time to event outcome (time) and a binary vector (event) that is 1 if the event is observed, and 0 in case of right-censoring;
- perc.cens the proportion of censored individuals in the simulated dataset;
- theta.true a list containing the true parameter values used to simulate data from the mixed model (beta0 and beta1) and from the Weibull model (tau.age, gamma, delta)

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

Examples

```
# generate example data
simdata = simulate_prclmm_data(n = 20, p = 10, p.relev = 4,
               t.values = c(0, 0.5, 1, 2), landmark = 2,
               seed = 19931101)
# view the longitudinal markers:
if(requireNamespace("ptmixed")) {
  ptmixed::make.spaghetti(x = age, y = marker1,
                id = id, group = id,
                data = simdata$long.data,
                legend.inset = - 1)
 }
# proportion of censored subjects
simdata$censoring.prop
# visualize KM estimate of survival
library(survival)
surv.obj = Surv(time = simdata$surv.data$time,
                event = simdata$surv.data$event)
kaplan <- survfit(surv.obj ~ 1,
                  type="kaplan-meier")
plot(kaplan)
```

`simulate_prcmlpmm_data` *Simulate data that can be used to fit the PRC-LMM model*

Description

This function allows to simulate a survival outcome from longitudinal predictors following the PRC MLPMM model presented in Signorelli et al. (2021). Specifically, the longitudinal predictors are simulated from multivariate latent process mixed models (MLPMMs), and the survival outcome from a Weibull model where the time to event depends on the random effects from the MLPMMs.

Usage

```
simulate_prcmlpmm_data(n = 100, p = 5, p.relev = 2, n.items = c(3, 2,
  3, 4, 1), type = "u", t.values = c(0, 0.5, 1, 2),
  landmark = max(t.values), seed = 1, lambda = 0.2, nu = 2,
  cens.range = c(landmark, 10), base.age.range = c(3, 5), tau.age = 0.2)
```

Arguments

`n`	sample size
`p`	number of longitudinal latent processes
`p.relev`	number of latent processes that are associated with the survival outcome (min: 1, max: p)
`n.items`	number of items that are observed for each latent process of interest. It must be either a scalar, or a vector of length `p`

type	the type of relation between the longitudinal outcomes and survival time. Two values can be used: 'u' refers to the PRC-MLPMM(U) model, and 'u+b' to the PRC-MLPMM(U+B) model presented in Section 2.3 of Signorelli et al. (2021). See the article for the mathematical details
t.values	vector specifying the time points at which longitudinal measurements are collected (NB: for simplicity, this function assumes a balanced designed; however, pencal is designed to work both with balanced and with unbalanced designs!)
landmark	the landmark time up until which all individuals survived. Default is equal to max(t.values)
seed	random seed (defaults to 1)
lambda	Weibull location parameter, positive
nu	Weibull scale parameter, positive
cens.range	range for censoring times. By default, the minimum of this range is equal to the landmark time
base.age.range	range for age at baseline (set it equal to c(0, 0) if you want all subjects to enter the study at the same age)
tau.age	the coefficient that multiplies baseline age in the linear predictor (like in formulas (7) and (8) from Signorelli et al. (2021))

Value

A list containing the following elements:

- a dataframe long.data with data on the longitudinal predictors, comprehensive of a subject id (id), baseline age (base.age), time from baseline (t.from.base) and the longitudinal biomarkers;
- a dataframe surv.data with the survival data: a subject id (id), baseline age (baseline.age), the time to event outcome (time) and a binary vector (event) that is 1 if the event is observed, and 0 in case of right-censoring;
- perc.cens the proportion of censored individuals in the simulated dataset.

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

Examples

```
# generate example data
simdata = simulate_prcmlpmm_data(n = 40, p = 6,
             p.relev = 3, n.items = c(3,4,2,5,4,2),
             type = 'u+b', t.values = c(0, 0.5, 1, 2),
             landmark = 2, seed = 19931101)

# names of the longitudinal outcomes:
names(simdata$long.data)
# markerx_y is the y-th item for latent process (LP) x
# we have 6 latent processes of interest, and for LP1
# we measure 3 items, for LP2 4, for LP3 2 items, and so on

# visualize trajectories of marker1_1
if(requireNamespace("ptmixed")) {
  ptmixed::make.spaghetti(x = age, y = marker1_1,
                 id = id, group = id,
                 data = simdata$long.data,
                 legend.inset = - 1)
 }
# proportion of censored subjects
simdata$censoring.prop
# visualize KM estimate of survival
library(survival)
surv.obj = Surv(time = simdata$surv.data$time,
                event = simdata$surv.data$event)
kaplan <- survfit(surv.obj ~ 1,
                 type="kaplan-meier")
plot(kaplan)
```

`simulate_t_weibull` *Generate survival data from a Weibull model*

Description

This function implements the algorithm proposed by Bender et al. (2005) to simulate survival times from a Weibull model. In essence, this is simply an implementation of the Inverse Transformation Method.

Usage

```
simulate_t_weibull(n, lambda, nu, X, beta, seed = 1)
```

Arguments

`n`	sample size
`lambda`	Weibull location parameter, positive
`nu`	Weibull scale parameter, positive

X	design matrix (n rows, p columns)
beta	p-dimensional vector of regression coefficients associated to X
seed	random seed (defaults to 1)

Value

A vector of survival times

Author(s)

Mirko Signorelli

References

Bender, R., Augustin, T., & Blettner, M. (2005). Generating survival times to simulate Cox proportional hazards models. Statistics in medicine, 24(11), 1713-1723.

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

Examples

```
# generate example data
set.seed(1)
n = 50
X = cbind(matrix(1, n, 1),
   matrix(rnorm(n*9, sd = 0.7), n, 9))
beta = rnorm(10, sd = 0.7)
times = simulate_t_weibull(n = n, lambda = 1, nu = 2,
   X = X, beta = beta)
hist(times, 20)
```

---

summarize_lmms *Step 2 of PRC-LMM (computation of the predicted random effects)*

Description

This function performs the second step for the estimation of the PRC-LMM model (see references for methodological details).

Usage

```
summarize_lmms(object, n.cores = 1, verbose = TRUE)
```

Arguments

object	a list of objects as produced by fit_lmms
n.cores	number of cores to use to parallelize part of the computations. If ncores = 1 (default), no parallelization is done. Pro tip: you can use parallel::detectCores() to check how many cores are available on your computer
verbose	if TRUE (default and recommended value), information on the ongoing computations is printed in the console

Value

A list containing the following objects:

- call: the function call
- ranef.orig: a matrix with the predicted random effects computed for the original data;
- n.boots: number of bootstrap samples;
- boot.ids: a list with the ids of bootstrapped subjects (when n.boots > 0);
- ranef.boot.train: a list where each element is a matrix that contains the predicted random effects for each bootstrap sample (when n.boots > 0);
- ranef.boot.valid: a list where each element is a matrix that contains the predicted random effects on the original data, based on the lmms fitted on the cluster bootstrap samples (when n.boots > 0);

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

fit_lmms (step 1), fit_prclmm (step 3), performance_prc

Examples

```
# generate example data
set.seed(1234)
p = 4 # number of longitudinal predictors
simdata = simulate_prclmm_data(n = 100, p = p, p.relev = 2,
             seed = 123, t.values = c(0, 0.2, 0.5, 1, 1.5, 2))

# specify options for cluster bootstrap optimism correction
# procedure and for parallel computing
```

```
do.bootstrap = FALSE
# IMPORTANT: set do.bootstrap = TRUE to compute the optimism correction!
n.boots = ifelse(do.bootstrap, 100, 0)
more.cores = FALSE
# IMPORTANT: set more.cores = TRUE to parallelize and speed computations up!
if (!more.cores) n.cores = 1
if (more.cores) {
   # identify number of available cores on your machine
   n.cores = parallel::detectCores()
   if (is.na(n.cores)) n.cores = 8
}

# step 1 of PRC-LMM: estimate the LMMs
y.names = paste('marker', 1:p, sep = '')
step1 = fit_lmms(y.names = y.names,
                 fixefs = ~ age, ranefs = ~ age | id,
                 long.data = simdata$long.data,
                 surv.data = simdata$surv.data,
                 t.from.base = t.from.base,
                 n.boots = n.boots, n.cores = n.cores)

# step 2 of PRC-LMM: compute the summaries
# of the longitudinal outcomes
step2 = summarize_lmms(object = step1, n.cores = n.cores)
summary(step2)
```

`summarize_mlpmms` *Step 2 of PRC-MLPMM (computation of the predicted random effects)*

Description

This function performs the second step for the estimation of the PRC-MLPMM model proposed in Signorelli et al. (2021)

Usage

```
summarize_mlpmms(object, n.cores = 1, verbose = TRUE)
```

Arguments

`object`	a list of objects as produced by `fit_mlpmms`
`n.cores`	number of cores to use to parallelize part of the computations. If `ncores = 1` (default), no parallelization is done. Pro tip: you can use `parallel::detectCores()` to check how many cores are available on your computer
`verbose`	if `TRUE` (default and recommended value), information on the ongoing computations is printed in the console

Value

A list containing the following objects:

- `call`: the function call
- `ranef.orig`: a matrix with the predicted random effects computed for the original data;
- `n.boots`: number of bootstrap samples;
- `boot.ids`: a list with the ids of bootstrapped subjects (when `n.boots > 0`);
- `ranef.boot.train`: a list where each element is a matrix that contains the predicted random effects for each bootstrap sample (when `n.boots > 0`);
- `ranef.boot.valid`: a list where each element is a matrix that contains the predicted random effects on the original data, based on the mlpmms fitted on the cluster bootstrap samples (when `n.boots > 0`);

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

`fit_mlpmms` (step 1), `fit_prcmlpmm` (step 3), `performance_prc`

Examples

```
# generate example data
set.seed(123)
n.items = c(4,2,2,3,4,2)
simdata = simulate_prcmlpmm_data(n = 100, p = length(n.items),
             p.relev = 3, n.items = n.items,
             type = 'u+b', seed = 1)

# specify options for cluster bootstrap optimism correction
# procedure and for parallel computing
do.bootstrap = FALSE
# IMPORTANT: set do.bootstrap = TRUE to compute the optimism correction!
n.boots = ifelse(do.bootstrap, 100, 0)
more.cores = FALSE
# IMPORTANT: set more.cores = TRUE to speed computations up!
if (!more.cores) n.cores = 2
if (more.cores) {
   # identify number of available cores on your machine
   n.cores = parallel::detectCores()
```

```
   if (is.na(n.cores)) n.cores = 2
}

# step 1 of PRC-MLPMM: estimate the MLPMMs
y.names = vector('list', length(n.items))
for (i in 1:length(n.items)) {
  y.names[[i]] = paste('marker', i, '_', 1:n.items[i], sep = '')
}

step1 = fit_mlpmms(y.names, fixefs = ~ contrast(age),
                   ranef.time = age, randint.items = TRUE,
                   long.data = simdata$long.data,
                   surv.data = simdata$surv.data,
                   t.from.base = t.from.base,
                   n.boots = n.boots, n.cores = n.cores)

# step 2 of PRC-MLPMM: compute the summaries
step2 = summarize_mlpmms(object = step1, n.cores = n.cores)
summary(step2)
```

`summary.lmmfit` *Extract model fits from step 1 of PRC-LMM*

Description

Summary function to extract the estimated fixed effect parameters and variances of the random effects from an object fitted using ‘fit_lmms‘

Usage

```
## S3 method for class 'lmmfit'
summary(object, yname, what = "betas", ...)
```

Arguments

`object`	the output of ‘fit_lmms‘
`yname`	a character giving the name of the longitudinal variable for which you want to extract information
`what`	one of the following: ‘’betas’‘ for the estimates of the regression coefficients; ‘’tTable’‘ for the usual T table produced by ‘nlme‘; ‘’variances’‘ for the estimates of the variances (and covariances) of the random effects and of the variance of the error term
`...`	additional arguments

Value

A vector containing the estimated fixed-effect parameters if ‘what = ’betas’‘, the usual T table produced by ‘nlme‘ if ‘what = ’tTable’‘, or the estimated variance-covariance matrix of the random effects and the estimated variance of the error if ‘what = ’variances’‘

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

`fit_lmms`

`summary.mlpmmfit` *Extract model fits from step 1 of PRC-LMM*

Description

Utility function to extract the MLPMM summaries from a model fit obtained through 'fit_mlpmms'

Usage

```
## S3 method for class 'mlpmmfit'
summary(object, yname, ...)
```

Arguments

`object`	the output of 'fit_lmms'
`yname`	a character giving the name of one of the longitudinal outcomes modelled within one of the MLPMM
`...`	additional arguments

Value

The model summary as returned by 'summary.multlcmm'

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

`fit_mlpmms` and `summary.multlcmm`

`summary.prclmm` *Summary method for PRC-LMM model fits*

Description

Summary method for PRC-LMM model fits

Usage

```
## S3 method for class 'prclmm'
summary(object, ...)
```

Arguments

`object`	an object of class `prclmm`
`...`	additional arguments

Value

An object of class 'sprclmm'

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

`fit_prclmm`, `print.prclmm`

`summary.prcmlpmm` *Summary method for PRC-MLPMM model fits*

Description

Summary method for PRC-MLPMM model fits

Usage

```
## S3 method for class 'prcmlpmm'
summary(object, ...)
```

Arguments

`object`	an object of class `prcmlpmm`
`...`	additional arguments

Value

An object of class ‘sprcmlpmm‘

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

`fit_prcmlpmm`, `print.prcmlpmm`

summary.ranefs *Summary for step 2 of PRC*

Description

Summary function to extract basic descriptives from ‘summarize_lmms‘ and ‘summarize_mlpmms‘

Usage

```
## S3 method for class 'ranefs'
summary(object, ...)
```

Arguments

object	the output of ‘summarize_lmms‘ or ‘summarize_mlpmms‘
...	additional arguments

Value

Information about number of predicted random effects and sample size

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

summarize_lmms, summarize_mlpmms

survplot_prc *Visualize survival predictions for a fitted PRC model*

Description

Visualize survival predictions for a fitted PRC model

Usage

```
survplot_prc(step1, step2, step3, ids, tmax = 5, res = 0.01, lwd = 1,
  lty = 1, legend.title = "Subject", legend.inset = -0.3,
  legend.space = 1)
```

Arguments

step1	the output of fit_lmms or fit_mlpmms
step2	the output of summarize_lmms or summarize_mlpmms
step3	the output of fit_prclmm or fit_prcmlpmm
ids	a vector with the identifiers of the subjects to show in the plot
tmax	maximum prediction time to consider for the chart. Default is 5
res	resolution at which to evaluate predictions for the chart. Default is 0.01
lwd	line width
lty	line type
legend.title	legend title
legend.inset	moves legend more to the left / right (default is -0.3)
legend.space	interspace between lines in the legend (default is 1)

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Examples

```
# generate example data
simdata = simulate_prclmm_data(n = 100, p = 4, p.relev = 2,
             t.values = c(0, 0.2, 0.5, 1, 1.5, 2),
             landmark = 2, seed = 123)

# estimate the PRC-LMM model
y.names = paste('marker', 1:4, sep = '')
```

```
step1 = fit_lmms(y.names = y.names,
                 fixefs = ~ age, ranefs = ~ age | id,
                 long.data = simdata$long.data,
                 surv.data = simdata$surv.data,
                 t.from.base = t.from.base,
                 n.boots = 0)
step2 = summarize_lmms(object = step1)
step3 = fit_prclmm(object = step2, surv.data = simdata$surv.data,
                   baseline.covs = ~ baseline.age,
                   penalty = 'ridge')

# visualize the predicted survival for subjects 1, 3, 7 and 13
survplot_prc(step1, step2, step3, ids = c(1, 3, 7, 13), tmax = 6)
```

`survpred_prclmm` *Compute the predicted survival probabilities obtained from the PRC models*

Description

This function computes the predicted survival probabilities for the for the PRC-LMM model (see references for methodological details)

Usage

```
survpred_prclmm(step1, step2, step3, times = 1, new.longdata = NULL,
  new.basecovs = NULL, keep.ranef = FALSE)
```

Arguments

`step1`	the output of `fit_lmms` (step 1 of the estimation of PRC-LMM)
`step2`	the output of `summarize_lmms` (step 2 of the estimation of PRC-LMM)
`step3`	the output of `fit_prclmm` (step 3 of the estimation of PRC-LMM)
`times`	numeric vector with the time points at which to estimate the time-dependent AUC
`new.longdata`	longitudinal data if you want to compute predictions for new subjects on which the model was not trained. It should comprise an identifier variable called 'id'. Default is `new.longdata = NULL`
`new.basecovs`	a dataframe with baseline covariates for the new subjects for which predictions are to be computed. It should comprise an identifier variable called 'id'. Only needed if baseline covariates were included in step 3 and `new.longdata` is specified. Default is `new.basecovs = NULL`
`keep.ranef`	should a data frame with the predicted random effects be included in the output? Default is `FALSE`

Value

A list containing the function call (`call`), a data frame with the predicted survival probabilities computed at the supplied time points (`predicted_survival`), and if `keep.ranef = TRUE` also the predicted random effects `predicted_ranefs`.

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

`fit_lmms` (step 1), `summarize_lmms` (step 2) and `fit_prclmm` (step 3)

Examples

```
# generate example data
set.seed(1234)
p = 4 # number of longitudinal predictors
simdata = simulate_prclmm_data(n = 100, p = p, p.relev = 2,
             t.values = c(0, 0.2, 0.5, 1, 1.5, 2),
             landmark = 2, seed = 123)

# step 1 of PRC-LMM: estimate the LMMs
y.names = paste('marker', 1:p, sep = '')
step1 = fit_lmms(y.names = y.names,
                 fixefs = ~ age, ranefs = ~ age | id,
                 long.data = simdata$long.data,
                 surv.data = simdata$surv.data,
                 t.from.base = t.from.base,
                 n.boots = 0)

# step 2 of PRC-LMM: compute the summaries
# of the longitudinal outcomes
step2 = summarize_lmms(object = step1)

# step 3 of PRC-LMM: fit the penalized Cox models
step3 = fit_prclmm(object = step2, surv.data = simdata$surv.data,
                   baseline.covs = ~ baseline.age,
                   penalty = 'ridge')

# predict survival probabilities at times 3 to 6
surv.probs = survpred_prclmm(step1, step2, step3, times = 3:6)
head(surv.probs$predicted_survival)
```

```
# predict survival probabilities for new subjects:
temp = simulate_prclmm_data(n = 10, p = p, p.relev = 2,
      seed = 321, t.values = c(0, 0.2, 0.5, 1, 1.5, 2))
new.longdata = temp$long.data
new.basecovs = temp$surv.data[ , 1:2]
surv.probs.new = survpred_prclmm(step1, step2, step3,
                    times = 3:6,
                    new.longdata = new.longdata,
                    new.basecovs = new.basecovs)
head(surv.probs.new$predicted_survival)
```

`survpred_prcmlpmm` *Compute the predicted survival probabilities obtained from the PRC models*

Description

This function computes the predicted survival probabilities for the for the PRC-MLPMM(U) and PRC-MLPMM(U+B) models proposed in Signorelli et al. (2021)

Usage

```
survpred_prcmlpmm(step2, step3, times = 1)
```

Arguments

`step2`	the output of `summarize_mlpmms` (step 2 of the estimation of PRC-MLPMM)
`step3`	the output of `fit_prcmlpmm` (step 3 of the estimation of PRC-MLPMM)
`times`	numeric vector with the time points at which to estimate the time-dependent AUC

Value

A data frame with the predicted survival probabilities computed at the supplied time points

Author(s)

Mirko Signorelli

References

Signorelli, M. (2024). pencal: an R Package for the Dynamic Prediction of Survival with Many Longitudinal Predictors. The R Journal, 16 (2), 134-153.

Signorelli, M., Spitali, P., Al-Khalili Szigyarto, C, The MARK-MD Consortium, Tsonaka, R. (2021). Penalized regression calibration: a method for the prediction of survival outcomes using complex longitudinal and high-dimensional data. Statistics in Medicine, 40 (27), 6178-6196.

See Also

fit_mlpmms (step 1), summarize_mlpmms (step 2) and fit_prcmlpmm (step 3).

Examples

```
data(fitted_prcmlpmm)

# predict survival probabilities at times 3 to 6
surv.probs = survpred_prcmlpmm(fitted_prcmlpmm$step2,
                fitted_prcmlpmm$step3, times = 3:6)
ls(surv.probs)
head(surv.probs$predicted_survival)
```

Index