# Package: landmaRk (via r-universe)

May 27, 2026

**Title** Time-to-Event Landmark Analysis using an Array of Longitudinal and Survival Sub-Models

**Version** 0.1.1

**Description** Provides a modular end-to-end framework for dynamic risk prediction based on time-to-event and longitudinal data. This allows flexible specifications for the longitudinal and survival sub-models. The 'landmaRk' package enables reproducible benchmarks of different model choices, including cross-validation to assess out-of-sample predictive performance. Methods are described in Velasco-Pardo, Constantine-Cooke, Lees and Vallejos (2026, manuscript under preparation) 'Landmarking with Latent Class Mixed Models for Dynamic Prediction of Time-to-event Data with Heterogeneous Biomarker Trajectories'.

**License** GPL (>= 3)

**BugReports** https://github.com/VallejosGroup/landmaRk/issues

**Encoding** UTF-8

**RoxygenNote** 7.3.3

**Depends** R (>= 4.1.0), survival

**Imports** dplyr, ggplot2, lme4, Matrix, methods, pec, riskRegression, rlang, lcmm (>= 2.2.2), doParallel, foreach, utils, timeROC, prodlim

**Suggests** JMbayes2, knitr, rmarkdown, testthat (>= 3.0.0), xml2, tidyverse, spelling, joineR, survminer, withr

**Config/testthat/edition** 3

**Config/testthat/parallel** true

**LazyData** true

**VignetteBuilder** knitr

**URL** https://vallejosgroup.github.io/landmaRk/, https://github.com/VallejosGroup/landmaRk

**Language** en-GB

**NeedsCompilation** no

**Author** Victor Velasco-Pardo [aut, cre] (ORCID: <https://orcid.org/0000-0002-7166-1573>), Nathan Constantine-Cooke [aut] (ORCID: <https://orcid.org/0000-0002-4437-8713>), Catalina Vallejos [aut] (ORCID: <https://orcid.org/0000-0003-3638-1960>), Charlie Lees [aut] (ORCID: <https://orcid.org/0000-0002-0732-8215>)

**Maintainer** Victor Velasco-Pardo <vvelasco@ed.ac.uk>

**Repository** https://cran.r-universe.dev

**Date/Publication** 2026-05-27 10:00:16 UTC

**RemoteUrl** https://github.com/cran/landmaRk

**RemoteRef** HEAD

**RemoteSha** 2ef85caa258ac32cca76ee005bbe856bea23d0d3

# Contents

## check_lcmm_convergence *Checks convergence of lcmm models*

### Description

Checks convergence of lcmm models

### Usage

```
check_lcmm_convergence(x)
```

### Arguments

| | |
|---|---|
| x | An object of class LandmarkAnalysis. |

### Value

No return value. Issues a message if all lcmm models converged, or a warning for each model that did not converge.

## compute_risk_sets *Compute the list of individuals at risk at landmark times*

### Description

Compute the list of individuals at risk at landmark times

### Usage

```
compute_risk_sets(x, landmarks, .warn_when_less_than = 0, ...)
```

### Arguments

| | |
|---|---|
| x | An object of class LandmarkAnalysis. |
| landmarks | Numeric vector of landmark times |
| .warn_when_less_than | Integer indicating that a warning will be raised when the number of observations prior to a landmark time is less than that integer for certain individuals. |
| ... | Additional arguments (not used) |

### Details

A risk set describes all subjects still at risk (i.e., not experienced the event of interest or censored) at a given time. In `LandmarkAnalysis`, risk sets define which subjects should be included in the longitudinal and survival sub-models for each landmark time.

The risk sets are stored in the `risk_sets` slot of the `LandmarkAnalysis` object, where each risk set is a list of indices corresponding to the subjects at risk at the respective landmark time.

### Value

An object of class `LandmarkAnalysis` including desired risk sets for the specified landmark times.

---

## `compute_risk_sets,LandmarkAnalysis-method`

*Compute the list of individuals at risk at landmark times*

---

### Description

Compute the list of individuals at risk at landmark times

### Usage

```
## S4 method for signature 'LandmarkAnalysis'
compute_risk_sets(x, landmarks, .warn_when_less_than = 0, ...)
```

### Arguments

| | |
|---|---|
| `x` | An object of class `LandmarkAnalysis`. |
| `landmarks` | Numeric vector of landmark times |
| `.warn_when_less_than` | Integer indicating that a warning will be raised when the number of observations prior to a landmark time is less than that integer for certain individuals. |
| `...` | Additional arguments (not used) |

### Details

A risk set describes all subjects still at risk (i.e., not experienced the event of interest or censored) at a given time. In `LandmarkAnalysis`, risk sets define which subjects should be included in the longitudinal and survival sub-models for each landmark time.

The risk sets are stored in the `risk_sets` slot of the `LandmarkAnalysis` object, where each risk set is a list of indices corresponding to the subjects at risk at the respective landmark time.

### Value

An object of class `LandmarkAnalysis`, including desired risk sets for the relevant landmark times.

`epileptic` *Dose calibration of anti-epileptic drugs data*

## Description

The SANAD (Standard and New Anti-epileptic Drugs) study (Marson et al., 2007) is a randomized control trial of standard and new anti-epileptic drugs, comparing effects on longer term clinical outcomes. The data consists of longitudinal measurements of calibrated dose for the groups randomized to a standard drug (CBZ) and a new drug (LTG). The objective of the analysis is to investigate the effect of drug titration on the relative effects of LTG and CBZ on treatment failure (withdrawal of the randomized drug). There are several baseline covariates available, and also data on the time to withdrawal from randomized drug. This version of the data has been modified from the joineR package.

## Usage

```
data(epileptic)
```

## Format

This is a data frame in the unbalanced format, that is, with one row per observation. The data consists of columns for patient identifier, time of measurement, calibrated dose, baseline covariates, and survival data. The column names are identified as follows:

`id` integer: patient identifier.

`time` integer: timing of clinic visit at which dose recorded (days).

`with.time` integer: time of drug withdrawal/maximum follow up time (days).

`with.status` censoring indicator (`1` = withdrawal of randomized drug and `0` = not withdrawn from randomized drug/lost to follow up).

`dose` numeric: calibrated dose.

`treat` factor: randomized treatment (CBZ or LTG).

`age` numeric: age of patient at randomization (years).

`gender` factor: gender of patient. `F` = female, `M` = male.

`learn.dis` factor: learning disability.

## Source

SANAD Trial Group, University of Liverpool. Data modified from the joineR package.

## References

Marson AG, Appleton R, Baker GA, et al. A randomised controlled trial examining longer-term outcomes of standard versus new antiepileptic drugs. The SANAD Trial. *Health Tech Assess*. 2007; **11(37)**.

Marson AG, Al-Kharusi AM, Alwaidh M, et al. The SANAD study of effectiveness of carbamazepine, gabapentin, lamotrigine, oxcarbazepine, or topiramate for treatment of partial epilepsy: an unblinded randomised controlled trial. *Lancet*. 2007; **365**: 2007-2013.

Williamson PR, Kolamunnage-Dona R, Philipson P, Marson AG. Joint modelling of longitudinal and competing risks data. *Stats Med.* 2008; **27(30)**: 6426-6438.

## See Also

joineR::epileptic, which this dataset was modified from.

---

`fit_longitudinal` *Fits the specified longitudinal model for time-varying covariates up to the landmark times*

---

## Description

Fits the specified longitudinal model for time-varying covariates up to the landmark times

## Usage

```
fit_longitudinal(
  x,
  landmarks,
  method,
  formula,
  dynamic_covariates,
  validation_fold = 0,
  cores = getOption("Ncpus", 1L),
  .warn_when_prop_few_obs = 0.25,
  ...
)
```

## Arguments

| | |
|---|---|
| `x` | An object of class `LandmarkAnalysis`. |
| `landmarks` | A vector of Landmark times. |
| `method` | Either `"lcmm"` or `"lme4"` or a function for fitting a longitudinal data model, where the first argument is a formula, and also has a `data` argument. |
| `formula` | A formula to be used in longitudinal sub-model fitting. |
| `dynamic_covariates` | Vector of time-varying covariates to be modelled as the outcome of a longitudinal model. |
| `validation_fold` | If positive, cross-validation fold where model is fitted. If 0 (default), model fitting is performed using the complete dataset. |

`cores` Number of cores/threads to be used for parallel computation on Linux and MacOS. Defaults to either `options("Ncpus")` if set, or 1 (single threaded) otherwise. Only single-threaded computation is currently supported on Windows.

`.warn_when_prop_few_obs` Threshold proportion (0-1) for warning when individuals have 0 or 1 observations. Defaults to 0.25 (i.e., warn when 25% or more individuals have few observations).

`...` Additional arguments passed to the longitudinal model fitting function (e.g. number of classes/clusters for lcmm).

## Details

**Parallel processing:**
As the longitudinal model for each landmark time is independent of the longitudinal models for other landmark times, parallel processing can be used to vastly speed up computation. However, due to issues with parallel processing in R, currently only Unix-like operating systems are supported by `landmaRk`.

## Value

An object of class `LandmarkAnalysis`.

## See Also

`lcmm::hlme()` and `lme4::lmer()` for additional arguments.

---

# `fit_longitudinal,LandmarkAnalysis-method`

*Fits the specified longitudinal model for time-varying covariates up to the landmark times*

---

## Description

Fits the specified longitudinal model for time-varying covariates up to the landmark times

## Usage

```
## S4 method for signature 'LandmarkAnalysis'
fit_longitudinal(
  x,
  landmarks,
  method,
  formula,
  dynamic_covariates,
  validation_fold = 0,
  cores = getOption("Ncpus", 1L),
  .warn_when_prop_few_obs = 0.25,
  ...
)
```

## Arguments

| | |
|---|---|
| x | An object of class LandmarkAnalysis. |
| landmarks | A vector of Landmark times. |
| method | Either "lcmm" or "lme4" or a function for fitting a longitudinal data model, where the first argument is a formula, and also has a data argument. |
| formula | A formula to be used in longitudinal sub-model fitting. |
| dynamic_covariates | Vector of time-varying covariates to be modelled as the outcome of a longitudinal model. |
| validation_fold | If positive, cross-validation fold where model is fitted. If 0 (default), model fitting is performed using the complete dataset. |
| cores | Number of cores/threads to be used for parallel computation on Linux and MacOS. Defaults to either options("Ncpus") if set, or 1 (single threaded) otherwise. Only single-threaded computation is currently supported on Windows. |
| .warn_when_prop_few_obs | Threshold proportion (0-1) for warning when individuals have 0 or 1 observations. Defaults to 0.25 (i.e., warn when 25% or more individuals have few observations). |
| ... | Additional arguments passed to the longitudinal model fitting function (e.g. number of classes/clusters for lcmm). |

## Details

### Parallel processing:

As the longitudinal model for each landmark time is independent of the longitudinal models for other landmark times, parallel processing can be used to vastly speed up computation. However, due to issues with parallel processing in R, currently only Unix-like operating systems are supported by landmaRk.

## Value

An object of class LandmarkAnalysis.

## See Also

lcmm::hlme() and lme4::lmer() for additional arguments.

`fit_survival` *Fits the specified survival model at the landmark times and up to the horizon times specified by the user*

## Description

Fits the specified survival model at the landmark times and up to the horizon times specified by the user

## Usage

```
fit_survival(
  x,
  formula,
  landmarks,
  horizons,
  method,
  dynamic_covariates = c(),
  include_clusters = FALSE,
  censor_at_horizon = FALSE,
  validation_fold = 0
)
```

## Arguments

| | |
|---|---|
| `x` | An object of class `LandmarkAnalysis`. |
| `formula` | A formula to be used in survival sub-model fitting. |
| `landmarks` | Numeric vector of landmark times. |
| `horizons` | Vector of prediction horizons up to when the survival submodel is fitted. |
| `method` | Method for survival analysis, either "survfit" or "coxph". |
| `dynamic_covariates` | Vector of time-varying covariates to be used in the survival model. |
| `include_clusters` | Boolean indicating whether to propagate cluster membership to survival analysis. |
| `censor_at_horizon` | Boolean indicating whether to censor observations at horizon times |
| `validation_fold` | If positive, cross-validation fold where model is fitted. If 0 (default), model fitting is performed on the complete dataset. |

## Details

**Mathematical formulation:**

This function estimates the conditional probability of survival to horizon $s + w$, conditioned on having survived to the landmark time, $s$, that is

$$\pi_i(s + w|s) = P(T_i > s + w|T_i \geq s, \bar{x}_i(s)),$$

where $i$ denotes an individual's index, $T_i$ is the time to event outcome for individual $i$ and $\bar{x}_i(s)$ are the covariates observed for individual $i$, including the observed history of dynamic covariates.

## Value

An object of class `LandmarkAnalysis`.

---

`fit_survival,LandmarkAnalysis-method`

*Fits the specified survival model at the landmark times and up to the horizon times specified by the user*

---

## Description

Fits the specified survival model at the landmark times and up to the horizon times specified by the user

## Usage

```
## S4 method for signature 'LandmarkAnalysis'
fit_survival(
  x,
  formula,
  landmarks,
  horizons,
  method,
  dynamic_covariates = c(),
  include_clusters = FALSE,
  censor_at_horizon = FALSE,
  validation_fold = 0
)
```

## Arguments

| | |
|---|---|
| `x` | An object of class `LandmarkAnalysis`. |
| `formula` | A formula to be used in survival sub-model fitting. |
| `landmarks` | Numeric vector of landmark times. |
| `horizons` | Vector of prediction horizons up to when the survival submodel is fitted. |
| `method` | Method for survival analysis, either "survfit" or "coxph". |

`dynamic_covariates`
: Vector of time-varying covariates to be used in the survival model.

`include_clusters`
: Boolean indicating whether to propagate cluster membership to survival analysis.

`censor_at_horizon`
: Boolean indicating whether to censor observations at horizon times

`validation_fold`
: If positive, cross-validation fold where model is fitted. If 0 (default), model fitting is performed on the complete dataset.

## Details

**Mathematical formulation:**

This function estimates the conditional probability of survival to horizon $s + w$, conditioned on having survived to the landmark time, $s$, that is

$$\pi_i(s + w|s) = P(T_i > s + w|T_i \geq s, \bar{x}_i(s)),$$

where $i$ denotes an individual's index, $T_i$ is the time to event outcome for individual $i$ and $\bar{x}_i(s)$ are the covariates observed for individual $i$, including the observed history of dynamic covariates.

## Value

An object of class `LandmarkAnalysis`.

---

`LandmarkAnalysis` *Creates an S4 class for a landmarking analysis*

---

## Description

Creates an S4 class for a landmarking analysis

## Usage

```
LandmarkAnalysis(
  data_static,
  data_dynamic,
  event_indicator,
  ids,
  event_time,
  times,
  measurements,
  censor_at_landmark = TRUE,
  K = 1
)
```

### Arguments

| | |
|---|---|
| `data_static` | A data frame containing subject ids, static covariates, |
| `data_dynamic` | Data frame in long format with subject ids, measurement values, measurement times and measurement name. |
| `event_indicator` | Name of the column indicating event or censoring. |
| `ids` | Name of the column indicating subject ids. |
| `event_time` | Name of the column indicating time of the event/censoring. |
| `times` | Name of the column indicating observation time in `data_dynamic`. |
| `measurements` | Name of the column indicating measurement values in `data_dynamic`. |
| `censor_at_landmark` | Boolean indicating whether to fit a single longitudinal model to the complete dataset (FALSE) or to censor observations at the landmark time prior to fitting the longitudinal model, iterating through landmark times (TRUE; default) |
| `K` | Number of cross-validation folds (by default, 1). |

### Value

An object of class `LandmarkAnalysis`

---

## `LandmarkAnalysis-class` *S4 class for performing a landmarking analysis*

---

### Description

S4 class for performing a landmarking analysis

### Slots

`landmarks` A numeric vector of landmark times.

`data_static` A data frame containing subject ids, static covariates,

`data_dynamic` Data frame in long format with subject ids, measurement values, measurement times and measurement name.

`event_indicator` Name of the column indicating event or censoring.

`ids` Name of the column indicating subject ids.

`times` Name of the column indicating observation time in `data_dynamic`.

`measurements` Name of the column indicating measurement values in `data_dynamic`.

`censor_at_landmark` Boolean indicating whether to fit a single longitudinal model to the complete dataset (FALSE) or to censor observations at the landmark time prior to fitting the longitudinal model, iterating through landmark times (TRUE; default)

`event_time` Name of the column indicating time of the event/censoring.

`risk_sets` List of indices.

`longitudinal_fits` List of model fits for the specified landmark times and biomarkers.

`longitudinal_predictions` List of model predictions for the specified landmark times and biomarkers.

`longitudinal_predictions_test` List of out-of-sample predictions for the specified landmark times and biomarkers.

`survival_datasets` List of survival dataframes used in the survival submodel.

`survival_datasets_test` List of survival dataframes used for out-of-sample predictions with the survival submodel.

`survival_fits` List of survival model fits at each of the specified landmark times.

`survival_predictions` List of time-to-event predictions for the specified landmark times and prediction horizons.

`survival_predictions_test` List of out-of-sample predictions for the time-to-event outcome if K > 1.

`K` Number of cross-validation folds (1 by default)

`cv_folds` Data frame associating individuals to cross-validation folds

---

`performance_metrics` *Performance metrics*

---

## Description

Computes concordance index (c-index) and Brier scores at the specified landmark times and prediction horizons.

## Usage

```
performance_metrics(
  x,
  landmarks,
  horizons,
  c_index = TRUE,
  brier = TRUE,
  auc_t = FALSE,
  train = TRUE,
  h_times = c()
)
```

## Arguments

| | |
|---|---|
| `x` | An object of class `LandmarkAnalysis`. |
| `landmarks` | A numeric vector of landmark times. |
| `horizons` | Vector of prediction horizons up to when the survival submodel is fitted. |

| | |
|---|---|
| c_index | A logical. If TRUE (default), C index is reported. |
| brier | A logical. If TRUE (default), Brier score is reported. |
| auc_t | A logical. If TRUE, AUC_t is reported. |
| train | A logical. If TRUE (default), performance metrics are computed in the training set. If FALSE, they are computed in the test set. |
| h_times | A numeric vector of prediction horizon times, specified relative to each landmark time, at which auc_t and Brier scores are calculated. |

## Value

Data frame with performance metrics across the specified landmark times and prediction horizons.

---

`performance_metrics,LandmarkAnalysis-method`
*Performance metrics*

---

## Description

Computes concordance index (c-index) and Brier scores at the specified landmark times and prediction horizons.

## Usage

```
## S4 method for signature 'LandmarkAnalysis'
performance_metrics(
  x,
  landmarks,
  horizons,
  c_index = TRUE,
  brier = TRUE,
  auc_t = FALSE,
  train = TRUE,
  h_times = c()
)
```

## Arguments

| | |
|---|---|
| x | An object of class LandmarkAnalysis. |
| landmarks | A numeric vector of landmark times. |
| horizons | Vector of prediction horizons up to when the survival submodel is fitted. |
| c_index | A logical. If TRUE (default), C index is reported. |
| brier | A logical. If TRUE (default), Brier score is reported. |
| auc_t | A logical. If TRUE, AUC_t is reported. |
| train | A logical. If TRUE (default), performance metrics are computed in the training set. If FALSE, they are computed in the test set. |
| h_times | A numeric vector of prediction horizon times, specified relative to each landmark time, at which auc_t and Brier scores are calculated. |

## Value

Data frame with performance metrics across the specified landmark times and prediction horizons.

---

`plot,LandmarkAnalysis-method`

*Plot longitudinal observations and predicted survival curve for one individual*

---

## Description

Produces a single-panel plot with a common time axis. The left of the landmark dashed line shows the individual's observed longitudinal measurements; the right shows their predicted survival curve. The summary value at the landmark that feeds into the survival sub-model is highlighted.

## Usage

```
## S4 method for signature 'LandmarkAnalysis'
plot(x, id, landmark, dynamic_covariate, horizon = NULL, train = TRUE, ...)
```

## Arguments

| | |
|---|---|
| `x` | An object of class `LandmarkAnalysis`. |
| `id` | Identifier of the individual to plot. Must match a value in the column `x@ids`. |
| `landmark` | Numeric landmark time. |
| `dynamic_covariate` | Character name of the dynamic covariate to display. |
| `horizon` | Numeric horizon time. If `NULL` (default), uses the single available horizon for `landmark`; errors when multiple horizons are available. |
| `train` | Logical. If `TRUE` (default), uses in-sample predictions. If `FALSE`, uses out-of-sample predictions (requires `validation_fold > 0` in `predict_survival`). |
| `...` | Additional arguments (not currently used). |

## Value

A `ggplot` object.

`predict_longitudinal` *Make predictions for time-varying covariates at specified landmark times*

## Description

Make predictions for time-varying covariates at specified landmark times

## Usage

```
predict_longitudinal(
  x,
  landmarks,
  method,
  dynamic_covariates,
  validation_fold = 0,
  ...
)
```

## Arguments

| | |
|---|---|
| `x` | An object of class `LandmarkAnalysis`. |
| `landmarks` | A numeric vector of landmark times. |
| `method` | Longitudinal data analysis method used to make predictions |
| `dynamic_covariates` | Vector of time-varying covariates to be modelled as the outcome of a longitudinal model. |
| `validation_fold` | If positive, cross-validation fold where model is fitted. If 0 (default), model fitting is performed in the complete dataset. |
| `...` | Additional arguments passed to the prediction function (e.g. number of classes/clusters for lcmm). |

## Value

An object of class `LandmarkAnalysis`.

`predict_longitudinal,LandmarkAnalysis-method` *Make predictions for time-varying covariates at specified landmark times*

## Description

Make predictions for time-varying covariates at specified landmark times

## Usage

```
## S4 method for signature 'LandmarkAnalysis'
predict_longitudinal(
  x,
  landmarks,
  method,
  dynamic_covariates,
  validation_fold = 0,
  ...
)
```

## Arguments

| | |
|---|---|
| `x` | An object of class `LandmarkAnalysis`. |
| `landmarks` | A numeric vector of landmark times. |
| `method` | Longitudinal data analysis method used to make predictions |
| `dynamic_covariates` | Vector of time-varying covariates to be modelled as the outcome of a longitudinal model. |
| `validation_fold` | If positive, cross-validation fold where model is fitted. If 0 (default), model fitting is performed in the complete dataset. |
| `...` | Additional arguments passed to the prediction function (e.g. number of classes/clusters for lcmm). |

## Value

An object of class `LandmarkAnalysis`.

predict_survival *Make predictions for time-to-event outcomes at specified horizon times*

## Description

Make predictions for time-to-event outcomes at specified horizon times

## Usage

```
predict_survival(
  x,
  landmarks,
  horizons,
  method = "survfit",
  dynamic_covariates = c(),
  include_clusters = FALSE,
  censor_at_horizon = FALSE,
  validation_fold = 0,
  ...
)
```

## Arguments

| | |
|---|---|
| x | An object of class LandmarkAnalysis. |
| landmarks | Numeric vector of landmark times. |
| horizons | Vector of prediction horizons up to when the survival submodel is fitted. |
| method | Name of the function that is used to make predictions. At the moment, 'survfit' is the only supported @method. |
| dynamic_covariates | Vector of time-varying covariates to be used in the survival model. |
| include_clusters | Boolean indicating whether to propagate cluster membership to survival analysis. |
| censor_at_horizon | Boolean indicating whether to censor observations at horizon times |
| validation_fold | If positive, cross-validation fold where model is fitted. If 0 (default), model fitting is performed on the complete dataset. |
| ... | Additional arguments passed to the prediction function (e.g. number of classes/clusters for lcmm). |

## Value

An object of class LandmarkAnalysis.

predict_survival,LandmarkAnalysis-method
*Make predictions for time-to-event outcomes at specified horizon times*

## Description

Make predictions for time-to-event outcomes at specified horizon times

## Usage

```
## S4 method for signature 'LandmarkAnalysis'
predict_survival(
  x,
  landmarks,
  horizons,
  method = "survfit",
  dynamic_covariates = c(),
  include_clusters = FALSE,
  censor_at_horizon = FALSE,
  validation_fold = 0,
  ...
)
```

## Arguments

| | |
|---|---|
| x | An object of class LandmarkAnalysis. |
| landmarks | Numeric vector of landmark times. |
| horizons | Vector of prediction horizons up to when the survival submodel is fitted. |
| method | Name of the function that is used to make predictions. At the moment, 'survfit' is the only supported @method. |
| dynamic_covariates | Vector of time-varying covariates to be used in the survival model. |
| include_clusters | Boolean indicating whether to propagate cluster membership to survival analysis. |
| censor_at_horizon | Boolean indicating whether to censor observations at horizon times |
| validation_fold | If positive, cross-validation fold where model is fitted. If 0 (default), model fitting is performed on the complete dataset. |
| ... | Additional arguments passed to the prediction function (e.g. number of classes/clusters for lcmm). |

## Value

An object of class LandmarkAnalysis.

---

`prune` *Prunes a landmark time from a* `LandmarkAnalysis`*, removing the risk set, longitudinal submodel and survival submodel from the object.*

---

## Description

Prunes a landmark time from a `LandmarkAnalysis`, removing the risk set, longitudinal submodel and survival submodel from the object.

## Usage

```
prune(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | An object of class `LandmarkAnalysis`. |
| `...` | Additional arguments |

## Value

An object of class `LandmarkAnalysis`.

---

`prune,LandmarkAnalysis-method`
*Prunes a landmark time from a* `LandmarkAnalysis`*, removing the risk set, longitudinal submodel and survival submodel from the object.*

---

## Description

Prunes a landmark time from a `LandmarkAnalysis`, removing the risk set, longitudinal submodel and survival submodel from the object.

## Usage

```
## S4 method for signature 'LandmarkAnalysis'
prune(x, landmark = NULL, ...)
```

## Arguments

| | |
|---|---|
| `x` | An object of class `LandmarkAnalysis`. |
| `landmark` | A numeric indicating the landmark time. |
| `...` | Additional arguments (not currently used) |

## Value

An object of class `LandmarkAnalysis`.

## prune_risk_sets *Prune a set of individuals from a risk set*

### Description

Prune a set of individuals from a risk set

### Usage

```
prune_risk_sets(x, landmark, individuals)
```

### Arguments

| | |
|---|---|
| x | An object of class LandmarkAnalysis. |
| landmark | a landmark time |
| individuals | Vector of individuals to be pruned from |

### Value

An object of class LandmarkAnalysis after having pruned the individuals indicated in individuals from the risk set at landmark time landmark.

## prune_risk_sets,LandmarkAnalysis-method *Prune a set of individuals from a risk set*

### Description

Prune a set of individuals from a risk set

### Usage

```
## S4 method for signature 'LandmarkAnalysis'
prune_risk_sets(x, landmark, individuals)
```

### Arguments

| | |
|---|---|
| x | An object of class LandmarkAnalysis. |
| landmark | a landmark time |
| individuals | Vector of individuals to be pruned from |

### Value

An object of class LandmarkAnalysis after having pruned the individuals indicated in individuals from the risk set at landmark time landmark.

show,LandmarkAnalysis-method
*Displays an object of class "LandmarkAnalysis"*

## Description

Displays an object of class "LandmarkAnalysis"

## Usage

```
## S4 method for signature 'LandmarkAnalysis'
show(object)
```

## Arguments

| | |
|---|---|
| object | An object of class LandmarkAnalysis. |

## Value

No return value(prints a summary to the console).

## Examples

```
data(epileptic)
epileptic_dfs <- split_wide_df(
  epileptic,
  ids = "id", times = "time",
  static = c("with.time", "with.status", "treat", "age", "gender", "learn.dis"),
  dynamic = c("dose"),
  measurement_name = "value"
)
x <- LandmarkAnalysis(
  data_static = epileptic_dfs$df_static,
  data_dynamic = epileptic_dfs$df_dynamic,
  event_indicator = "with.status",
  ids = "id", event_time = "with.time",
  times = "time", measurements = "value"
) |>
  compute_risk_sets(365.25)
show(x)
```

`split_wide_df` *Split a wide dataframe containing static and dynamic covariates and splits in into a dataframe with the static covariates and a list of dataframes, each associated to a dynamic covariate.*

## Description

Split a wide dataframe containing static and dynamic covariates and splits in into a dataframe with the static covariates and a list of dataframes, each associated to a dynamic covariate.

## Usage

```
split_wide_df(df, ids, times, static, dynamic, measurement_name)
```

## Arguments

| | |
|---|---|
| `df` | A dataframe in wide format. |
| `ids` | The name of the column that identifies individuals in `df`. |
| `times` | The name of the column that identifies measurement times in `df`. |
| `static` | A vector with the column names in `df` that store static covariates. |
| `dynamic` | A vector with the column names in `df` that store dynamic covariates. |
| `measurement_name` | The name for the columns where values of dynamic covariates will be stored. |

## Value

A data frame with the static covariates, and a list of data frames, one per dynamic covariate.

`summary,LandmarkAnalysis-method` *Summarise a LandmarkAnalysis object*

## Description

Summarise a LandmarkAnalysis object

## Usage

```
## S4 method for signature 'LandmarkAnalysis'
summary(
  object,
  type = c("longitudinal", "survival"),
  landmark,
  horizon = NULL,
  dynamic_covariate = NULL
)
```

## Arguments

| | |
|---|---|
| object | An object of class LandmarkAnalysis. |
| type | If longitudinal, it summarises the longitudinal submodel. If survival, it summarises the survival submodel. |
| landmark | A numeric indicating the landmark time. |
| horizon | For survival submodels, a numeric indicating the horizon time. |
| dynamic_covariate | For longitudinal submodels, a character indicating the dynamic covariate |

## Value

A summary of the desired submodel, printed to the console. Returns NULL invisibly.

## Examples

```
data(epileptic)
epileptic_dfs <- split_wide_df(
  epileptic,
  ids = "id", times = "time",
  static = c("with.time", "with.status", "treat", "age", "gender", "learn.dis"),
  dynamic = c("dose"),
  measurement_name = "value"
)
x <- LandmarkAnalysis(
  data_static = epileptic_dfs$df_static,
  data_dynamic = epileptic_dfs$df_dynamic,
  event_indicator = "with.status",
  ids = "id", event_time = "with.time",
  times = "time", measurements = "value"
) |>
  compute_risk_sets(365.25) |>
  fit_survival(
    formula = survival::Surv(event_time, event_status) ~ treat + age,
    landmarks = 365.25,
    horizons = 2 * 365.25,
    method = "coxph"
  )
summary(x, type = "survival", landmark = 365.25, horizon = 2 * 365.25)

# Full pipeline including longitudinal submodel
x2 <- LandmarkAnalysis(
  data_static = epileptic_dfs$df_static,
  data_dynamic = epileptic_dfs$df_dynamic,
  event_indicator = "with.status",
  ids = "id", event_time = "with.time",
  times = "time", measurements = "value"
) |>
  compute_risk_sets(365.25) |>
  fit_longitudinal(
    landmarks = 365.25,
```

```
    method = "lme4",
    formula = value ~ treat + age + gender + learn.dis + (time | id),
    dynamic_covariates = c("dose")
  ) |>
  predict_longitudinal(
    landmarks = 365.25,
    method = "lme4",
    allow.new.levels = TRUE,
    dynamic_covariates = c("dose")
  ) |>
  fit_survival(
    formula = survival::Surv(event_time, event_status) ~
      treat + age + gender + learn.dis + dose,
    landmarks = 365.25,
    horizons = 2 * 365.25,
    method = "coxph",
    dynamic_covariates = c("dose")
  ) |>
  predict_survival(landmarks = 365.25, horizons = 2 * 365.25)
summary(x2, type = "longitudinal", landmark = 365.25, dynamic_covariate = "dose")
summary(x2, type = "survival", landmark = 365.25, horizon = 2 * 365.25)
```

# Index