Package ‘elasdics’

December 9, 2022

Type Package

Title Elastic Analysis of Sparse, Dense and Irregular Curves

Version 1.1.1

Author Lisa Steyer <lisa.steyer@hu-berlin.de>

Maintainer Lisa Steyer <lisa.steyer@hu-berlin.de>

Description Provides functions to align curves and to compute mean curves based on the elastic distance defined in the square-root-velocity framework. For more details on this framework see Srivastava and Klassen (2016, <doi:10.1007/978-1-4939-4020-2>). For more theoretical details on our methods and algorithms see Steyer et al. (2021, <arXiv:2104.11039>).

License GPL-3

Encoding UTF-8

Imports splines, stats, numDeriv

RoxygenNote 7.2.2

Suggests testthat, covr

NeedsCompilation no

Repository CRAN

Date/Publication 2022-12-09 18:50:06 UTC

R topics documented:

`align_curves` *Align two curves measured at discrete points*

Description

Finds the optimal reparametrization of the second curve (stored in `data_curve2`) to the first one (stored in `data_curve1`) with respect to the elastic distance. Constructor function for class `aligned_curves`.

Usage

```
align_curves(data_curve1, data_curve2, closed = FALSE, eps = 0.01)
```

Arguments

`data_curve1`	`data.frame` with observed points in each row. Each variable is one coordinate direction. If there is a variable `t`, it is treated as the time parametrization, not as an additional coordinate.
`data_curve2`	same as `data_curve1`
`closed`	`TRUE` if the curves should be treated as closed.
`eps`	convergence tolerance

Value

an object of class `aligned_curves`, which is a `list` with entries

`data_curve1`	`data_curve1` with parametrization variable `t`
`data_curve2_aligned`	`data_curve2` with initial parametrization variable `t` and optimal parametrization `t_optim`
`elastic_dist`	elastic distance between curve1 and curve2
`closed`	`TRUE` if the curves should have been treated as closed.

Examples

```
#open curves
data_curve1 <- data.frame(x1 = c(1, 0.5, -1, -1), x2 = c(1, -0.5, -1, 1))
data_curve2 <- data.frame(x1 = c(0.1,0.7)*sin(1:6), x2 = cos(1:6))
aligned_curves <- align_curves(data_curve1, data_curve2)
plot(aligned_curves)

#different parametrization of the first curve
data_curve1$t <- 0:3/3
align_curves(data_curve1, data_curve2)

#closed curves
data_curve1 <- data.frame(x1 = sin(0:12/5), x2 = cos(0:12/5))
data_curve2 <- data.frame(x1 = c(1, 0.5, -1, -1), x2 = c(1, -0.5, -1, 1))
aligned_curves_closed <- align_curves(data_curve1, data_curve2, closed = TRUE)
plot(aligned_curves_closed, asp = 1)
```

---

center_curve — *Centers curves for plotting*

Description

Centers curves for plotting

Usage

```
center_curve(data_curve)
```

Arguments

data_curve	curve data

Value

a `data.frame` with evaluations of the curve centered at the origin

---

compute_elastic_mean — *Compute a elastic mean for a collection of curves*

Description

Computes a Fréchet mean for the curves stored in `data_curves`) with respect to the elastic distance. Constructor function for class `elastic_mean`.

Usage

```
compute_elastic_mean(
  data_curves,
  knots = seq(0, 1, len = 5),
  type = c("smooth", "polygon"),
  closed = FALSE,
  eps = 0.01,
  pen_factor = 100,
  max_iter = 50
)
```

Arguments

`data_curves`	list of `data.frames` with observed points in each row. Each variable is one coordinate direction. If there is a variable `t`, it is treated as the time parametrization, not as an additional coordinate.
`knots`	set of knots for the mean spline curve
`type`	if "smooth" linear srv-splines are used which results in a differentiable mean curve if "polygon" the mean will be piecewise linear.
`closed`	`TRUE` if the curves should be treated as closed.
`eps`	the algorithm stops if L2 norm of coefficients changes less
`pen_factor`	penalty factor forcing the mean to be closed
`max_iter`	maximal number of iterations

Value

an object of class `elastic_mean`, which is a `list` with entries

`type`	"smooth" if mean was modeled using linear srv-splines or "polygon" if constant srv-splines are used
`coefs`	spline coeffiecients
`knots`	spline knots
`data_curves`	list of `data.frames` with observed points in each row. First variable `t` gives the initial parametrization, second variable `t_optim` the optimal parametrization when the curve is aligned to the mean.
`closed`	`TRUE` if the mean is supposed to be a closed curve.

Examples

```
curve <- function(t){
  rbind(t*cos(13*t), t*sin(13*t))
}
set.seed(18)
data_curves <- lapply(1:4, function(i){
  m <- sample(10:15, 1)
  delta <- abs(rnorm(m, mean = 1, sd = 0.05))
```

```
  t <- cumsum(delta)/sum(delta)
  data.frame(t(curve(t)) + 0.07*t*matrix(cumsum(rnorm(2*length(delta))),
             ncol = 2))
})

#compute elastic means
knots <- seq(0,1, length = 11)
smooth_elastic_mean <- compute_elastic_mean(data_curves, knots = knots)
plot(smooth_elastic_mean)

knots <- seq(0,1, length = 15)
polygon_elastic_mean <- compute_elastic_mean(data_curves, knots = knots, type = "poly")
lines(get_evals(polygon_elastic_mean), col = "blue", lwd = 2)

#compute closed smooth mean, takes a little longer

knots <- seq(0,1, length = 11)
closed_elastic_mean <- compute_elastic_mean(data_curves, knots = knots, closed = TRUE)
plot(closed_elastic_mean)
```

`elasdics` *elasdics: **el**astic **a**nalysis of **s**parse, **d**ense and **i**rregular **c**urve**s**.*

Description

The elasdics package provides functions to align observed curves and to compute elastic means for collections of curves.

Main functions

Align two observed curves: `align_curves`
Compute a mean for a set of observed curves: `compute_elastic_mean`

`find_optimal_t` *Optimal alignment to a smooth curve*

Description

Finds optimal alignment for a discrete open srv curve to a smooth curve

Usage

```
find_optimal_t(srv_curve, s, q, initial_t = s, eps = 10 * .Machine$double.eps)
```

Arguments

`srv_curve`	srv transformation of the smooth curve, needs to be vectorized
`s`	time points for q, first has to be 0, last has to be 1
`q`	square root velocity vectors, one less than time points in s
`initial_t`	starting value for the optimization algorithm
`eps`	convergence tolerance

Value

optimal time points for q, without first value 0 and last value 1, optimal time points have the distance of the observation to the srv_curve as an attribute

---

find_optimal_t_discrete

Finds optimal alignment for discrete open curves

Description

Finds optimal aligned time points for srv curve q to srv curve p using coordinate wise optimization.

Usage

```
find_optimal_t_discrete(r, p, s, q, initial_t = s, eps = 10^-3)
```

Arguments

`r`	time points for p, first has to be 0, last has to be 1
`p`	square root velocity vectors, one less than time points in r
`s`	time points for q, first has to be 0, last has to be 1
`q`	square root velocity vectors, one less than time points in s
`initial_t`	starting value for the optimization algorithm
`eps`	convergence tolerance

Value

optimal time points for q, without first value 0 and last value 1 optimal time points have the distance of the observation to the srv_curve as an attribute

---

`find_optimal_t_discrete_closed`
Finds optimal alignment for discrete closed curves

Description

Finds optimal aligned time points for srv curve q to srv curve p using coordinate wise optimization.

Usage

```
find_optimal_t_discrete_closed(r, p, s, q, initial_t, eps = 10^-3)
```

Arguments

`r`	time points for p, first is last - 1
`p`	square root velocity vectors, one less than time points in r
`s`	time points for q, first is last - 1
`q`	square root velocity vectors, one less than time points in s
`initial_t`	starting value for the optimization algorithm
`eps`	convergence tolerance

Value

optimal time points for q, first is last -1

`fit_elastic_regression`
Compute a elastic mean for a collection of curves

Description

Computes a Fréchet mean for the curves stored in `data_curves` with respect to the elastic distance. Constructor function for class `elastic_reg_model`.

Usage

```
fit_elastic_regression(
  formula,
  data_curves,
  x_data,
  knots = seq(0, 1, 0.2),
  type = "smooth",
  closed = FALSE,
  max_iter = 10,
  eps = 0.001
)
```

Arguments

formula	an object of class "formula" of the form data_curves ~ ...".
data_curves	list of data.frames with observed points in each row. Each variable is one coordinate direction. If there is a variable t, it is treated as the time parametrization, not as an additional coordinate.
x_data	a data.frame with covariates.
knots	set of knots for the parameter curves of the regression model
type	if "smooth" linear srv-splines are used which results in a differentiable mean curve if "polygon" the mean will be piecewise linear.
closed	TRUE if the curves should be treated as closed.
max_iter	maximal number of iterations
eps	the algorithm stops if L2 norm of coefficients changes less

Value

an object of class elastic_reg_model, which is a list with entries

type	"smooth" if linear srv-splines or "polygon" if constant srv-splines were used
coefs	spline coeffiecients
knots	spline knots
data_curves	list of data.frames with observed points in each row. First variable t gives the initial parametrization, second variable t_optim the optimal parametrization when the curve is aligned to the model prediction.
closed	TRUE if the regression model fitted closed curves.

Examples

```
curve <- function(x_1, x_2, t){
  rbind(2*t*cos(6*t) - x_1*t , x_2*t*sin(6*t))
}
set.seed(18)
x_data <- data.frame(x_1 = runif(10,-1,1), x_2 = runif(10,-1,1))
data_curves <- apply(x_data, 1, function(x){
  m <- sample(10:15, 1)
  delta <- abs(rnorm(m, mean = 1, sd = 0.05))
  t <- cumsum(delta)/sum(delta)
  data.frame(t(curve((x[1] + 1), (x[2] + 2), t))
   + 0.07*t*matrix(cumsum(rnorm(2*length(delta))), ncol = 2))
})
reg_model <- fit_elastic_regression(data_curves ~ x_1 + x_2,
                                    data_curves = data_curves, x_data = x_data)
plot(reg_model)
```

fit_mean *Fitting function for open curves*

Description

Fits an elastic mean for open curves. Is usually called from `compute_elastic_mean`.

Usage

```
fit_mean(srv_data_curves, knots, max_iter, type, eps)
```

Arguments

`srv_data_curves`	list of `data.frames` with srv vectors in each row. Usually a result of a call to `get_srv_from_points`
`knots`	set of knots for the mean spline curve
`max_iter`	maximal number of iterations
`type`	if "smooth" linear srv-splines are used which results in a differentiable mean curve if "polygon" the mean will be piecewise linear.
`eps`	the algorithm stops if L2 norm of coefficients changes less

Value

a `list` with entries

`type`	"smooth" or "polygon"
`coefs`	`coefs` srv spline coefficients of the estimated mean
`knots`	spline knots
`t_optims`	optimal parametrization

fit_mean_closed *Fitting function for open curves*

Description

Fits an elastic mean for open curves. Is usually called from `compute_elastic_mean`.

Usage

```
fit_mean_closed(srv_data_curves, knots, max_iter, type, eps, pen_factor)
```

Arguments

srv_data_curves	list of data.frames with srv vectors in each row. Usually a result of a call to get_srv_from_points
knots	set of knots for the mean spline curve
max_iter	maximal number of iterations
type	if "smooth" linear srv-splines are used which results in a differentiable mean curve
eps	the algorithm stops if L2 norm of coefficients changes less
pen_factor	penalty factor forcing the mean to be closed if "polygon" the mean will be piecewise linear.

Value

a list with entries

type	"smooth" or "polygon"
coefs	coefs srv spline coefficients of the estimated mean
knots	spline knots
t_optims	optimal parametrization
shift_idxs	index of the starting point of the closed curve after alignment

---

get_evals *Evaluate a curve on a grid*

Description

Evaluate a curve on a grid

Usage

```
get_evals(curve, t_grid = NULL, ...)

## S3 method for class 'data.frame'
get_evals(curve, t_grid = NULL, ...)

## S3 method for class 'elastic_mean'
get_evals(curve, t_grid = NULL, centering = TRUE, ...)
```

Arguments

curve	a one parameter function which is to be evaluated on a grid
t_grid	the curve is evaluated at the values in t_grid, first value needs to be 0, last value needs to be 1. If t_grid = NULL, a default regular grid with grid length 0.01 is chosen
...	other arguments
centering	TRUE if curves shall be centered

Value

a data.frame with evaluations of the curve at the values in t_grid in its rows.

Examples

```
curve <- function(t){c(t*sin(10*t), t*cos(10*t))}
plot(get_evals(curve), type = "b")
```

---

get_srv_from_points *Helper functions for curve data measured at discrete points*

Description

Compute the square-root-velocity transformation or the parametrization with respect to arc length for a curve observed at discrete points.

Usage

```
get_srv_from_points(data_curve)

get_points_from_srv(srv_data)

get_arc_length_param(data_curve)
```

Arguments

data_curve	A data.frame with observed points on a curve. Each row is one point, each variable one coordinate direction. If there is a variable t, it is treated as the time parametrization, not as an additional coordinate.
srv_data	A data.frame with first column t corresponding to the parametrization and square-root-velocity vectors in the remaining columns.

Value

get_srv_from_points returns a data.frame with first column t corresponding to the parametrization and square-root-velocity vectors in the remaining columns. If no parametrization is given, the curve will be parametrized with respect to arc length. This parametrization will be computed by a call to get_arc_length_param as well.

Functions

- get_srv_from_points(): Compute square-root-velocity transformation for curve data measured at discrete points. The inverse transformation can be computed with get_points_from_s
- get_points_from_srv(): The inverse transformation to get_srv_from_points. Transforms square-root-velocity data to points representing a curve (with no parametrization).
- get_arc_length_param(): Compute arc length parametrization.

Examples

```
data_curve1 <- data.frame(x1 = 1:6*sin(1:6), x2 = cos(1:6))
get_arc_length_param(data_curve1) #same parametrization as in
get_srv_from_points(data_curve1)

data_curve2 <- data.frame(t = seq(0,1, length = 6), data_curve1)
plot(data_curve2[,2:3], type = "l", xlim = c(-6, 2), ylim = c(-2, 1))
srv_data <- get_srv_from_points(data_curve2)
#back transformed curve starts at (0,0)
lines(get_points_from_srv(srv_data), col = "red")
```

---

optimise_one_coord_analytic

Does optimization in one parameter direction

Description

Does optimization in one parameter direction

Usage

```
optimise_one_coord_analytic(t, i, r, p, s, q)
```

Arguments

t	current time points, first has to be 0, last has to be 1
i	index of t that should be updated
r	time points for p, first has to be 0, last has to be 1
p	square root velocity vectors, one less than time points in r
s	time points for q, first has to be 0, last has to be 1
q	square root velocity vectors, one less than time points in s

Value

optimal time points for q with respect to optimization only in the i-th coordinate direction

optimise_one_coord_analytic_closed *Does optimization in one parameter direction*

Description

Does optimization in one parameter direction

Usage

```
optimise_one_coord_analytic_closed(t, i, r, p, s, q)
```

Arguments

t	current time points, first has to be 0, last has to be 1
i	index of t that should be updated
r	time points for p, first is last - 1
p	square root velocity vectors, one less than time points in r
s	time points for q, first is last - 1
q	square root velocity vectors, one less than time points in s

Value

optimal time points for q with respect to optimization only in the i-th coordinate direction

plot.aligned_curves *Plot method for aligned curves*

Description

Plots objects of class `aligned_curves`. Points of same color correspond after the second curve is optimally aligned to the first curve.

Usage

```
## S3 method for class 'aligned_curves'
plot(x, points_col = rainbow, ...)
```

Arguments

x	object of class `aligned_curves`, usually a result of a call to `align_curves`
points_col	which color palette is used for points on the curves, default is rainbow, see `rainbow` for further options.
...	further plotting parameters.

Value

No value

See Also

For examples see documentation of `align_curves`.

`plot.elastic_mean` *Plot method for planar elastic mean curves*

Description

Plots objects of class `elastic_mean`.

Usage

```
## S3 method for class 'elastic_mean'
plot(x, asp = 1, col = "red", ...)
```

Arguments

`x`	object of class `elastic_mean`, usually a result of a call to `compute_elastic_mean`
`asp`	numeric, giving the aspect ratio of the two coordinates, see `plot.window` for details.
`col`	color of the mean curve.
`...`	further plotting parameters.

Value

No value

See Also

For examples see documentation of `compute_elastic_mean`.

`plot.elastic_reg_model` *Plot method for planar elastic regression models*

Description

Plots objects of class `elastic_reg_model`.

Usage

```
## S3 method for class 'elastic_reg_model'
plot(x, asp = 1, col = "red", ...)
```

Arguments

`x`	object of class `elastic_reg_model`, usually a result of a call to `fit_elastic_regression`
`asp`	numeric, giving the aspect ratio of the two coordinates, see `plot.window` for details.
`col`	color of the predicted curves.
`...`	further plotting parameters.

Value

No value

See Also

For examples see documentation of `fit_elastic_regression`.

`predict.elastic_reg_model` *Predict method for elastic regression models*

Description

predicted curves for elastic regression model objects.

Usage

```
## S3 method for class 'elastic_reg_model'
predict(object, newdata = NULL, t_grid = seq(0, 1, 0.01), ...)
```

Arguments

object	object of class elastic_reg_model, usually a result of a call to fit_elastic_regression
newdata	an optional data.frame in which to look for variables with which to predict. If not given, the fitted values are used.
t_grid	grid on which the predicted curves are evaluated.
...	further arguments passed to or from other methods.

Value

a list of data.frames with predicted curves

See Also

For examples see documentation of fit_elastic_regression.

---

project_curve_on_closed

Close open curve via projection on derivative level.

Description

Close open curve via projection on derivative level.

Usage

```
project_curve_on_closed(data_curve)
```

Arguments

data_curve	data.frame with values of the curve.

Value

a data.frame with closed curve.

`srvf_to_curve` *Re-transform srv curve back to curve*

Description

Re-transform srv curve back to curve

Usage

```
srvf_to_curve(t, srv_curve)
```

Arguments

`t`	time points at which the resulting curve shall be evaluated.
`srv_curve`	srv curve as a function of one parameter, needs to be vectorized.

Value

a `matrix` with curve evaluations at time points `t` in its columns, rows correspond to coordinate directions

Index