Package ‘fdasrvf’

October 13, 2022

Type Package

Title Elastic Functional Data Analysis

Version 1.9.8

Date 2022-06-04

Author J. Derek Tucker <jdtuck@sandia.gov>

Maintainer J. Derek Tucker <jdtuck@sandia.gov>

Description Performs alignment, PCA, and modeling of multidimensional and unidimensional functions using the square-root velocity framework (Srivastava et al., 2011 <arXiv:1103.3817> and Tucker et al., 2014 <DOI:10.1016/j.csda.2012.12.001>). This framework allows for elastic analysis of functional data through phase and amplitude separation.

License GPL-3

LazyData TRUE

SystemRequirements C++11

Imports Rcpp (>= 0.12.1), coda, foreach, mvtnorm, methods, matrixcalc, splines, parallel, fields, doParallel, viridisLite, tolerance, lpSolve, Matrix, testthat

Suggests interp, plot3D, plot3Drgl, rgl

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 3.5.0),

RoxygenNote 7.2.0

Encoding UTF-8

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-06-07 14:50:06 UTC

R topics documented:

`align_fPCA` *Group-wise function alignment and PCA Extractions*

Description

This function aligns a collection of functions while extracting principal components.

Usage

```
align_fPCA(
  f,
  time,
  num_comp = 3,
  showplot = T,
  smooth_data = FALSE,
  sparam = 25,
  parallel = FALSE,
  cores = 8,
  MaxItr = 51,
  lambda = 0
)
```

Arguments

f	matrix (N x M) of M functions with N samples
time	vector of size N describing the sample points
num_comp	number of principal components to extract (default = 3)
showplot	shows plots of functions (default = T)
smooth_data	smooth data using box filter (default = F)
sparam	number of times to apply box filter (default = 25)
parallel	enable parallel mode using foreach and doParallel package
cores	set number of cores to use with doParallel (default = 2)
MaxItr	maximum number of iterations
lambda	controls the elasticity (default = 0)

Value

Returns a list containing

f0	original functions
fn	aligned functions - matrix (N x M) of M functions with N samples
qn	aligned srvfs - similar structure to fn
q0	original srvfs - similar structure to fn
mqn	srvf mean - vector of length N
gam	warping functions - vector of length N
Dx	cost function
vfpca	list containing
q_pca	srvf principal directions
f_pca	f principal directions
latent	latent values
coef	coefficients
U	eigenvectors

References

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
## Not run:
data("simu_data")
out = align_fPCA(simu_data$f,simu_data$time)

## End(Not run)
```

`AmplitudeBoxplot` *Amplitude Boxplot*

Description

This function constructs the amplitude boxplot

Usage

```
AmplitudeBoxplot(warp_median, alpha = 0.05, ka = 1, showplot = TRUE)
```

Arguments

`warp_median`	fdawarp object from time_warping of aligned data using the median
`alpha`	quantile value (default=.05, i.e., 95%)
`ka`	scalar for outlier cutoff (default=1)
`showplot`	shows plots of functions (default = T)

Value

Returns a ampbox object containing

`median_y`	median function
`Q1`	First quartile
`Q3`	Second quartile
`Q1a`	First quantile based on alpha
`Q3a`	Second quantile based on alpha
`minn`	minimum extreme function
`maxx`	maximum extreme function
`outlier_index`	indexes of outlier functions
`fmedian`	median function

References

Xie, W., S. Kurtek, K. Bharath, and Y. Sun (2016). "A Geometric Approach to Visualization of Variability in Functional Data." Journal of the American Statistical Association in press: 1-34.

Examples

```
data("simu_warp_median")
out <- AmplitudeBoxplot(simu_warp_median, showplot=FALSE)
```

beta *MPEG7 Curve Dataset*

Description

Contains the MPEG7 curve data set which is 20 curves in 65 classes. The array is structured with dimension (2,100,65,20)

Usage

```
data("mpeg7")
```

Format

an array of shape (2,100,65,20)

bootTB *Tolerance Bound Calculation using Bootstrap Sampling*

Description

This function computes tolerance bounds for functional data containing phase and amplitude variation using bootstrap sampling

Usage

```
bootTB(f, time, a = 0.05, p = 0.99, B = 500, no = 5, parallel = T)
```

Arguments

`f`	matrix of functions
`time`	vector describing time sampling
`a`	confidence level of tolerance bound (default = 0.05)
`p`	coverage level of tolerance bound (default = 0.99)
`B`	number of bootstrap samples (default = 500)
`no`	number of principal components (default = 5)
`parallel`	enable parallel processing (default = T)

Value

Returns a list containing

`amp`	amplitude tolerance bounds
`ph`	phase tolerance bounds

References

J. D. Tucker, J. R. Lewis, C. King, and S. Kurtek, “A Geometric Approach for Computing Tolerance Bounds for Elastic Functional Data,” Journal of Applied Statistics, 10.1080/02664763.2019.1645818, 2019.

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Jung, S. L. a. S. (2016). "Combined Analysis of Amplitude and Phase Variations in Functional Data." arXiv:1603.01775 [stat.ME].

Examples

```
## Not run:
  data("simu_data")
  out1 = bootTB(simu_data$f,simu_data$time)

## End(Not run)
```

---

calc_shape_dist *Elastic Shape Distance*

Description

Calculate elastic shape distance between two curves beta1 and beta2

Usage

```
calc_shape_dist(beta1, beta2, mode = "O", scale = F)
```

Arguments

beta1	array describing curve1 (n,T)
beta2	array describing curve
mode	Open ("O") or Closed ("C") curves
scale	Include scale (default =F)

Value

Returns a list containing

d	geodesic distance
dx	phase distance

References

Srivastava, A., Klassen, E., Joshi, S., Jermyn, I., (2011). Shape analysis of elastic curves in euclidean spaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (7), 1415-1428.

Examples

```
data("mpeg7")
out = calc_shape_dist(beta[,,1,1],beta[,,1,4])
```

---

curve_geodesic *Form geodesic between two curves*

Description

Form geodesic between two curves using Elastic Method

Usage

```
curve_geodesic(beta1, beta2, k = 5)
```

Arguments

beta1	array describing curve 1 (n,T)
beta2	array describing curve 2 (n,T)
k	number of curves along geodesic (default 5)

Value

a list containing

geod	curves along geodesic (n,T,k)
geod_q	srvf's along geodesic

References

Srivastava, A., Klassen, E., Joshi, S., Jermyn, I., (2011). Shape analysis of elastic curves in euclidean spaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (7), 1415-1428.

Examples

```
data("mpeg7")
out = curve_geodesic(beta[,,1,1], beta[,,1,5])
```

curve_karcher_cov *Curve Karcher Covariance*

Description

Calculate Karcher Covariance of a set of curves

Usage

```
curve_karcher_cov(v, len = NA)
```

Arguments

v	array (n,T,N) for N number of shooting vectors
len	lengths of curves (default=NA)

Value

K covariance matrix

References

Srivastava, A., Klassen, E., Joshi, S., Jermyn, I., (2011). Shape analysis of elastic curves in euclidean spaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (7), 1415-1428.

Examples

```
data("mpeg7")
out = curve_karcher_mean(beta[,,1,1:2], maxit=2) # note: use more shapes, small for speed
K = curve_karcher_cov(out$v)
```

curve_karcher_mean *Karcher Mean of Curves*

Description

Calculates Karcher mean or median of a collection of curves using the elastic square-root velocity (srvf) framework.

Usage

```
curve_karcher_mean(
  beta,
  mode = "O",
  rotated = T,
  scale = F,
  maxit = 20,
  ms = "mean"
)
```

Arguments

`beta`	array (n,T,N) for N number of curves
`mode`	Open ("O") or Closed ("C") curves
`rotated`	Optimize over rotation (default = T)
`scale`	Include scale (default = F)
`maxit`	maximum number of iterations
`ms`	string defining whether the Karcher mean ("mean") or Karcher median ("median") is returned (default = "mean")

Value

Returns a list containing

`mu`	mean srvf
`beta`	centered data
`betamean`	mean or median curve
`type`	string indicating whether mean or median is returned
`v`	shooting vectors
`q`	array of srvfs
`gam`	array of warping functions
`cent`	centers of original curves
`len`	length of curves
`len_q`	length of srvfs
`mean_scale`	mean length
`mean_scale_q`	mean length srvf
`E`	energy
`qun`	cost function

References

Srivastava, A., Klassen, E., Joshi, S., Jermyn, I., (2011). Shape analysis of elastic curves in euclidean spaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (7), 1415-1428.

Examples

```
data("mpeg7")
out = curve_karcher_mean(beta[,,1,1:2],maxit=2)  # note: use more shapes, small for speed
```

`curve_pair_align` *Pairwise align two curves*

Description

This function aligns to curves using Elastic Framework

Usage

```
curve_pair_align(beta1, beta2)
```

Arguments

beta1	array describing curve 1 (n,T)
beta2	array describing curve 2 (n,T)

Value

a list containing

beta2n	aligned curve 2 to 1
q2n	aligned srvf 2 to 1
gam	warping function
q1	srvf of curve 1

References

Srivastava, A., Klassen, E., Joshi, S., Jermyn, I., (2011). Shape analysis of elastic curves in euclidean spaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (7), 1415-1428.

Examples

```
data("mpeg7")
out = curve_pair_align(beta[,,1,1], beta[,,1,5])
```

`curve_principal_directions` *Curve PCA*

Description

Calculate principal directions of a set of curves

Usage

```
curve_principal_directions(v, K, mu, len = NA, no = 3, N = 5, mode = "O")
```

Arguments

v	array (n,T,N1) of shooting vectors
K	array (n*T,n*T) covariance matrix
mu	array (n,T) of mean srvf
len	length of original curves (default NA)
no	number of components
N	number of samples on each side of mean
mode	Open ("O") or Closed ("C") curves

Value

Returns a list containing

s	singular values
U	singular vectors
coef	principal coefficients
pd	principal directions

References

Srivastava, A., Klassen, E., Joshi, S., Jermyn, I., (2011). Shape analysis of elastic curves in euclidean spaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (7), 1415-1428.

Examples

```
data("mpeg7")
out = curve_karcher_mean(beta[,,1,1:2], maxit=2) # note: use more shapes, small for speed
K = curve_karcher_cov(out$v)
out = curve_principal_directions(out$v, K, out$mu)
```

`curve_srvf_align` *Align Curves*

Description

Aligns a collection of curves using the elastic square-root velocity (srvf) framework.

Usage

```
curve_srvf_align(
  beta,
  mode = "O",
  rotated = T,
  scale = F,
  maxit = 20,
  ms = "mean"
)
```

Arguments

`beta`	array (n,T,N) for N number of curves
`mode`	Open ("O") or Closed ("C") curves
`rotated`	Optimize over rotation (default = T)
`scale`	Include scale (default = F)
`maxit`	maximum number of iterations
`ms`	string defining whether the Karcher mean ("mean") or Karcher median ("median") is returned (default = "mean")

Value

Returns a list containing

`betan`	aligned curves
`qn`	aligned srvfs
`betamean`	mean curve
`q_mu`	mean SRVFs

References

Srivastava, A., Klassen, E., Joshi, S., Jermyn, I., (2011). Shape analysis of elastic curves in euclidean spaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (7), 1415-1428.

Examples

```
data("mpeg7")
out = curve_srvf_align(beta[,,1,1:2],maxit=2) # note: use more shapes, small for speed
```

curve_to_q *Convert to SRVF space*

Description

This function converts curves to SRVF

Usage

```
curve_to_q(beta)
```

Arguments

beta	array describing curve (n,T)

Value

q array describing srvf

References

Srivastava, A., Klassen, E., Joshi, S., Jermyn, I., (2011). Shape analysis of elastic curves in euclidean spaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (7), 1415-1428.

Examples

```
data("mpeg7")
q = curve_to_q(beta[,,1,1])$q
```

elastic.depth *Calculates elastic depth*

Description

This functions calculates the elastic depth between set of functions

Usage

```
elastic.depth(f, time, lambda = 0, parallel = FALSE)
```

Arguments

f	matrix of N function of M time points (MxN)
time	sample points of functions
lambda	controls amount of warping (default = 0)
parallel	run computation in parallel (default = T)

Value

Returns a list containing

amp	amplitude depth
phase	phase depth

References

T. Harris, J. D. Tucker, B. Li, and L. Shand, "Elastic depths for detecting shape anomalies in functional data," Technometrics, 10.1080/00401706.2020.1811156, 2020.

Examples

```
data("simu_data")
depths <- elastic.depth(simu_data$f[,1:4],simu_data$time)
```

---

elastic.distance — *Calculates two elastic distance*

Description

This functions calculates the distances between functions, D_y and D_x, where function 1 is aligned to function 2

Usage

```
elastic.distance(f1, f2, time, lambda = 0)
```

Arguments

f1	sample function 1
f2	sample function 2
time	sample points of functions
lambda	controls amount of warping (default = 0)

Value

Returns a list containing

Dy	amplitude distance
Dx	phase distance

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("simu_data")
distances <- elastic.distance(simu_data$f[,1],simu_data$f[,2],simu_data$time)
```

---

elastic.logistic *Elastic Logistic Regression*

Description

This function identifies a logistic regression model with phase-variability using elastic methods

Usage

```
elastic.logistic(
  f,
  y,
  time,
  B = NULL,
  df = 20,
  max_itr = 20,
  smooth_data = FALSE,
  sparam = 25,
  parallel = FALSE,
  cores = 2
)
```

Arguments

f	matrix (N x M) of M functions with N samples
y	vector of size M labels (1/-1)
time	vector of size N describing the sample points
B	matrix defining basis functions (default = NULL)
df	scalar controlling degrees of freedom if B=NULL (default=20)
max_itr	scalar number of iterations (default=20)
smooth_data	smooth data using box filter (default = F)
sparam	number of times to apply box filter (default = 25)
parallel	enable parallel mode using foreach and doParallel package
cores	set number of cores to use with doParallel (default = 2)

Value

Returns a list containing

alpha	model intercept
beta	regressor function
fn	aligned functions - matrix (N x M) of M functions with N samples
qn	aligned srvfs - similar structure to fn
gamma	warping functions - similar structure to fn
q	original srvf - similar structure to fn
B	basis matrix
b	basis coefficients
Loss	logistic loss
type	model type ('logistic')

References

Tucker, J. D., Wu, W., Srivastava, A., Elastic Functional Logistic Regression with Application to Physiological Signal Classification, Electronic Journal of Statistics (2014), submitted.

---

elastic.lpcr.regression

Elastic logistic Principal Component Regression

Description

This function identifies a logistic regression model with phase-variability using elastic pca

Usage

```
elastic.lpcr.regression(
  f,
  y,
  time,
  pca.method = "combined",
  no = 5,
  smooth_data = FALSE,
  sparam = 25
)
```

Arguments

f	matrix (N x M) of M functions with N samples
y	vector of size M lables
time	vector of size N describing the sample points
pca.method	string specifying pca method (options = "combined", "vert", or "horiz", default = "combined")
no	scalar specify number of principal components (default=5)
smooth_data	smooth data using box filter (default = F)
sparam	number of times to apply box filter (default = 25)

Value

Returns a lpcr object containing

alpha	model intercept
b	regressor vector
y	label vector
warp_data	fdawarp object of aligned data
pca	pca object of principal components
Loss	logistic loss
pca.method	string specifying pca method used

References

J. D. Tucker, J. R. Lewis, and A. Srivastava, “Elastic Functional Principal Component Regression,” Statistical Analysis and Data Mining, 10.1002/sam.11399, 2018.

---

elastic.mlogistic *Elastic Multinomial Logistic Regression*

Description

This function identifies a multinomial logistic regression model with phase-variability using elastic methods

Usage

```
elastic.mlogistic(
  f,
  y,
  time,
  B = NULL,
  df = 20,
```

```
  max_itr = 20,
  smooth_data = FALSE,
  sparam = 25,
  parallel = FALSE,
  cores = 2
)
```

Arguments

f	matrix (N x M) of M functions with N samples
y	vector of size M labels 1,2,...,m for m classes
time	vector of size N describing the sample points
B	matrix defining basis functions (default = NULL)
df	scalar controlling degrees of freedom if B=NULL (default=20)
max_itr	scalar number of iterations (default=20)
smooth_data	smooth data using box filter (default = F)
sparam	number of times to apply box filter (default = 25)
parallel	enable parallel mode using foreach and doParallel package
cores	set number of cores to use with doParallel (default = 2)

Value

Returns a list containing

alpha	model intercept
beta	regressor function
fn	aligned functions - matrix (N x M) of M functions with N samples
qn	aligned srvfs - similar structure to fn
gamma	warping functions - similar structure to fn
q	original srvf - similar structure to fn
B	basis matrix
b	basis coefficients
Loss	logistic loss
type	model type ('mlogistic')

References

Tucker, J. D., Wu, W., Srivastava, A., Elastic Functional Logistic Regression with Application to Physiological Signal Classification, Electronic Journal of Statistics (2014), submitted.

elastic.mlpcr.regression

Elastic Multinomial logistic Principal Component Regression

Description

This function identifies a multinomial logistic regression model with phase-variability using elastic pca

Usage

```
elastic.mlpcr.regression(
  f,
  y,
  time,
  pca.method = "combined",
  no = 5,
  smooth_data = FALSE,
  sparam = 25
)
```

Arguments

f	matrix (N x M) of M functions with N samples
y	vector of size M labels
time	vector of size N describing the sample points
pca.method	string specifying pca method (options = "combined", "vert", or "horiz", default = "combined")
no	scalar specify number of principal components (default=5)
smooth_data	smooth data using box filter (default = F)
sparam	number of times to apply box filter (default = 25)

Value

Returns a mlpcr object containing

alpha	model intercept
b	regressor vector
y	label vector
Y	Coded labels
warp_data	fdawarp object of aligned data
pca	pca object of principal components
Loss	logistic loss
pca.method	string specifying pca method used

References

J. D. Tucker, J. R. Lewis, and A. Srivastava, "Elastic Functional Principal Component Regression," Statistical Analysis and Data Mining, 10.1002/sam.11399, 2018.

`elastic.pcr.regression` *Elastic Linear Principal Component Regression*

Description

This function identifies a regression model with phase-variability using elastic pca

Usage

```
elastic.pcr.regression(
  f,
  y,
  time,
  pca.method = "combined",
  no = 5,
  smooth_data = FALSE,
  sparam = 25,
  parallel = F,
  C = NULL
)
```

Arguments

f	matrix (N x M) of M functions with N samples
y	vector of size M responses
time	vector of size N describing the sample points
pca.method	string specifying pca method (options = "combined", "vert", or "horiz", default = "combined")
no	scalar specify number of principal components (default=5)
smooth_data	smooth data using box filter (default = F)
sparam	number of times to apply box filter (default = 25)
parallel	run in parallel (default = F)
C	scale balance parameter for combined method (default = NULL)

Value

Returns a pcr object containing

alpha	model intercept
b	regressor vector
y	response vector
warp_data	fdawarp object of aligned data
pca	pca object of principal components
SSE	sum of squared errors
pca.method	string specifying pca method used

References

J. D. Tucker, J. R. Lewis, and A. Srivastava, "Elastic Functional Principal Component Regression," Statistical Analysis and Data Mining, 10.1002/sam.11399, 2018.

---

elastic.prediction *Elastic Prediction from Regression Models*

Description

This function performs prediction from an elastic regression model with phase-variability

Usage

```
elastic.prediction(f, time, model, y = NULL, smooth_data = FALSE, sparam = 25)
```

Arguments

f	matrix (N x M) of M functions with N samples
time	vector of size N describing the sample points
model	list describing model from elastic regression methods
y	responses of test matrix f (default=NULL)
smooth_data	smooth data using box filter (default = F)
sparam	number of times to apply box filter (default = 25)

Value

Returns a list containing

y_pred	predicted values of f or probabilities depending on model
SSE	sum of squared errors if linear
y_labels	labels if logistic model
PC	probability of classification if logistic

References

Tucker, J. D., Wu, W., Srivastava, A., Elastic Functional Logistic Regression with Application to Physiological Signal Classification, Electronic Journal of Statistics (2014), submitted.

`elastic.regression` *Elastic Linear Regression*

Description

This function identifies a regression model with phase-variability using elastic methods

Usage

```
elastic.regression(
  f,
  y,
  time,
  B = NULL,
  lam = 0,
  df = 20,
  max_itr = 20,
  smooth_data = FALSE,
  sparam = 25,
  parallel = FALSE,
  cores = 2
)
```

Arguments

f	matrix (N x M) of M functions with N samples
y	vector of size M responses
time	vector of size N describing the sample points
B	matrix defining basis functions (default = NULL)
lam	scalar regularization parameter (default=0)
df	scalar controlling degrees of freedom if B=NULL (default=20)
max_itr	scalar number of iterations (default=20)
smooth_data	smooth data using box filter (default = F)
sparam	number of times to apply box filter (default = 25)
parallel	enable parallel mode using foreach and doParallel package
cores	set number of cores to use with doParallel (default = 2)

Value

Returns a list containing

alpha	model intercept
beta	regressor function
fn	aligned functions - matrix (N x M) of M functions with N samples
qn	aligned srvfs - similar structure to fn
gamma	warping functions - similar structure to fn
q	original srvf - similar structure to fn
B	basis matrix
b	basis coefficients
SSE	sum of squared errors
type	model type ('linear')

References

Tucker, J. D., Wu, W., Srivastava, A., Elastic Functional Logistic Regression with Application to Physiological Signal Classification, Electronic Journal of Statistics (2014), submitted.

---

fdasrvf *Elastic Functional Data Analysis*

Description

A library for functional data analysis using the square root velocity framework which performs pair-wise and group-wise alignment as well as modeling using functional component analysis

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

J. D. Tucker, W. Wu, and A. Srivastava, "Phase-Amplitude Separation of Proteomics Data Using Extended Fisher-Rao Metric," Electronic Journal of Statistics, Vol 8, no. 2. pp 1724-1733, 2014.

J. D. Tucker, W. Wu, and A. Srivastava, "Analysis of signals under compositional noise With applications to SONAR data," IEEE Journal of Oceanic Engineering, Vol 29, no. 2. pp 318-330, Apr 2014.

Tucker, J. D. 2014, Functional Component Analysis and Regression using Elastic Methods. Ph.D. Thesis, Florida State University.

Robinson, D. T. 2012, Function Data Analysis and Partial Shape Matching in the Square Root Velocity Framework. Ph.D. Thesis, Florida State University.

Huang, W. 2014, Optimization Algorithms on Riemannian Manifolds with Applications. Ph.D. Thesis, Florida State University.

Cheng, W., Dryden, I. L., and Huang, X. (2016). Bayesian registration of functions and curves. Bayesian Analysis, 11(2), 447-475.

Srivastava, A., Klassen, E., Joshi, S., Jermyn, I., (2011). Shape analysis of elastic curves in euclidean spaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (7), 1415-1428.

Cheng, W., Dryden, I. L., and Huang, X. (2016). Bayesian registration of functions and curves. Bayesian Analysis, 11(2), 447-475.

W. Xie, S. Kurtek, K. Bharath, and Y. Sun, A geometric approach to visualization of variability in functional data, Journal of American Statistical Association 112 (2017), pp. 979-993.

Lu, Y., R. Herbei, and S. Kurtek, 2017: Bayesian registration of functions with a Gaussian process prior. Journal of Computational and Graphical Statistics, 26, no. 4, 894–904.

Lee, S. and S. Jung, 2017: Combined analysis of amplitude and phase variations in functional data. arXiv:1603.01775 [stat.ME], 1–21.

J. D. Tucker, J. R. Lewis, and A. Srivastava, “Elastic Functional Principal Component Regression,” Statistical Analysis and Data Mining, vol. 12, no. 2, pp. 101-115, 2019.

J. D. Tucker, J. R. Lewis, C. King, and S. Kurtek, “A Geometric Approach for Computing Tolerance Bounds for Elastic Functional Data,” Journal of Applied Statistics, 10.1080/02664763.2019.1645818, 2019.

T. Harris, J. D. Tucker, B. Li, and L. Shand, "Elastic depths for detecting shape anomalies in functional data," Technometrics, 10.1080/00401706.2020.1811156, 2020.

`function_group_warp_bayes`

Bayesian Group Warping

Description

This function aligns a set of functions using Bayesian SRSF framework

Usage

```
function_group_warp_bayes(
  f,
  time,
  iter = 50000,
  powera = 1,
  times = 5,
  tau = ceiling(times * 0.04),
  gp = seq(dim(f)[2]),
  showplot = TRUE
)
```

Arguments

`f`	matrix (N x M) of M functions with N samples
`time`	sample points of functions
`iter`	number of iterations (default = 150000)
`powera`	Dirichlet prior parameter (default 1)
`times`	factor of length of subsample points to look at (default = 5)
`tau`	standard deviation of Normal prior for increment (default ceil(times*.4))
`gp`	number of colors in plots (default seq(dim(f)[2]))
`showplot`	shows plots of functions (default = T)

Value

Returns a list containing

`f0`	original functions
`f_q`	f aligned quotient space
`gam_q`	warping functions quotient space
`f_a`	f aligned ambient space
`gam_a`	warping ambient space
`qmn`	mean srsf

References

Cheng, W., Dryden, I. L., and Huang, X. (2016). Bayesian registration of functions and curves. Bayesian Analysis, 11(2), 447-475.

Examples

```
## Not run:
data("simu_data")
out = function_group_warp_bayes(simu_data$f, simu_data$time)

## End(Not run)
```

`function_mean_bayes` *Bayesian Karcher Mean Calculation*

Description

This function calculates karcher mean of functions using Bayesian method

Usage

```
function_mean_bayes(f, time, times = 5, group = 1:dim(f)[2], showplot = TRUE)
```

Arguments

`f`	matrix (N x M) of M functions with N samples
`time`	sample points of functions
`times`	factor of length of subsample points to look at (default = 5)
`group`	(default 1:dim(f)[2])
`showplot`	shows plots of functions (default = T)

Value

Returns a list containing

`distfamily`	dist matrix
`match.matrix`	matrix of warping functions
`position`	position
`mu_5`	function mean
`rtmatrix`	rtmatrix
`sumdist`	sumdist
`qt.fitted`	aligned srsf functions
`estimator`	estimator
`estimator2`	estimator2
`regfuncs`	registered functions

References

Cheng, W., Dryden, I. L., and Huang, X. (2016). Bayesian registration of functions and curves. Bayesian Analysis, 11(2), 447-475.

Examples

```
## Not run:
data("simu_data")
out = function_mean_bayes(simu_data$f, simu_data$time)

## End(Not run)
```

`f_to_srvf` *Convert to SRSF*

Description

This function converts functions to srvf

Usage

```
f_to_srvf(f, time)
```

Arguments

`f`	matrix of functions
`time`	time

Value

q matrix of SRVFs

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("simu_data")
q <- f_to_srvf(simu_data$f,simu_data$time)
```

`gauss_model` *Gaussian model of functional data*

Description

This function models the functional data using a Gaussian model extracted from the principal components of the srvfs

Usage

```
gauss_model(warp_data, n = 1, sort_samples = FALSE)
```

Arguments

`warp_data`	fdawarp object from time_warping of aligned data
`n`	number of random samples (n = 1)
`sort_samples`	sort samples (default = F)

Value

Returns a fdawarp object containing

`fs`	random aligned samples
`gams`	random warping function samples
`ft`	random function samples

References

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("simu_warp")
out1 = gauss_model(simu_warp,n = 10)
```

`gradient` *Gradient using finite differences*

Description

This function takes the gradient of f using finite differences

Usage

```
gradient(f, binsize)
```

Arguments

`f`	vector with N samples
`binsize`	scalar of time samples

Value

g vector with N samples which is the gradient of f

Examples

```
data("simu_data")
out = gradient(simu_data$f[,1],mean(diff(simu_data$time)))
```

growth_vel *Berkley Growth Velocity Dataset*

Description

Combination of both boys and girls growth velocity from the Berkley Dataset

Usage

```
data("growth_vel")
```

Format

A list which contains f and time

horizFPCA *Horizontal Functional Principal Component Analysis*

Description

This function calculates vertical functional principal component analysis on aligned data

Usage

```
horizFPCA(warp_data, no, ci = c(-1, 0, 1), showplot = TRUE)
```

Arguments

warp_data	fdawarp object from time_warping of aligned data
no	number of principal components to extract
ci	geodesic standard deviations (default = c(-1,0,1))
showplot	show plots of principal directions (default = T)

Value

Returns a hfpca object containing

gam_pca	warping functions principal directions
psi_pca	srvf principal directions
latent	latent values
U	eigenvectors
vec	shooting vectors
mu	Karcher Mean

References

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("simu_warp")
hfpca = horizFPCA(simu_warp,no = 3)
```

`im` *Example Image Data set*

Description

Contains two simulated images for registration

Usage

```
data("image")
```

Format

a list containing two images of dimension (64,64)

`invertGamma` *Invert Warping Function*

Description

This function calculates the inverse of gamma

Usage

```
invertGamma(gam)
```

Arguments

`gam`	vector of N samples

Value

Returns gamI inverted vector

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("simu_warp")
out = invertGamma(simu_warp$gam[,1])
```

---

jointFPCA — *Joint Vertical and Horizontal Functional Principal Component Analysis*

Description

This function calculates amplitude and phase joint functional principal component analysis on aligned data

Usage

```
jointFPCA(
  warp_data,
  no,
  id = round(length(warp_data$time)/2),
  C = NULL,
  ci = c(-1, 0, 1),
  showplot = T
)
```

Arguments

`warp_data`	fdawarp object from time_warping of aligned data
`no`	number of principal components to extract
`id`	integration point for f0 (default = midpoint)
`C`	balance value (default = NULL)
`ci`	geodesic standard deviations (default = c(-1,0,1))
`showplot`	show plots of principal directions (default = T)

Value

Returns a list containing

q_pca	srvf principal directions
f_pca	f principal directions
latent	latent values
coef	coefficients
U	eigenvectors
mu_psi	mean psi function
mu_g	mean g function
id	point use for f(0)
C	optimized phase amplitude ratio

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Jung, S. L. a. S. (2016). "Combined Analysis of Amplitude and Phase Variations in Functional Data." arXiv:1603.01775 [stat.ME].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("simu_warp")
data("simu_data")
jfpca = jointFPCA(simu_warp, no = 3)
```

---

joint_gauss_model *Gaussian model of functional data using joint Model*

Description

This function models the functional data using a Gaussian model extracted from the principal components of the srvfs using the joint model

Usage

```
joint_gauss_model(warp_data, n = 1, no = 5)
```

Arguments

warp_data	fdawarp object from time_warping of aligned data
n	number of random samples (n = 1)
no	number of principal components (n=4)

Value

Returns a fdawarp object containing

fs	random aligned samples
gams	random warping function samples
ft	random function samples
qs	random srvf samples

References

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Jung, S. L. a. S. (2016). "Combined Analysis of Amplitude and Phase Variations in Functional Data." arXiv:1603.01775 [stat.ME].

Examples

```
data("simu_warp")
out1 = joint_gauss_model(simu_warp,n = 10)
```

---

kmeans_align *K-Means Clustering and Alignment*

Description

This function clusters functions and aligns using the elastic square-root slope (srsf) framework.

Usage

```
kmeans_align(
  f,
  time,
  K,
  seeds = NULL,
  nonempty = 0,
  lambda = 0,
  showplot = TRUE,
  smooth_data = FALSE,
  sparam = 25,
  parallel = FALSE,
  alignment = TRUE,
  omethod = "DP",
  MaxItr = 50,
  thresh = 0.01
)
```

Arguments

f	matrix (N x M) of M functions with N samples
time	vector of size N describing the sample points
K	number of clusters
seeds	indexes of cluster center functions (default = NULL)
nonempty	minimum number of functions per cluster in assignment step of k-means. Set it as a positive integer to avoid the problem of empty clusters (default = 0)
lambda	controls the elasticity (default = 0)
showplot	shows plots of functions (default = T)
smooth_data	smooth data using box filter (default = F)
sparam	number of times to apply box filter (default = 25)
parallel	enable parallel mode using foreach and doParallel package (default=F)
alignment	whether to perform alignment (default = T)
omethod	optimization method (DP,DP2,RBFGS)
MaxItr	maximum number of iterations
thresh	cost function threshold

Value

Returns a fdakma object containing

f0	original functions
fn	aligned functions - matrix (N x M) of M functions with N samples which is a list for each cluster
qn	aligned SRSFs - similar structure to fn
q0	original SRSFs
labels	cluster labels
templates	cluster center functions
templates.q	cluster center SRSFs
gam	warping functions - similar structure to fn
qun	Cost Function Value

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Sangalli, L. M., et al. (2010). "k-mean alignment for curve clustering." Computational Statistics & Data Analysis 54(5): 1219-1233.

Examples

```
## Not run:
data("growth_vel")
out <- kmeans_align(growth_vel$f,growth_vel$time, K=2)

## End(Not run)
```

---

multiple_align_functions

Group-wise function alignment to specified mean

Description

This function aligns a collection of functions using the elastic square-root slope (srsf) framework.

Usage

```
multiple_align_functions(
  f,
  time,
  mu,
  lambda = 0,
  showplot = TRUE,
  smooth_data = FALSE,
  sparam = 25,
  parallel = FALSE,
  omethod = "DP",
  MaxItr = 20,
  iter = 2000
)
```

Arguments

f	matrix (N x M) of M functions with N samples
time	vector of size N describing the sample points
mu	vector of size N that f is aligned to
lambda	controls the elasticity (default = 0)
showplot	shows plots of functions (default = T)
smooth_data	smooth data using box filter (default = F)
sparam	number of times to apply box filter (default = 25)
parallel	enable parallel mode using foreach and doParallel package (default=F)
omethod	optimization method (DP,DP2,RBFGS,dBayes,expBayes)
MaxItr	maximum number of iterations
iter	bayesian number of mcmc samples (default 2000)

Value

Returns a fdawarp object containing

f0	original functions
fn	aligned functions - matrix (N x M) of M functions with N samples
qn	aligned SRSFs - similar structure to fn
q0	original SRSF - similar structure to fn
fmean	function mean or median - vector of length N
mqn	SRSF mean or median - vector of length N
gam	warping functions - similar structure to fn
orig.var	Original Variance of Functions
amp.var	Amplitude Variance
phase.var	Phase Variance
qun	Cost Function Value

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

optimum.reparam *Align two functions*

Description

This function aligns two SRSF functions using Dynamic Programming

Usage

```
optimum.reparam(
  Q1,
  T1,
  Q2,
  T2,
  lambda = 0,
  method = "DP",
  w = 0.01,
  f1o = 0,
  f2o = 0
)
```

Arguments

Q1	srsf of function 1
T1	sample points of function 1
Q2	srsf of function 2
T2	sample points of function 2
lambda	controls amount of warping (default = 0)
method	controls which optimization method (default="DP") options are Dynamic Programming ("DP"), Coordinate Descent ("DP2"), and Riemannian BFGS ("RBFGS")
w	controls LRBFGS (default = 0.01)
f1o	initial value of f1, vector or scalar depending on q1, defaults to zero
f2o	initial value of f2, vector or scalar depending on q1, defaults to zero

Value

gam warping function

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("simu_data")
q = f_to_srvf(simu_data$f,simu_data$time)
gam = optimum.reparam(q[,1],simu_data$time,q[,2],simu_data$time)
```

---

outlier.detection *Outlier Detection*

Description

This function calculates outlier's using geodesic distances of the SRVFs from the median

Usage

```
outlier.detection(q, time, mq, k = 1.5)
```

Arguments

q	matrix (N x M) of M SRVF functions with N samples
time	vector of size N describing the sample points
mq	median calculated using time_warping
k	cutoff threshold (default = 1.5)

Value

`q_outlier`	outlier functions

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("toy_data")
data("toy_warp")
q_outlier = outlier.detection(toy_warp$q0,toy_data$time,toy_warp$mqn,k=.1)
```

`pair_align_functions` *Align two functions*

Description

This function aligns two functions using SRSF framework. It will align f2 to f1

Usage

```
pair_align_functions(
  f1,
  f2,
  time,
  lambda = 0,
  method = "DP",
  w = 0.01,
  iter = 2000
)
```

Arguments

`f1`	function 1
`f2`	function 2
`time`	sample points of functions
`lambda`	controls amount of warping (default = 0)
`method`	controls which optimization method (default="DP") options are Dynamic Programming ("DP"), Coordinate Descent ("DP2"), Riemannian BFGS ("RBFGS"), Simultaneous Alignment ("SIMUL"), Dirichlet Bayesian ("dBayes"), and Expo-Map Bayesian ("expBayes")
`w`	controls LRBFGS (default = 0.01)
`iter`	number of mcmc iterations for mcmc method (default 2000)

Value

Returns a list containing

`f2tilde` aligned f2

`gam` warping function

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Cheng, W., Dryden, I. L., and Huang, X. (2016). Bayesian registration of functions and curves. Bayesian Analysis, 11(2), 447-475.

Lu, Y., Herbei, R., and Kurtek, S. (2017). Bayesian registration of functions with a Gaussian process prior. Journal of Computational and Graphical Statistics, DOI: 10.1080/10618600.2017.1336444.

Examples

```
data("simu_data")
out = pair_align_functions(simu_data$f[,1],simu_data$f[,2],simu_data$time)
```

---

pair_align_functions_bayes

Align two functions

Description

This function aligns two functions using Bayesian SRSF framework. It will align f2 to f1

Usage

```
pair_align_functions_bayes(
  f1,
  f2,
  timet,
  iter = 15000,
  times = 5,
  tau = ceiling(times * 0.4),
  powera = 1,
  showplot = TRUE,
  extrainfo = FALSE
)
```

Arguments

`f1`	function 1
`f2`	function 2
`timet`	sample points of functions
`iter`	number of iterations (default = 15000)
`times`	factor of length of subsample points to look at (default = 5)
`tau`	standard deviation of Normal prior for increment (default ceil(times*.4))
`powera`	Dirichlet prior parameter (default 1)
`showplot`	shows plots of functions (default = T)
`extrainfo`	T/F whether additional information is returned

Value

Returns a list containing

`f1`	function 1
`f2_q`	registered function using quotient space
`gam_q`	warping function quotient space
`f2_a`	registered function using ambient space
`q2_a`	warping function ambient space
`match_collect`	posterior samples from warping function (returned if extrainfo=TRUE)
`dist_collect`	posterior samples from the distances (returned if extrainfo=TRUE)
`kappa_collect`	posterior samples from kappa (returned if extrainfo=TRUE)
`log_collect`	log-likelihood of each sample (returned if extrainfo=TRUE)
`pct_accept`	vector of acceptance ratios for the warping function (returned if extrainfo=TRUE)

References

Cheng, W., Dryden, I. L., and Huang, X. (2016). Bayesian registration of functions and curves. Bayesian Analysis, 11(2), 447-475.

Examples

```
data("simu_data")
out = pair_align_functions_bayes(simu_data$f[,1], simu_data$f[,2], simu_data$time)
```

pair_align_functions_expomap

Align two functions using geometric properties of warping functions

Description

This function aligns two functions using Bayesian framework. It will align f2 to f1. It is based on mapping warping functions to a hypersphere, and a subsequent exponential mapping to a tangent space. In the tangent space, the Z-mixture pCN algorithm is used to explore both local and global structure in the posterior distribution.

Usage

```
pair_align_functions_expomap(
  f1,
  f2,
  timet,
  iter = 20000,
  burnin = min(5000, iter/2),
  alpha0 = 0.1,
  beta0 = 0.1,
  zpcn = list(betas = c(0.5, 0.05, 0.005, 1e-04), probs = c(0.1, 0.1, 0.7, 0.1)),
  propvar = 1,
  init.coef = rep(0, 2 * 10),
  npoints = 200,
  extrainfo = FALSE
)
```

Arguments

`f1`	observed data, numeric vector
`f2`	observed data, numeric vector
`timet`	sample points of functions
`iter`	length of the chain
`burnin`	number of burnin MCMC iterations
`alpha0, beta0`	IG parameters for the prior of sigma1
`zpcn`	list of mixture coefficients and prior probabilities for Z-mixture pCN algorithm of the form list(betas, probs), where betas and probs are numeric vectors of equal length
`propvar`	variance of proposal distribution
`init.coef`	initial coefficients of warping function in exponential map; length must be even
`npoints`	number of sample points to use during alignment
`extrainfo`	T/F whether additional information is returned

Details

The Z-mixture pCN algorithm uses a mixture distribution for the proposal distribution, controlled by input parameter zpcn. The zpcn$betas must be between 0 and 1, and are the coefficients of the mixture components, with larger coefficients corresponding to larger shifts in parameter space. The zpcn$probs give the probability of each shift size.

Value

Returns a list containing

`f2_warped`	f2 aligned to f1
`gamma`	Posterior mean gamma function
`g.coef`	matrix with iter columns, posterior draws of g.coef
`psi`	Posterior mean psi function
`sigma1`	numeric vector of length iter, posterior draws of sigma1
`accept`	Boolean acceptance for each sample (if extrainfo=TRUE)
`betas.ind`	Index of zpcn mixture component for each sample (if extrainfo=TRUE)
`logl`	numeric vector of length iter, posterior loglikelihood (if extrainfo=TRUE)
`gamma_mat`	Matrix of all posterior draws of gamma (if extrainfo=TRUE)
`gamma_q025`	Lower 0.025 quantile of gamma (if extrainfo=TRUE)
`gamma_q975`	Upper 0.975 quantile of gamma (if extrainfo=TRUE)
`sigma_eff_size`	Effective sample size of sigma (if extrainfo=TRUE)
`psi_eff_size`	Vector of effective sample sizes of psi (if extrainfo=TRUE)
`xdist`	Vector of posterior draws from xdist between registered functions (if extrainfo=TRUE)
`ydist`	Vector of posterior draws from ydist between registered functions (if extrainfo=TRUE)

References

Lu, Y., Herbei, R., and Kurtek, S. (2017). Bayesian registration of functions with a Gaussian process prior. Journal of Computational and Graphical Statistics, DOI: 10.1080/10618600.2017.1336444.

Examples

```
## Not run:
# This is a mcmc algorithm and takes a long time to run
data("simu_data")
myzpcn <- list(betas = c(0.1, 0.01, 0.005, 0.0001),
  probs = c(0.2, 0.2, 0.4, 0.2))
out = pair_align_functions_expomap(simu_data$f[,1], simu_data$f[,2],
  timet = simu_data$time, zpcn = myzpcn, extrainfo = TRUE)
# overall acceptance ratio
mean(out$accept)
# acceptance ratio by zpcn coefficient
with(out, tapply(accept, myzpcn$betas[betas.ind], mean))
## End(Not run)
```

`pair_align_image` *Pairwise align two images This function aligns to images using the q-map framework*

Description

Pairwise align two images This function aligns to images using the q-map framework

Usage

```
pair_align_image(
  I1,
  I2,
  M = 5,
  ortho = TRUE,
  basis_type = "t",
  resizei = FALSE,
  N = 64,
  stepsize = 1e-05,
  itermax = 1000
)
```

Arguments

`I1`	reference image
`I2`	image to warp
`M`	number of basis elements (default=5)
`ortho`	orthonormalize basis (default=TRUE)
`basis_type`	("t","s","i","o"; default="t")
`resizei`	resize image (default=TRUE)
`N`	size of resized image (default=64)
`stepsize`	gradient stepsize (default=1e-5)
`itermax`	maximum number of iterations (default=1000)

Value

Returns a list containing

`Inew`	aligned I2
`gam`	warping function

References

Q. Xie, S. Kurtek, E. Klassen, G. E. Christensen and A. Srivastava. Metric-based pairwise and multiple image registration. IEEE European Conference on Computer Vision (ECCV), September, 2014

Examples

```
## Not run:
# This is a gradient descent algorithm and takes a long time to run
data("image")
out <- pair_align_image(im$I1, im$I2)
## End(Not run)
```

---

pcaTB *Tolerance Bound Calculation using Elastic Functional PCA*

Description

This function computes tolerance bounds for functional data containing phase and amplitude variation using principal component analysis

Usage

```
pcaTB(f, time, m = 4, B = 1e+05, a = 0.05, p = 0.99)
```

Arguments

f	matrix of functions
time	vector describing time sampling
m	number of principal components (default = 4)
B	number of monte carlo iterations
a	confidence level of tolerance bound (default = 0.05)
p	coverage level of tolerance bound (default = 0.99)

Value

Returns a list containing

pca	pca output
tol	tolerance factor

References

J. D. Tucker, J. R. Lewis, C. King, and S. Kurtek, "A Geometric Approach for Computing Tolerance Bounds for Elastic Functional Data," Journal of Applied Statistics, 10.1080/02664763.2019.1645818, 2019.

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Jung, S. L. a. S. (2016). "Combined Analysis of Amplitude and Phase Variations in Functional Data." arXiv:1603.01775 [stat.ME].

Examples

```
## Not run:
  data("simu_data")
  out1 = pcaTB(simu_data$f,simu_data$time)

## End(Not run)
```

PhaseBoxplot *Phase Boxplot*

Description

This function constructs the amplitude boxplot

Usage

```
PhaseBoxplot(warp_median, alpha = 0.05, kp = 1, showplot = TRUE)
```

Arguments

warp_median	fdawarp object from time_warping of aligned data using the median
alpha	quantile value (default=.05, i.e., 95%)
kp	scalar for outlier cutoff (default=1)
showplot	shows plots of functions (default = T)

Value

Returns a phbox object containing

median_x	median warping function
Q1	First quartile
Q3	Second quartile
Q1a	First quantile based on alpha
Q3a	Second quantile based on alpha
minn	minimum extreme function
maxx	maximum extreme function
outlier_index	indexes of outlier functions

References

Xie, W., S. Kurtek, K. Bharath, and Y. Sun (2016). "A Geometric Approach to Visualization of Variability in Functional Data." Journal of the American Statistical Association in press: 1-34.

Examples

```
data("simu_warp_median")
out <- PhaseBoxplot(simu_warp_median, showplot=FALSE)
```

predict.lpcr — *Elastic Prediction for functional logistic PCR Model*

Description

This function performs prediction from an elastic logistic fPCR regression model with phase-variability

Usage

```
## S3 method for class 'lpcr'
predict(object, newdata = NULL, y = NULL, ...)
```

Arguments

`object`	Object of class inheriting from "elastic.pcr.regression"
`newdata`	An optional matrix in which to look for variables with which to predict. If omitted, the fitted values are used.
`y`	An optional vector of labels to calculate PC. If omitted, PC is NULL
`...`	additional arguments affecting the predictions produced

Value

Returns a list containing

`y_pred`	predicted probabilities of the class of newdata
`y_labels`	class labels of newdata
`PC`	probability of classification

References

J. D. Tucker, J. R. Lewis, and A. Srivastava, “Elastic Functional Principal Component Regression,” Statistical Analysis and Data Mining, 10.1002/sam.11399, 2018.

predict.mlpcr — *Elastic Prediction for functional multinomial logistic PCR Model*

Description

This function performs prediction from an elastic multinomial logistic fPCR regression model with phase-variability

Usage

```
## S3 method for class 'mlpcr'
predict(object, newdata = NULL, y = NULL, ...)
```

Arguments

object	Object of class inheriting from "elastic.pcr.regression"
newdata	An optional matrix in which to look for variables with which to predict. If omitted, the fitted values are used.
y	An optional vector of labels to calculate PC. If omitted, PC is NULL
...	additional arguments affecting the predictions produced

Value

Returns a list containing

y_pred	predicted probabilities of the class of newdata
y_labels	class labels of newdata
PC	probability of classification per class
PC.comb	total probability of classification

References

J. D. Tucker, J. R. Lewis, and A. Srivastava, “Elastic Functional Principal Component Regression,” Statistical Analysis and Data Mining, 10.1002/sam.11399, 2018.

---

predict.pcr — *Elastic Prediction for functional PCR Model*

Description

This function performs prediction from an elastic pcr regression model with phase-variability

Usage

```
## S3 method for class 'pcr'
predict(object, newdata = NULL, y = NULL, ...)
```

Arguments

object	Object of class inheriting from "elastic.pcr.regression"
newdata	An optional matrix in which to look for variables with which to predict. If omitted, the fitted values are used.
y	An optional vector of responses to calculate SSE. If omitted, SSE is NULL
...	additional arguments affecting the predictions produced

Value

Returns a list containing

y_pred	predicted values of newdata
SSE	sum of squared errors

References

J. D. Tucker, J. R. Lewis, and A. Srivastava, "Elastic Functional Principal Component Regression," Statistical Analysis and Data Mining, 10.1002/sam.11399, 2018.

---

q_to_curve — *Convert to curve space*

Description

This function converts SRVFs to curves

Usage

```
q_to_curve(q, scale = 1)
```

Arguments

q	array describing SRVF (n,T)
scale	scale of original beta (default 1)

Value

beta array describing curve

References

Srivastava, A., Klassen, E., Joshi, S., Jermyn, I., (2011). Shape analysis of elastic curves in euclidean spaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (7), 1415-1428.

Examples

```
data("mpeg7")
q = curve_to_q(beta[,,1,1])$q
beta1 = q_to_curve(q)
```

`reparam_curve` *Align two curves*

Description

This function aligns two SRVF functions using Dynamic Programming

Usage

```
reparam_curve(
  beta1,
  beta2,
  lambda = 0,
  method = "DP",
  w = 0.01,
  rotated = T,
  isclosed = F,
  mode = "O"
)
```

Arguments

`beta1`	array defining curve 1
`beta2`	array defining curve 1
`lambda`	controls amount of warping (default = 0)
`method`	controls which optimization method (default="DP") options are Dynamic Programming ("DP"), Coordinate Descent ("DP2"), Riemannian BFGS ("RBFGS")
`w`	controls LRBFGS (default = 0.01)
`rotated`	boolean if rotation is desired
`isclosed`	boolean if curve is closed
`mode`	Open ("O") or Closed ("C") curves

Value

return a List containing

`gam`	warping function
`R`	rotation matrix
`tau`	seed point

References

Srivastava, A., Klassen, E., Joshi, S., Jermyn, I., (2011). Shape analysis of elastic curves in euclidean spaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (7), 1415-1428.

Examples

```
data("mpeg7")
gam = reparam_curve(beta[,,1,1],beta[,,1,5])$gam
```

`reparam_image` *Find optimum reparameterization between two images*

Description

Finds the optimal warping function between two images using the elastic framework

Usage

```
reparam_image(It, Im, gam, b, stepsize = 1e-05, itermax = 1000, lmark = FALSE)
```

Arguments

`It`	template image matrix
`Im`	test image matrix
`gam`	initial warping array
`b`	basis matrix
`stepsize`	gradient stepsize (default=1e-5)
`itermax`	maximum number of iterations (default=1000)
`lmark`	use landmarks (default=FALSE)

Value

Returns a list containing

`gamnew`	final warping
`Inew`	aligned image
`H`	energy
`stepsize`	final stepsize

References

Q. Xie, S. Kurtek, E. Klassen, G. E. Christensen and A. Srivastava. Metric-based pairwise and multiple image registration. IEEE European Conference on Computer Vision (ECCV), September, 2014

resamplecurve *Resample Curve*

Description

This function resamples a curve to a number of points

Usage

```
resamplecurve(x, N = 100, mode = "O")
```

Arguments

x	matrix defining curve (n,T)
N	Number of samples to re-sample curve, N usually is > T
mode	Open ("O") or Closed ("C") curves

Value

xn matrix defining resampled curve

References

Srivastava, A., Klassen, E., Joshi, S., Jermyn, I., (2011). Shape analysis of elastic curves in euclidean spaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (7), 1415-1428.

Examples

```
data("mpeg7")
xn = resamplecurve(beta[,,1,1],200)
```

rgam *Random Warping*

Description

Generates random warping functions

Usage

```
rgam(N, sigma, num)
```

Arguments

`N`	length of warping function
`sigma`	variance of warping functions
`num`	number of warping functions

Value

gam warping functions

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
gam = rgam(N=101, sigma=.01, num=35)
```

---

sample_shapes *Sample shapes from model*

Description

Sample shapes from model

Usage

```
sample_shapes(mu, K, mode = "O", no = 3, numSamp = 10)
```

Arguments

`mu`	array (n,T) of mean srvf
`K`	array (2*T,2*T) covariance matrix
`mode`	Open ("O") or Closed ("C") curves
`no`	number of principal components
`numSamp`	number of samples

Value

samples list of sample curves

References

Srivastava, A., Klassen, E., Joshi, S., Jermyn, I., (2011). Shape analysis of elastic curves in euclidean spaces. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (7), 1415-1428.

Examples

```
data("mpeg7")
out = curve_karcher_mean(beta[,,1,1:2], maxit=2) # note: use more shapes, small for speed
K = curve_karcher_cov(out$v)
samples = sample_shapes(out$mu, K)
```

---

simu_data *Simulated two Gaussian Dataset*

Description

A functional dataset where the individual functions are given by: $y_i(t) = z_{i,1}e^{-(t-1.5)^2/2} + z_{i,2}e^{-(t+1.5)^2/2}$, $t \in [-3,3]$, $i = 1,2,\ldots,21$, where $z_{i,1}$ and $z_{i,2}$ are *i.i.d.* normal with mean one and standard deviation 0.25. Each of these functions is then warped according to: $\gamma_i(t) = 6(\frac{e^{a_i(t+3)/6}-1}{e^{a_i}-1}) - 3$ if $a_i \neq 0$, otherwise $\gamma_i = \gamma_{id}$ ($gamma_{id}(t) = t$) is the identity warping). The variables are as follows: f containing the 21 functions of 101 samples and time which describes the sampling

Usage

```
data("simu_data")
```

Format

A list which contains f and time

---

simu_warp *Aligned Simulated two Gaussian Dataset*

Description

A functional dataset where the individual functions are given by: $y_i(t) = z_{i,1}e^{-(t-1.5)^2/2} + z_{i,2}e^{-(t+1.5)^2/2}$, $t \in [-3,3]$, $i = 1,2,\ldots,21$, where $z_{i,1}$ and $z_{i,2}$ are *i.i.d.* normal with mean one and standard deviation 0.25. Each of these functions is then warped according to: $\gamma_i(t) = 6(\frac{e^{a_i(t+3)/6}-1}{e^{a_i}-1}) - 3$ if $a_i \neq 0$, otherwise $\gamma_i = \gamma_{id}$ ($gamma_{id}(t) = t$) is the identity warping). The variables are as follows: f containing the 21 functions of 101 samples and time which describes the sampling which has been aligned

Usage

```
data("simu_warp")
```

Format

A list which contains the outputs of the time_warping function

---

`simu_warp_median` *Aligned Simulated two Gaussian Dataset using Median*

Description

A functional dataset where the individual functions are given by: $y_i(t) = z_{i,1} e^{-(t-1.5)^2/2} + z_{i,2} e^{-(t+1.5)^2/2}$, $t \in [-3, 3]$, $i = 1, 2, \ldots, 21$, where $z_{i,1}$ and $z_{i,2}$ are *i.i.d.* normal with mean one and standard deviation 0.25. Each of these functions is then warped according to: $\gamma_i(t) = 6(\frac{e^{a_i(t+3)/6}-1}{e^{a_i}-1}) - 3$ if $a_i \neq 0$, otherwise $\gamma_i = \gamma_{id}$ ($gamma_{id}(t) = t$) is the identity warping). The variables are as follows: f containing the 21 functions of 101 samples and time which describes the sampling which has been aligned

Usage

```
data("simu_warp_median")
```

Format

A list which contains the outputs of the time_warping function finding the median

---

`smooth.data` *Smooth Functions*

Description

This function smooths functions using standard box filter

Usage

```
smooth.data(f, sparam)
```

Arguments

`f`	matrix (N x M) of M functions with N samples
`sparam`	number of times to run box filter

Value

fo smoothed functions

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("simu_data")
fo = smooth.data(simu_data$f,25)
```

SqrtMean *SRVF transform of warping functions*

Description

This function calculates the srvf of warping functions with corresponding shooting vectors and finds the mean

Usage

```
SqrtMean(gam)
```

Arguments

gam	matrix (N x M) of M warping functions with N samples

Value

Returns a list containing

mu	Karcher mean psi function
gam_mu	Karcher mean warping function
psi	srvf of warping functions
vec	shooting vectors

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("simu_warp")
out = SqrtMean(simu_warp$gam)
```

SqrtMedian *SRVF transform of warping functions*

Description

This function calculates the srvf of warping functions with corresponding shooting vectors and finds the median

Usage

```
SqrtMedian(gam)
```

Arguments

gam	matrix (N x M) of M warping functions with N samples

Value

Returns a list containing

median	Karcher median psi function
gam_median	Karcher mean warping function
psi	srvf of warping functions
vec	shooting vectors

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("simu_warp_median")
out = SqrtMedian(simu_warp_median$gam)
```

srvf_to_f *Convert SRVF to f*

Description

This function converts SRVFs to functions

Usage

```
srvf_to_f(q, time, f0 = 0)
```

Arguments

q	matrix of srvf
time	time
f0	initial value of f

Value

f matrix of functions

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("simu_data")
q = f_to_srvf(simu_data$f,simu_data$time)
f = srvf_to_f(q,simu_data$time,simu_data$f[1,])
```

time_warping *Group-wise function alignment*

Description

This function aligns a collection of functions using the elastic square-root slope (srsf) framework.

Usage

```
time_warping(
  f,
  time,
  lambda = 0,
  method = "mean",
  center = TRUE,
  showplot = TRUE,
  smooth_data = FALSE,
  sparam = 25,
  parallel = FALSE,
  omethod = "DP",
  MaxItr = 20
)
```

Arguments

`f`	matrix (N x M) of M functions with N samples
`time`	vector of size N describing the sample points
`lambda`	controls the elasticity (default = 0)
`method`	warp and calculate to Karcher Mean or Median (options = "mean" or "median", default = "mean")
`center`	center warping functions (default = T)
`showplot`	shows plots of functions (default = T)
`smooth_data`	smooth data using box filter (default = F)
`sparam`	number of times to apply box filter (default = 25)
`parallel`	enable parallel mode using `foreach` and `doParallel` package (default=F)
`omethod`	optimization method (DP,DP2,RBFGS)
`MaxItr`	maximum number of iterations

Value

Returns a fdawarp object containing

`f0`	original functions
`fn`	aligned functions - matrix (N x M) of M functions with N samples
`qn`	aligned SRSFs - similar structure to fn
`q0`	original SRSF - similar structure to fn
`fmean`	function mean or median - vector of length N
`mqn`	SRSF mean or median - vector of length N
`gam`	warping functions - similar structure to fn
`orig.var`	Original Variance of Functions
`amp.var`	Amplitude Variance
`phase.var`	Phase Variance
`qun`	Cost Function Value

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
## Not run:
data("simu_data")
out = time_warping(simu_data$f,simu_data$time)

## End(Not run)
```

`toy_data` *Distributed Gaussian Peak Dataset*

Description

A functional dataset where the individual functions are given by a Gaussian peak with locations along the x-axis. The variables are as follows: f containing the 29 functions of 101 samples and time which describes the sampling

Usage

```
data("toy_data")
```

Format

A list which contains f and time

`toy_warp` *Aligned Distributed Gaussian Peak Dataset*

Description

A functional dataset where the individual functions are given by a Gaussian peak with locations along the x-axis. The variables are as follows: f containing the 29 functions of 101 samples and time which describes the sampling which as been aligned

Usage

```
data("toy_warp")
```

Format

A list which contains the outputs of the time_warping function

vertFPCA — *Vertical Functional Principal Component Analysis*

Description

This function calculates vertical functional principal component analysis on aligned data

Usage

```
vertFPCA(
  warp_data,
  no,
  id = round(length(warp_data$time)/2),
  ci = c(-1, 0, 1),
  showplot = TRUE
)
```

Arguments

warp_data	fdawarp object from time_warping of aligned data
no	number of principal components to extract
id	point to use for f(0) (default = midpoint)
ci	geodesic standard deviations (default = c(-1,0,1))
showplot	show plots of principal directions (default = T)

Value

Returns a vfpca object containing

q_pca	srvf principal directions
f_pca	f principal directions
latent	latent values
coef	coefficients
U	eigenvectors
id	point used for f(0)

References

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("simu_warp")
vfpca = vertFPCA(simu_warp,no = 3)
```

warp_f_gamma *Warp Function*

Description

This function warps function f by γ

Usage

```
warp_f_gamma(f, time, gamma, spl.int = FALSE)
```

Arguments

`f`	vector function
`time`	time
`gamma`	vector warping function
`spl.int`	use spline interpolation (default F)

Value

fnew warped function

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("simu_data")
fnew = warp_f_gamma(simu_data$f[,1],simu_data$time,seq(0,1,length.out=101))
```

warp_q_gamma *Warp SRSF*

Description

This function warps srsf q by γ

Usage

```
warp_q_gamma(q, time, gamma, spl.int = FALSE)
```

Arguments

`q`	vector
`time`	time
`gamma`	vector warping function
`spl.int`	use spline interpolation (default F)

Value

qnew warped function

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., Marron, J. S., May 2011. Registration of functional data using fisher-rao metric, arXiv:1103.3817v2 [math.ST].

Tucker, J. D., Wu, W., Srivastava, A., Generative Models for Function Data using Phase and Amplitude Separation, Computational Statistics and Data Analysis (2012), 10.1016/j.csda.2012.12.001.

Examples

```
data("simu_data")
q = f_to_srvf(simu_data$f,simu_data$time)
qnew = warp_q_gamma(q[,1],simu_data$time,seq(0,1,length.out=101))
```

Index