Package ‘RvtkStatismo’

July 1, 2014

Type Package

Title Integrates statismo and R using the vtkStandardMeshRepresenter

Version 0.2.140701

Date 2014-07-01

Author Stefan Schlager, the authors of Statismo

Maintainer Stefan Schlager <`zarquon42@gmail.com`>

Description Integrates statismo and R using the vtkStandardMeshRepresenter.
Statismo shape models will be stored as objects of class ``pPCA''. (this is work in progress).

License GPL >=2

Imports Rcpp (>= 0.11.1),Morpho,Rvcg,methods

LinkingTo Rcpp,RcppEigen

SystemRequirement VTK5.8, statismo (>= 0.9 best ist freshly from github)

URL `http://github.com/zarquon42b/RvtkStatismo`, URL:
`http://github.com/statismo/statismo`

R topics documented:

`RvtkStatismo-package` *Integrates statismo and R using the vtkStandardMeshRepresente*

Description

Integrates statismo and R using the vtkStandardMeshRepresenter. Statismo shape models will be stored as objects of class "pPCA". (this is work in progress).

Details

Package:	RvtkStatismo
Type:	Package
Version:	0.2.140701
Date:	2014-07-01
License:	GPL
LazyLoad:	yes

Author(s)

Stefan Schlager

Maintainer: Stefan Schlager <zarquon42@gmail.com>

References

To be announced

align2domain *align a sample to a model*

Description

align a sample to a model

Usage

```
align2domain(model, sample, scale = TRUE, ptDomain = NULL,
  ptSample = NULL)
```

Arguments

`model`	statistical model of class "pPCA"
`sample`	matrix or mesh3d
`scale`	logical: request scaling during alignment
`ptDomain`	integer vector: specifies the indices of the domain points that are to be used for registration (order is important).
`ptSample`	integer vector: specifies the indices of the sample that are to be used for registration (order is important).

Value

a rotated (and scaled) mesh or matrix - depending on the input.

ComputeConstrainedModel *Constrains a model of class pPCA by a subset of coordinates*

Description

Constrains a model of class pPCA by a subset of coordinates

Usage

```
ComputeConstrainedModel(x, model, align = FALSE, use.lm, deselect = FALSE,
  origSpace = FALSE)
```

Arguments

`x`	a k x 3 matrix containing the coordinates of the reduces model
`model`	an object of class `pPCA`
`align`	logical: if TRUE, `x` will be aligned to the models mean
`use.lm`	integer vector, specifying which coordinates from the full model are to be used/missing (see note)
`deselect`	logical: if TRUE, use.lm specifies the missing coordinates instead of those present.
`origSpace`	logical: if align=TRUE and origSpace=TRUE, the representer of the returned model will contain the estimated full shape in the original coordinate system of `x`

Value

an object of class pPCA constrained to `x`

Note

if deselect = F, the order of the entries in `use.lm` is important: the i-th entry in use.lm specifies the index of the meanshapes coordinate belonging to the i-th coordinate of `x`.

Examples

```
## create a model superimposed with missing landmarks 3 and 4
newmod <- pPCA(boneLM[,,-1],sigma=0,scale=TRUE,use.lm = 3:4,deselect=TRUE)
## predict the left out shape from the constrained model
boneLM1 <- ComputeConstrainedModel(boneLM[-c(3:4),,1],newmod,align=TRUE,use.lm=3:4,deselect=T,origSpace=TRUE)
## the coordinates of the estimated complete config are now stored in the representer's vertices
## Not run:
##visualize prediction error
deformGrid3d(vert2points(boneLM1$representer),boneLM[,,1],ngrid=0)

## End(Not run)
```

`getCoordVar` *get per coordinate variance from a statistical model*

Description

get per coordinate variance from a statistical model

Usage

```
getCoordVar(model)
```

Arguments

`model`	object of class pPCA

Note

calculates the per-coordinate variance as described in Luethi(2009)

References

Lüthi M, Albrecht T, Vetter T. 2009. Probabilistic modeling and visualization of the flexibility in morphable models. In: Mathematics of Surfaces XIII. Springer. p 251-264

`getDataLikelihood` *calculate probability/coefficients for a matrix/mesh given a statistical model*

Description

calculate probability for a matrix/mesh given a statistical model

Usage

```
getDataLikelihood(x, model, align = FALSE, use.lm)

## S3 method for class 'matrix'
getDataLikelihood(x, model, align = FALSE, use.lm = NULL)

## S3 method for class 'mesh3d'
getDataLikelihood(x, model, align = FALSE, use.lm = NULL)

getCoefficients(x, model, align = TRUE, use.lm = NULL)
```

Arguments

`x`	matrix or mesh3d
`model`	a model of class pPCA
`align`	logical: if TRUE the data will be aligned to the model's mean
`use.lm`	integer vector specifying row indices of the coordinates to use for rigid registration on the model's meanshape.

Details

`getDataLikelihood` estimates the likelihood of a dataset for belonging to the model by exploiting the χ^2-distribution of the (squared) Mahalanobisdistance, which, in turn, is simply the squared norm of the sample's coefficients in the latent space.

Value

`getDataLikelihood` returns a probability, while `getCoefficients` returns the (scaled) scores in the pPCA space.

`mesh2vtp` *exports a triangular mesh of class mesh3d to a vtp file*

Description

exports a triangular mesh of class mesh3d to a vtp file

Usage

```
mesh2vtp(mesh, filename = dataname)
```

Arguments

`mesh`	mesh of class mesh3d
`filename`	character

`meshalign` *align meshes stored in a list by their vertices*

Description

align meshes stored in a list by their vertices

Usage

```
meshalign(meshlist, scale = TRUE, use.lm = NULL, deselect = FALSE,
  array = FALSE)
```

Arguments

`meshlist`	list containing triangular meshes of class "mesh3d"
`scale`	logical: request scaling during alignment
`deselect`	logical: if TRUE, missingIndex references the existing coordinates instead of the missing ones.
`use.lm`	integer vector: specifies the indices of the points that are to be used in the constrained model
`array`	logical: if TRUE the superimposed vertices will be returned as 3D array.

Value

returns a list of aligned meshes or an array of dimensions k x 3 x n, where k=number of vertices and n=sample size.

`meshlist2array`	*convert meshes to array consisting of vertex coordinates*

Description

convert meshes to array consisting of vertex coordinates

Usage

```
meshlist2array(meshlist)
```

Arguments

`meshlist`	list containing triangular meshes of class "mesh3d"

Value

returns an array with k x 3 x n dimensions where k=number of vertices, and n=sample size.

`pPCA`	*calculate or modify a probablistic PCA based on 3D-coordinates*

Description

calculate or modify a probablistic PCA based on 3D-coordinates

Usage

```
pPCA(array, align = TRUE, use.lm = NULL, deselect = FALSE, sigma = NULL,
  exVar = 1, scale = TRUE, representer = NULL)

UpdateModel(model, sigma = NULL, exVar = 1)
```

Arguments

`array`	array of dimensions k x 3 x n, where k=number of coordinates and n=sample size.
`align`	logical: if TRUE, the data will be aligned first
`use.lm`	integer vector: specifies the indices of the points that are to be used in the constrained model
`deselect`	logical: if TRUE, `use.lm` references the missing coordinates instead of the present ones.
`sigma`	estimate of error variance (sensible is a value estimating coordinate error in terms of observer error)

`exVar`	numeric value with `0 < exVar <= 1` specifying the PCs to be included by their cumulative explained Variance
`scale`	logical: allow scaling in Procrustes fitting
`fullfit`	logical: if FALSE only the non-missing points will be used for registration.
`representer`	a triangular mesh, where the vertices correspond to the coordinates in `array`, leave NULL for pointclouds.
`model`	object of class `pPCA`

Value

returns a probabilistic PCA model as S4 class "pPCA" (see `pPCA-class`). `UpdateModel` is used to modify existing models by changing sigma and exVar.

References

Lüthi M, Albrecht T, Vetter T. 2009. Probabilistic modeling and visualization of the flexibility in morphable models. In: Mathematics of Surfaces XIII. Springer. p 251-264

Examples

```
require(Morpho)
data(boneData)
model <- pPCA(boneLM[,,])
## change parameters without recomputing Procrustes fit
model1 <- UpdateModel(model, sigma=1, exVar=0.8)
```

`pPCA-class` *Documentation of class pPCA*

Description

Documentation of class pPCA

Details

The class contains the the follwing slots (still not yet set in stone)

PCA a list containing

- `sdev`: the square roots of the covariance matrix' eigenvalues
- `rotation`: matrix containing the orthonormal PCBasis vectos
- `x`: the scores within the latent space(scaled by 1/sdev)
- `center`: a vector of the mean shape in with coordinates ordered `(x1,y1,z1, x2, y2,z2, ..., xn,yn,zn)`

scale logical: indicating if the data was aligned including scaling

representer an object of class mesh3d or a list with entry `vb` being a matrix with the columns containing coordinates and `it` a 0x0 matrix

sigma the noise estimation of the data

Variance a data.frame containing the Variance, cumulative Variance and Variance explained by each Principal component

rawdata optional data: a matrix with rows containing the mean centred coordinates in order `(x1,y1,z1, x2, y2,z2, ..., xn,yn,zn)`

`PredictSample` *predict or restrict a mesh or matrix based on a statistical model*

Description

predict or restrict a mesh or matrix based on a statistical model

Usage

```
PredictSample(model, dataset, representer = TRUE, ...)

## S4 method for signature 'pPCA,matrix,ANY'
PredictSample(model, dataset, representer = TRUE,
  origSpace = TRUE, use.lm = NULL, deselect = FALSE, sdmax,
  mahaprob = c("none", "chisq", "dist"), align = TRUE, ...)

## S4 method for signature 'pPCA,mesh3d,logical'
PredictSample(model, dataset,
  representer = TRUE, origSpace = TRUE, use.lm = NULL, deselect = FALSE,
  sdmax, mahaprob = c("none", "chisq", "dist"), align = TRUE, ...)
```

Arguments

`model`	model of class `pPCA`
`dataset`	a matrix or a mesh3d
`representer`	if TRUE and the model contains a representer mesh, a surface mesh will be returned, coordinate matrix otherwise.
`origSpace`	logical: rotate the estimation back into the original coordinate system.
`pPCA`	logical: if TRUE, a constrained pPCA model is returned. "chisq" uses the Chi-Square distribution of the squared Mahalanobisdistance, while "dist" restricts the values to be within a multi-dimensional sphere of radius `sdmax`. If FALSE the probability will be determined per PC separately.
`use.lm`	optional: integer vector specifying row indices of the coordinates to use for rigid registration on the model's meanshape.
`sdmax`	maximum allowed standard deviation (per Principal axis) within the model space. Defines the probabilistic boundaries.
`mahaprob`	character: if != "none", use mahalanobis-distance to determine overall probability (of the shape projected into the model space.

Value

`PredictSample` returns a matrix/mesh3d restricted to the boundaries given by the modelspace.

See Also

`StatismoModelMembers`

`read.vtk` *imports vtk and vtp files*

Description

imports vtk and vtp files

Usage

```
read.vtk(filename)
```

Arguments

`filename`	character string

Value

list of class mesh3d

`representer2sample` *get the representer from a model of class "pPCA"*

Description

get the representer from a model of class "pPCA"

Usage

```
representer2sample(model)
```

Arguments

`model`	object of class `pPCA`

Value

an object of class mesh3d or matrix, depending whether a point cloud or a triangular mesh is the model's representer.

`rigidAlign` *Fast Procrustes align of coordinates*

Description

Fast Procrustes align of coordinates

Usage

```
rigidAlign(array, scale = TRUE, use.lm = NULL, deselect = FALSE)
```

Arguments

`array`	array of coordinates
`scale`	logical: request scaling during alignment
`use.lm`	integer vector: specifies the indices of the points that are to be used in the constrained model
`deselect`	logical: if TRUE, `use.lm` references the missing coordinates instead of the present ones.

Value

a list containing

`rotated`	array containing registered coordinates
`mshape`	matrix containing meanshape

`statismoBuildModel` *generate a statistical model using an array of superimposed landmarks or a list of meshes*

Description

generate a statistical model using an array of superimposed landmarks

Usage

```
statismoBuildModel(x, representer, sigma = 0, scale = TRUE)
```

Arguments

`x`	array of aligned 3D-coordinates or a list of aligned registered meshes.
`representer`	matrix or triangular mesh of class "mesh3d" with vertices corresponding to rows in the array.
`sigma`	noise in the data
`scale`	logical: set to TRUE, if scaling was involved in the registration.

Value

an object of class pPCA (pPCA-class)

See Also

pPCA, pPCA-class, rigidAlign, meshalign

Examples

```
require(Morpho)
data(boneData)
align <- rigidAlign(boneLM)$rotated
mymod <- statismoBuildModel(align,representer=align[,,1],sigma=2,scale=TRUE)
## save it
statismoSaveModel(mymod,"mymod.h5")
```

---

statismoGPmodel *expands a models variability by adding a Gaussian kernel function*

Description

expands a models variability by adding a Gaussian kernel function to the empiric covariance matrix and builds a low-rank approximation of the resulting PCA

Usage

```
statismoGPmodel(model, useEmpiric = TRUE, kernel = list(c(100, 70)),
  ncomp = 10, nystroem = 500)
```

Arguments

model	shape model of class pPCA
useEmpiric	logical: if TRUE, the empiric covariance kernel will be added to the Gaussian ones.
kernel	a list containing two valued vectors containing with the first entry specifiying the bandwidth and the second the scaling of the Gaussian kernels.
ncomp	integer: number of PCs to approximate
nystroem	number of samples to compute Nystroem approximation of eigenvectors

Value

returns a shape model of class pPCA

See Also

pPCA, pPCA-class

Examples

```
### this is a silly example with only 10 landmarks
require(Morpho)
data(boneData)
align <- rigidAlign(boneLM)$rotated
mod <- statismoBuildModel(align)
GPmod <- statismoGPmodel(mod,kernel=list(c(10,1),c(1,1)))##extend flexibility using two Gaussian kernels
GPmodNoEmp <- statismoGPmodel(mod,kernel=list(c(10,1),c(1,1)),useEmpiric = FALSE)##extend flexibility using two
PC1orig <- DrawSample(mod,2)# get shape in 2sd of first PC of originial model
PC1 <- DrawSample(GPmod,2)# get shape in 2sd of first PC of the extended model
PC1NoEmp <- DrawSample(GPmodNoEmp,2)# get shape in 2sd of first PC
##visualize the differences from the mean (green spheres)
deformGrid3d(PC1,DrawMean(GPmod),ngrid=0)##
deformGrid3d(PC1NoEmp,DrawMean(GPmod),ngrid=0,col1=4,add=TRUE)##only deviates in 5 landmarks from the mean (dark
deformGrid3d(PC1orig,DrawMean(GPmod),ngrid=0,col1=5,add=TRUE)
```

`statismoLoadModel/statismoSaveModel`

save and load a statistical model of class pPCA to statismo hdf5 format

---

Description

save and load a statistical model of class pPCA to statismo hdf5 format

Usage

```
statismoSaveModel(model, modelname = dataname)

statismoLoadModel(modelname, scale = TRUE)
```

Arguments

`model`	object of class `pPCA`
`modelname`	filename to read/save

Value

statismoLoadModel returns an object of class `pPCA` while statismoSaveModel saves an object of class `pPCA` to disk in the statismo file format.

See Also

`pPCA`

StatismoMatrices *Get Matrices from StatisticalModel class*

Description

Get Matrices from StatisticalModel class - such as projection matrices, covariance matrices or Jacobian

Usage

```
GetPCABasisMatrix(model)

GetOrthonormalPCABasisMatrix(model)

GetCovarianceMatrix(model)

GetJacobian(model, pt)

GetProjectionMatrix(model)
```

Arguments

`model`	model of class "pPCA"
`pt`	either an integer pointing to the index of the domain or a numeric vector of length 3 specifying a point on the domain of the model
`pt1`	either an integer pointing to the index of the domain or a numeric vector of length 3 specifying a point on the domain of the model
`pt2`	either an integer pointing to the index of the domain or a numeric vector of length 3 specifying a point on the domain of the model

Value

`GetPCABasisMatrix`	returns the (scaled) Basis of the latent space
`GetOrthonormalPCABasisMatrix`	returns the orthonormal Basis of the latent space
`GetCovarianceAtPoint`	returns the 3 x 3 covariance matrix for `pt1` and `pt2`
`GetJacobian`	returns the 3 x 3 Jacobian matrix at `pt`
`GetProjectionMatrix`	returns matrix to project a sample vector into the latent space (this is not a member function but might prove useful anyway)

StatismoModelMembers *Implementation/Emulation of the statismo StatisticalModel class.*

Description

Implementation/Emulation of the statismo StatisticalModel class.

Usage

```
DrawMean(model)

DrawMeanAtPoint(model, pt)

DrawSample(model, coefficients = NULL, addNoise = FALSE)

DrawSampleVector(model, coefficients, addNoise = FALSE)

DrawSampleAtPoint(model, coefficients, pt, addNoise = FALSE)

ComputeCoefficientsForDataset(model, dataset)

ComputeCoefficientsForPointValues(model, sample, pt, ptNoise = 0)

GetDomainPoints(model)

GetDomainSize(model)

EvaluateSampleAtPoint(model, sample, pt)

GetCovarianceAtPoint(model, pt1, pt2)
```

Arguments

`model`	object of class `pPCA`
`dataset`	an (already aligned) mesh or k x 3 matrix containing the datasets coordinates.
`coefficients`	specify coefficients in the latent space to draw a sample
`addNoise`	logical: if TRUE noise as specified in the model will be added to the returned sample
`ptNoise`	specify the noise estimated in the points.

Details

see `http://statismo.github.io/statismo/classdoc/html/classstatismo_1_1StatisticalModel.html` for details.

Value

`DrawMean`	Get the mean (either a matrix or a mesh3d)
`GetMeanVector`	Get the mean vector
`DrawMeanAtPoint`	Get a specific point of the mean (numeric vector)
`DrawSample`	Draw a sample from the model (either a matrix or a mesh3d)
`DrawMeanAtPoint`	Get a specific point of the mean (numeric vector)
`DrawSampleAtPoint`	Draw a sample of a specific point from the model (numeric vector)
`ComputeCoefficientsForDataset`	Computes the coefficients of the latent variables
`ComputeCoefficientsForPointValues`	Returns the coefficients of the latent variables for the given values provided in two k x 3 matrices or two vectors of length 3, or one matrix/vector and a vector containing the indices on the domain corresponding to these points
`GetDomainPoints`	a matrix containing the points of the model's domain
`GetDomainSize`	get the size of the model's domain
`EvaluateSampleAtPoint`	Returns the value of the given sample at the point specified (either as point on the domain or as an index)

`StatismoParameters` *Get model parameters*

Description

Get model parameters such as variance or noise variance

Usage

```
GetNoiseVariance(model)

GetMeanVector(model)

GetPCAVarianceVector(model)
```

Arguments

`model`	model of class "pPCA"

Value

`GetNoiseVariance`	returns the estimated noise in the model
`GetPCAVarianceVector`	returns the variance in the model
`GetMeanVector`	returns the model's mean vector

`StatismoSample` *Retrieve information about a sample from the model*

Description

Retrieve information about a sample from the model

Usage

```
ComputeLogProbabilityOfDataset(model, dataset)

ComputeProbabilityOfDataset(model, dataset)
```

Arguments

`model`	model of class "pPCA"
`dataset`	a matrix or mesh3d aligned to the model's mean

Value

`ComputeLogProbabilityOfDataset`	returns the log-probability density for the sample
`ComputeProbabilityOfDataset`	returns the probability density for the sample

See Also

`getDataLikelihood`

Index