Package 'ProDenICA'

January 20, 2025

Type Package

Title Product Density Estimation for ICA using Tilted Gaussian Density Estimates

Version 1.1

Date 2022-02-20

Author Trevor Hastie, Rob Tibshirani

Maintainer Trevor Hastie <hastie@stanford.edu>

Depends gam

Description A direct and flexible method for estimating an ICA model. This approach estimates the densities for each component directly via a tilted Gaussian. The tilt functions are estimated via a GAM Poisson model. Details can be found in ``Elements of Statistical Learning (2nd Edition)'' in Section 14.7.4.

License GPL-2

URL https://hastie.su.domains/ElemStatLearn/printings/ESLII_print12_toc.pdf

NeedsCompilation no

Repository CRAN

Date/Publication 2022-02-21 08:50:23 UTC

Contents

amari *Compute the 'Amari' distance between two matrices*

Description

The Amari distance is a measure between two nonsingular matrices. Useful for checking for convergence in ICA algorithms, and for comparing solutions.

Usage

```
amari(V, W, orth = FALSE)
```

Arguments

`V`	first matrix
`W`	second matrix
`orth`	are the matrices orthogonal; default is `orth=FALSE`

Details

Formula is given in second reference below, page 570.

Value

a numeric distance metween 0 and 1

Author(s)

Trevor Hastie

References

Bach, F. and Jordan, M. (2002). Kernel independent component analysis, Journal of Machine Learning Research 3: 1-48
Hastie, T., Tibshirani, R. and Friedman, J. (2009) Elements of Statistical Learning (2nd edition), Springer.
`https://hastie.su.domains/ElemStatLearn/printings/ESLII_print12_toc.pdf`

See Also

`ProDenICA`

Examples

```
dist="n"
N=1024
p=2
A0<-mixmat(p)
s<-scale(cbind(rjordan(dist,N),rjordan(dist,N)))
x <- s %*% A0
###Whiten the data
x <- scale(x, TRUE, FALSE)
sx <- svd(x) ### orthogonalization function
x <- sqrt(N) * sx$u
target <- solve(A0)
target <- diag(sx$d) %*% t(sx$v) %*% target/sqrt(N)
W0 <- matrix(rnorm(2*2), 2, 2)
W0 <- ICAorthW(W0)
W1 <- ProDenICA(x, W0=W0,trace=TRUE,Gfunc=G1)$W
fit=ProDenICA(x, W0=W0,Gfunc=GPois,trace=TRUE, density=TRUE)
W2 <- fit$W
#distance of FastICA from target
amari(W1,target)
#distance of ProDenICA from target
amari(W2,target)
```

G1 *FastICA contrast functions.*

Description

contrast functions for computing the negentropy criteria used in FastICA; see references.

Usage

```
G1(s, a=1)
G0(s, a=1)
```

Arguments

`s`	estimated independent component
`a`	additional tuning parameter (only used in `G1`)

Value

a list with components

`Gs`	contrast function evaluated at values of `x`. `mean(Gs)` is measure of negentropy.
`gs`	estimated first derivative of `Gs` at x
`gps`	estimated second derivative of `Gs` at x

Author(s)

Trevor Hastie and Rob Tibshirani

References

Hyvarinen, A., Karhunen, J. and Oja, E. (2001). Independent Component Analysis, Wiley, New York
Hastie, T. and Tibshirani, R. (2003) *Independent Component Analysis through Product Density Estimation* in *Advances in Neural Information Processing Systems 15* (Becker, S. and Obermayer, K., eds), MIT Press, Cambridge, MA. pp 649-656
Hastie, T., Tibshirani, R. and Friedman, J. (2009) Elements of Statistical Learning (2nd edition), Springer.
https://hastie.su.domains/ElemStatLearn/printings/ESLII_print12_toc.pdf

See Also

`GPois` and `ProDenICA`

Examples

```
p=2
### Can use letters a-r below for dist
dist="n"
N=1024
A0<-mixmat(p)
s<-scale(cbind(rjordan(dist,N),rjordan(dist,N)))
x <- s %*% A0
fit=ProDenICA(x,Gfunc=G1, whiten=TRUE)
```

GPois *Fit a tilted Gaussian density via a Poisson GAM*

Description

This is a contrast method for `ProDenICA`. It fits a tilted Gaussian density estimate by multiplying the Gaussian density by an exponential tilt function using a cubic smoothing spline

Usage

```
GPois(x, df = 6, B = 500, order = 1, widen = 1.2, density.return = FALSE, ...)
```

Arguments

`x`	vector of real values
`df`	degrees of freedom for the smoothing-spline fit; default is 6
`B`	number of grid points for density estimate; default is 500

`order` A robustness parameter to avoid responding to outliers in `x`. The range of `x` is estimated by the `order`th and `n-order+1`th order statistics. Default is `order=1`

`widen` an expansion factor to widen the range of `x`; default is `widen=1.2`

`density.return` logical variable, with default `FALSE`. If `density.return=TRUE`, the estimated density is returned

`...` additional arguments to GAM; typically not used

Details

See Section 14.7.4 of 'Elements of Statistical Learning (Hastie, Tibshirani and Friedman, 2009, 2nd Edition)' for details

Value

a list with components

`Gs` estimated contrast function, which is the log of the tilting function, evaluated at the original values of `x`. `mean(Gs)` is measure of negentropy

`gs` estimated first derivative of `Gs` at `x`

`gps` estimated second derivative of `Gs` at `x`

`density` if `density.return=TRUE`, a list with components `$x` the grid of B values of `x`, and `$y` the estimated density.

Author(s)

Trevor Hastie and Rob Tibshirani

References

Hastie, T. and Tibshirani, R. (2003) *Independent Component Analysis through Product Density Estimation* in *Advances in Neural Information Processing Systems 15* (Becker, S. and Obermayer, K., eds), MIT Press, Cambridge, MA. pp 649-656

Hastie, T., Tibshirani, R. and Friedman, J. (2009) Elements of Statistical Learning (2nd edition), Springer.

https://hastie.su.domains/ElemStatLearn/printings/ESLII_print12_toc.pdf

See Also

`ProDenICA`, `G1` and `G0`

Examples

```
p=2
### Can use letters a-r below for dist
dist="n"
N=1024
A0<-mixmat(p)
s<-scale(cbind(rjordan(dist,N),rjordan(dist,N)))
x <- s %*% A0
```

```
fit=ProDenICA(x,Gfunc=GPois, whiten=TRUE, density=TRUE)
par(mfrow=c(2,1))
plot(fit)
```

`ICAorthW` *turn a matrix W into an orthogonal matrix*

Description

use the SVD to orthogonalize a matrix

Usage

```
ICAorthW(W)
```

Arguments

`W`	input matrix

Details

simply replace the D matrix of the SVD of W by the identity

Value

orthogonalized version of W

Note

If W=UDV', then returns UV'

Author(s)

Trevor Hastie

Examples

```
W0 <- matrix(rnorm(2*2), 2, 2)
W0 <- ICAorthW(W0)
```

`mixmat` *Generate a random mixing matrix with condition number between 1 and 2*

Description

A simple function for generating a 'well behaved' random square mixing matrix

Usage

```
mixmat(p = 2)
```

Arguments

`p`	dimnesion of matrix

Details

Generates a random matrix by constructing its SVD. The singular values are drawn from a uniform on [1,2], hence guaranteeing a condition number between 1 and 2

Value

a pxp matrix

Author(s)

Trevor Hastie

`ProDenICA` *Product Density Independent Component Analysis*

Description

Fits an ICA model by directly estimating the densities of the independent components using Poisson GAMs. The densities have the form of tilted Gaussians, and hense directly estimate the contrast functions that lead to negentropy measures. This function supports Section 14.7.4 of 'Elements of Statistical Learning (Hastie, Tibshirani and Friedman, 2009, 2nd Edition)'. Models include 'FastICA'.

Usage

```
ProDenICA(x, k = p, W0 = NULL, whiten = FALSE, maxit = 20, thresh = 1e-07,
restarts = 0, trace = FALSE, Gfunc = GPois, eps.rank = 1e-07, ...)
```

Arguments

`x` input matrix

`k` Number of components required, less than or equal to the number of columns of x

`W0` Optional initial matrix (for comparing algorithms)

`whiten` Logical variable - should x be whitened. If TRUE, the SVD of X=UDV' is computed, and U is used (up to rank(X) columns). Also k is reduced to min(k,rank(X)). If FALSE (default), it is assumed that the user has pre-whitened x (and if not, the function may not perform properly)

`maxit` Maximum number of iterations; default is 20

`thresh` Convergence threshold, in terms of relative change in Amari metric; dfault is 1e-7

`restarts` Number of random restarts; default is 0

`trace` Trace iterations; default is FALSE

`Gfunc` Contrast functional which is basis for negentropy measure. Default is `'GPois'` which fits a tilted Gaussian density using a Poisson GAM. Other options are `'G1'` (cosh negentropy) and `'G0'` (kurtosis negentropy)

`eps.rank` Threshold for deciding rank of x if option `whiten=TRUE`. Any singular value less than `eps.thresh` smaller than the largest is treated as zero

`...` Additional arguments for `Gfunc` areguments

Details

See Section 14.7.4 of *Elements of Statistical Learning* (Hastie, Tibshirani and Friedman, 2009, 2nd Edition)

Value

An object of S3 class `"ProDenICA"` is returned, with the following components:

`W` Orthonormal matrix that takes the whitened version of x to the independent components

`negentropy` The total negentropy measure of this solution

`s` the matrix of k independent components

`whitner` if `whiten=TRUE`, the matrix that whitens `x`, else `NULL`

`call` the call that produced this object

`density` If `Gfunc=GPois`, an list of length `k` with the density estimates for each component

Author(s)

Trevor Hastie and Rob Tibshirani

References

Hastie, T. and Tibshirani, R. (2003) *Independent Component Analysis through Product Density Estimation* in *Advances in Neural Information Processing Systems 15* (Becker, S. and Obermayer, K., eds), MIT Press, Cambridge, MA. pp 649-656
Hastie, T., Tibshirani, R. and Friedman, J. (2009) Elements of Statistical Learning (2nd edition), Springer.
https://hastie.su.domains/ElemStatLearn/printings/ESLII_print12_toc.pdf

See Also

GPois, G1 and plot method.

Examples

```
p=2
### Can use letters a-r below for dist
dist="n"
N=1024
A0<-mixmat(p)
s<-scale(cbind(rjordan(dist,N),rjordan(dist,N)))
x <- s %*% A0
###Whiten the data
x <- scale(x, TRUE, FALSE)
sx <- svd(x) ### orthogonalization function
x <- sqrt(N) * sx$u
target <- solve(A0)
target <- diag(sx$d) %*% t(sx$v) %*% target/sqrt(N)
W0 <- matrix(rnorm(2*2), 2, 2)
W0 <- ICAorthW(W0)
W1 <- ProDenICA(x, W0=W0,trace=TRUE,Gfunc=G1)$W
fit=ProDenICA(x, W0=W0,Gfunc=GPois,trace=TRUE, density=TRUE)
W2 <- fit$W
#distance of FastICA from target
amari(W1,target)
#distance of ProDenICA from target
amari(W2,target)
par(mfrow=c(2,1))
plot(fit)
```

---

rjordan *Generate source densities for ICA*

Description

Functions for generating the source densities used in Bach and Jordan (2002), and reused in Hastie and Tibshirani (2003)

Usage

```
rjordan(letter, n, ...)
djordan(letter, x, ...)
```

Arguments

`letter`	one of the 18 letters `a-r`; see Figure 14.42 on page 569 of 'Elements of Statistical Learning'
`n`	number of samples
`x`	ordinates at which to compute density
`...`	place filler for additional arguments

Details

This function produces the example densities used in Bach and Jordan (2002), and copied by Hastie and Tibshirani (2003). They include the 't', uniform, mixtures of exponentials and many mixtures of gaussian densities. Each are standardized to have mean zero and variance 1.

Value

Either a vector of density values the length of `x` for `djordan`, or a vector of `n` draws for `rjordan`

Author(s)

Trevor Hastie

References

Bach, F. and Jordan, M. (2002). Kernel independent component analysis, Journal of Machine Learning Research 3: 1-48
Hastie, T. and Tibshirani, R. (2003) *Independent Component Analysis through Product Density Estimation* in *Advances in Neural Information Processing Systems 15* (Becker, S. and Obermayer, K., eds), MIT Press, Cambridge, MA. pp 649-656
Hastie, T., Tibshirani, R. and Friedman, J. (2009) Elements of Statistical Learning (2nd edition), Springer.
https://hastie.su.domains/ElemStatLearn/printings/ESLII_print12_toc.pdf

See Also

`ProDenICA`

Examples

```
dist="n"
N=1024
s<-scale(cbind(rjordan(dist,N),rjordan(dist,N)))
```

Index