Package 'eegkit'

September 10, 2014

Type Package

Title Toolkit for electroencephalography data

Version 1.0-0

Date 2014-09-09

Author Nathaniel E. Helwig <helwig@umn.edu>

Maintainer Nathaniel E. Helwig <helwig@umn.edu>

Depends R (>= 3.1.1), bigsplines, eegkitdata, ica, rgl

Description Analysis and visualization tools for electroencephalography (EEG) data. Includes functions for plotting (a) EEG caps, (b) single- and multi-
channel EEG time courses, and (c) EEG spatial maps. Also includes smoothing and Independent Component Analysis functions for EEG data analysis.

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2014-09-10 00:55:52

R topics documented:

eegkit-package 2
eegcap 3
eegcoord 6
eegdense 7
eeghead 8
eegica 9
eegmesh 11
eegsmooth 12
eegspace 16
eegtime 18

Index **21**

eegkit-package *Toolkit for electroencephalography data*

Description

Analysis and visualization tools for electroencephalography (EEG) data. Includes functions for plotting (a) EEG caps, (b) single- and multi-channel EEG time courses, and (c) EEG spatial maps. Also includes smoothing and Independent Component Analysis functions for EEG data analysis.

Details

The function `eegcap` plots EEG caps (according to 10-20 system) in two or three dimensions. The function `eegtime` plots EEG time courses. The function `eegspace` plots EEG spatial maps. The function `eegica` calculates temporal or spatial ICA decomposition of EEG data. The function `eegsmooth` performs temporal and/or spatial smoothing of EEG data.

Author(s)

Nathaniel E. Helwig <helwig@umn.edu>

Maintainer: Nathaniel E. Helwig <helwig@umn.edu>

References

Adler, D., Murdoch, D., and others (2014). *rgl: 3D visualization device system (OpenGL)*. http://CRAN.R-project.org/package=rgl

Bache, K. & Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Begleiter, H. *Neurodynamics Laboratory*. State University of New York Health Center at Brooklyn.

Bell, A.J. & Sejnowski, T.J. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation, 7*, 1129-1159.

Cardoso, J.F., & Souloumiac, A. (1993). Blind beamforming for non-Gaussian signals. *IEE Proceedings-F, 140*, 362-370.

Cardoso, J.F., & Souloumiac, A. (1996). Jacobi angles for simultaneous diagonalization. *SIAM Journal on Matrix Analysis and Applications, 17*, 161-164.

Helwig, N.E. (in prep). On the relationship between FastICA and Infomax: Fast and robust fixed point algorithms for information-maximization.

Helwig, N. E. (2013). *Fast and stable smoothing spline analysis of variance models for large samples with applications to electroencephalography data analysis*. Unpublished doctoral dissertation. University of Illinois at Urbana-Champaign.

Helwig, N.E. (2014). *bigsplines: Smoothing splines for large samples*. http://CRAN.R-project.org/package=bigsplines

Helwig, N.E. (2014). *ica: Independent Componenet Analysis*. http://CRAN.R-project.org/package=ica

Helwig, N.E. & Hong, S. (2013). A critique of Tensor Probabilistic Independent Component Analysis: Implications and recommendations for multi-subject fMRI data analysis. *Journal of Neuroscience Methods, 213*, 263-273.

Helwig, N. E. and Ma, P. (in prep). Nonparametric Gaussian regression for ultra large samples: Scalable computation via rounding parameters.

Helwig, N. E. and Ma, P. (in press). Fast and stable multiple smoothing parameter selection in smoothing spline analysis of variance models with large samples. *Journal of Computational and Graphical Statistics*.

Ingber, L. (1997). Statistical mechanics of neocortical interactions: Canonical momenta indicatros of electroencephalography. *Physical Review E, 55*, 4578-4593.

Ingber, L. (1998). Statistical mechanics of neocortical interactions: Training and testing canonical momenta indicators of EEG. *Mathematical Computer Modelling, 27*, 33-64.

Oostenveld, R., and Praamstra, P. (2001). The Five percent electrode system for high-resolution EEG and ERP measurements. *Clinical Neurophysiology, 112*, 713-719.

Schlager, S. & authors of VCGLIB. (2014). Rvcg: Manipulations of triangular meshes (smoothing, quadric edge collapse decimation, im- and export of various mesh file-formats, cleaning, etc.) based on the VCGLIB API. R packge version 0.7.1. http://CRAN.R-project.org/package=Rvcg.

Examples

```
# See examples for eegcap, eegtime, eegspace, eegica, and eegsmooth
```

`eegcap` *Draws EEG cap with selected electrodes (3d or 2d)*

Description

Creates two- or three-dimensional plot of electroencephalography (EEG) cap with user-input electrodes. Three-dimensional plots are created using the `eegcoord` data and the `plot3d` function (from `rgl` package). Currently supports 84 scalp electrodes, and plots according to the international 10-10 system. Includes customization options (e.g., each electrode can have a unique plotting color, size, label color, etc.).

Usage

```
eegcap(electrodes="10-10",type=c("3d","2d"),plotlabels=TRUE,
       plotaxes=FALSE,main="",xyzlab=NULL,cex.point=NULL,
       col.point=NULL,cex.label=NULL,col.label=NULL,nose=TRUE,
       ears=TRUE,head=TRUE,col.head="AntiqueWhite",...)
```

Arguments

`electrodes`	Character vector with electrodes to plot. Each element of `electrodes` must match one of the 89 reference electrodes (see Notes). Mismatches are ignored (not plotted). Input is NOT case sensitive. Default plots all available electrodes (full 10-10 system).
`type`	Type of plot to create: `type="3d"` produces three-dimensional plot, whereas `type="2d"` produces two-dimensional projection plot (bird's eye view).

plotlabels	If TRUE, the electrode labels are plotted.
plotaxes	If TRUE, the axes are plotted.
main	Title to use for plot. Default is no title
xyzlab	Axis labels to use for plot. If type="2d", then xyzlab should be two-element character vector giving x and y axis labels. If type="3d", then xyzlab should be three-element character vector giving x, y, and z axis labels.
cex.point	Size of electrode points. Can have a unique size for each electrode.
col.point	Color of electrode points. Can have a unique color for each electrode.
cex.label	Size of electrode labels. Can have a unique size for each electrode label. Input is ignored if plotlabels=FALSE is used.
col.label	Color of electrode labels. Can have a unique color for each electrode label. Input is ignored if plotlabels=FALSE is used.
nose	If TRUE, triangle is plotted to represent the subject's nose. Ignored if type="3d".
ears	If TRUE, ovals are plotted to represent the subject's ears. Ignored if type="3d".
head	If TRUE, head is plotted. Ignored if type="2d".
col.head	Color for dummy head in 3d plot. Ignored if type="2d".
...	Optional inputs for plot or plot3d function.

Value

Produces plot of EEG cap with NULL return value.

Note

Currently supports 84 scalp electrodes (plus ears and nose): A1 A2 AF1 AF2 AF3 AF4 AF5 AF6 AF7 AF8 AFZ C1 C2 C3 C4 C5 C6 CP1 CP2 CP3 CP4 CP5 CP6 CPZ CZ F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 FC1 FC2 FC3 FC4 FC5 FC6 FCZ FP1 FP2 FPZ FT7 FT8 FT9 FT10 FZ I1 I2 IZ NZ O1 O2 OZ P1 P2 P3 P4 P5 P6 P7 P8 P9 P10 PO1 PO2 PO3 PO4 PO5 PO6 PO7 PO8 PO9 PO10 POZ PZ T7 T8 T9 T10 TP7 TP8 TP9 TP10

See eegcoord for the coordinates used to create plot.

To save three-dimensional plots, use the rgl.postscript function (from rgl package).

Author(s)

Nathaniel E. Helwig <helwig@umn.edu>

References

Adler, D., Murdoch, D., and others (2014). *rgl: 3D visualization device system (OpenGL)*. http://CRAN.R-project.org/package=rgl

Oostenveld, R., and Praamstra, P. (2001). The Five percent electrode system for high-resolution EEG and ERP measurements. *Clinical Neurophysiology, 112*, 713-719.

Examples

```
##########   EXAMPLE 1   ##########

# plot 10-10 system (default):

# plot full cap 3d (default options)
eegcap()

# plot full cap 2d (default options)
eegcap(type="2d")

# plot full cap 3d (different color for ears and nose)
data(eegcoord)
mycols=rep("black",87)
mycols[rownames(eegcoord)=="A1"]="green"
mycols[rownames(eegcoord)=="A2"]="light blue"
mycols[rownames(eegcoord)=="NZ"]="pink"
eegcap(col.point=mycols)

# plot full cap 2d (different color for ears and nose)
mycols=rep("white",87)
mycols[rownames(eegcoord)=="A1"]="green"
mycols[rownames(eegcoord)=="A2"]="light blue"
mycols[rownames(eegcoord)=="NZ"]="pink"
eegcap(type="2d",col.point=mycols)



##########   EXAMPLE 2   ##########

# plot 10-20 system:

# plot 3d cap with labels
eegcap(electrodes="10-20")

# plot 3d cap without labels
eegcap("10-20",plotlabels=FALSE)

# plot 2d cap with labels
eegcap("10-20","2d")

# plot 2d cap without labels
eegcap("10-20","2d",plotlabels=FALSE)



##########   EXAMPLE 3   ##########

# plot custom subset of electrodes
myelectrodes=c("FP1","FP2","FPZ","F7","F3","FZ",
               "F4","F8","T7","C3","CZ","C4","T8",
```

```
                "P7","P3","PZ","P4","P8","O1","O2")
eegcap(myelectrodes,"2d")
```

`eegcoord` *EEG cap coordinates (3d and 2d projection)*

Description

Three-dimensional electroencephalography (EEG) electrode coordinates (measured in cm), and corresponding projection onto two-dimensional xy plane. Contains 84 scalp electrodes, as well as nose and ears.

Usage

```
data(eegcoord)
```

Format

A data frame with 87 observations and the following 5 variables:

x x-coordinate of 3d cap (numeric).

y y-coordinate of 3d cap (numeric).

z z-coordinate of 3d cap (numeric).

xproj Projected x-coordinate of 2d cap (numeric).

yproj Projected y-coordinate of 2d cap (numeric).

Electrode channel name labels can be obtained using `rownames(eegcoord)`.

Author(s)

Nathaniel E. Helwig <helwig@umn.edu>

Source

Created by Nathaniel E. Helwig (2014) using:

Adler, D., Murdoch, D., and others (2014). *rgl: 3D visualization device system (OpenGL)*. http://CRAN.R-project.org/package=rgl

Oostenveld, R., and Praamstra, P. (2001). The Five percent electrode system for high-resolution EEG and ERP measurements. *Clinical Neurophysiology, 112*, 713-719.

Schlager, S. & authors of VCGLIB. (2014). Rvcg: Manipulations of triangular meshes (smoothing, quadric edge collapse decimation, im- and export of various mesh file-formats, cleaning, etc.) based on the VCGLIB API. R packge version 0.7.1. http://CRAN.R-project.org/package=Rvcg.

Examples

```
##########   EXAMPLE   ##########

data(eegcoord)
enames=rownames(eegcoord)
plot3d(eegcoord[,1],eegcoord[,2],eegcoord[,3],size=10,col="green")
text3d(eegcoord[,1],eegcoord[,2],eegcoord[,3],texts=enames,col="blue")
plot(eegcoord[,4],eegcoord[,5],cex=2,col="green",pch=19)
text(eegcoord[,4],eegcoord[,5],labels=enames,col="blue")
```

---

eegdense *Dense EEG cap coordinates (3d and 2d projection)*

Description

Dense (hypothetical) three-dimensional electroencephalography (EEG) electrode coordinates, and corresponding projection onto two-dimensional plane. Dense cap spans the 84 scalp electrodes defined in eegcoord.

Usage

```
data(eegdense)
```

Format

A data frame with 977 observations and the following 5 variables:

x x-coordinate of 3d cap (numeric).

y y-coordinate of 3d cap (numeric).

z z-coordinate of 3d cap (numeric).

xproj Projected x-coordinate of 2d cap (numeric).

yproj Projected y-coordinate of 2d cap (numeric).

Author(s)

Nathaniel E. Helwig <helwig@umn.edu>

Source

Created by Nathaniel E. Helwig (2014) using:

Adler, D., Murdoch, D., and others (2014). *rgl: 3D visualization device system (OpenGL)*. http://CRAN.R-project.org/package=rgl

Oostenveld, R., and Praamstra, P. (2001). The Five percent electrode system for high-resolution EEG and ERP measurements. *Clinical Neurophysiology, 112*, 713-719.

Schlager, S. & authors of VCGLIB. (2014). Rvcg: Manipulations of triangular meshes (smoothing, quadric edge collapse decimation, im- and export of various mesh file-formats, cleaning, etc.) based on the VCGLIB API. R packge version 0.7.1. http://CRAN.R-project.org/package=Rvcg.

Examples

```
##########   EXAMPLE   ##########

data(eegdense)
plot3d(eegdense[,1],eegdense[,2],eegdense[,3],size=10,col="green")
plot(eegdense[,4],eegdense[,5],cex=1,col="green",pch=19)
```

---

eeghead *Dummy head for 3d EEG plots*

Description

Contains mesh3d object of dummy head, which is used in the plotting functions eegcap and eegspace. This is a transformed (translated, rotated, and rescaled) vesion of the dummyhead object from the Rvcg package.

Usage

```
data(eeghead)
```

Format

mesh3d object

Author(s)

Nathaniel E. Helwig <helwig@umn.edu>

Source

Created by Nathaniel E. Helwig (2014) using:

Adler, D., Murdoch, D., and others (2014). *rgl: 3D visualization device system (OpenGL)*. http://CRAN.R-project.org/package=rgl

Schlager, S. & authors of VCGLIB. (2014). Rvcg: Manipulations of triangular meshes (smoothing, quadric edge collapse decimation, im- and export of various mesh file-formats, cleaning, etc.) based on the VCGLIB API. R packge version 0.7.1. http://CRAN.R-project.org/package=Rvcg.

Examples

```
##########   EXAMPLE   ##########

data(eeghead)
shade3d(eeghead)
eeghead$material$color=rep("black",length(eeghead$material$color))
wire3d(eeghead)
```

eegica *Independent Component Analysis of EEG data*

Description

Computes temporal (default) or spatial ICA decomposition of EEG data. Can use Infomax (default), FastICA, or JADE algorithm. ICA computations are conducted via icaimax, icafast, or icajade from the ica package.

Usage

```
eegica(X,nc,center=TRUE,maxit=100,tol=1e-6,Rmat=diag(nc),
       type=c("time","space"),method=c("imax","fast","jade"),...)
```

Arguments

X	Data matrix with n rows (channels) and p columns (time points).
nc	Number of components to extract.
center	If TRUE, columns of X are mean-centered before ICA decomposition.
maxit	Maximum number of algorithm iterations to allow.
tol	Convergence tolerance.
Rmat	Initial estimate of the nc-by-nc orthogonal rotation matrix.
type	Type of ICA decomposition: type="time" extracts temporally independent components, and type="space" extracts spatially independent components.
method	Method for ICA decomposition: method="imax" uses Infomax, method="fast" uses FastICA, and method="jade" uses JADE.
...	Additional inputs to icaimax or icafast function.

Details

ICA Model The ICA model can be written as X=tcrossprod(S,M)+E, where columns of S contain the source signals, M is the mixing matrix, and columns of E contain the noise signals. Columns of X are assumed to have zero mean. The goal is to find the unmixing matrix W such that columns of S=tcrossprod(X,W) are independent as possible.

Whitening Without loss of generality, we can write M=P%*%R where P is a tall matrix and R is an orthogonal rotation matrix. Letting Q denote the pseudoinverse of P, we can whiten the data using Y=tcrossprod(X,Q). The goal is to find the orthongal rotation matrix R such that the source signal estimates S=Y%*%R are as independent as possible. Note that W=crossprod(R,Q).

Infomax The Infomax approach finds the orthogonal rotation matrix R that (approximately) maximizes the joint entropy of a nonlinear function of the estimated source signals. See Bell and Sejnowski (1995) and Helwig (in prep) for specifics of algorithms.

FastICA The FastICA algorithm finds the orthogonal rotation matrix R that (approximately) maximizes the negentropy of the estimated source signals. Negentropy is approximated using

$$J(s) = [E\{G(s)\} - E\{G(z)\}]^2$$

where E denotes the expectation, G is the contrast function, and z is a standard normal variable. See Hyvarinen (1999) for specifics of fixed-point algorithm.

JADE The JADE approach finds the orthogonal rotation matrix `R` that (approximately) diagonalizes the cumulant array of the source signals. See Cardoso and Souloumiac (1993,1996) and Helwig and Hong (2013) for specifics of the JADE algorithm.

Value

`S`	Matrix of source signal estimates (`S=Y%*%R`).
`M`	Estimated mixing matrix.
`W`	Estimated unmixing matrix (`W=crossprod(R,Q)`).
`Y`	Whitened data matrix.
`Q`	Whitening matrix.
`R`	Orthogonal rotation matrix.
`vafs`	Variance-accounted-for by each component.
`iter`	Number of algorithm iterations.
`type`	ICA type (same as input).
`method`	ICA method (same as input).

Note

If `type="time"`, the data matrix is transposed before calling ICA algorithm (i.e., `X=t(X)`), and the columns of the tranposed data matrix are centered.

Author(s)

Nathaniel E. Helwig <helwig@umn.edu>

References

Bell, A.J. & Sejnowski, T.J. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation, 7*, 1129-1159.

Cardoso, J.F., & Souloumiac, A. (1993). Blind beamforming for non-Gaussian signals. *IEE Proceedings-F, 140*, 362-370.

Cardoso, J.F., & Souloumiac, A. (1996). Jacobi angles for simultaneous diagonalization. *SIAM Journal on Matrix Analysis and Applications, 17*, 161-164.

Helwig, N.E. (in prep). On the relationship between FastICA and Infomax: Fast and robust fixed point algorithms for information-maximization.

Helwig, N.E. (2014). *ica: Independent Componenet Analysis*. http://CRAN.R-project.org/package=ica

Helwig, N.E. & Hong, S. (2013). A critique of Tensor Probabilistic Independent Component Analysis: Implications and recommendations for multi-subject fMRI data analysis. *Journal of Neuroscience Methods, 213*, 263-273.

Hyvarinen, A. (1999). Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks, 10*, 626-634.

Examples

```
##########   EXAMPLE   ##########

# get "c" subjects of "eegdata" data
data(eegdata)
idx=which(eegdata$group=="c")
eegdata=eegdata[idx,]

# get average data (across subjects)
eegmean=tapply(eegdata$voltage,list(eegdata$channel,eegdata$time),mean)

# remove ears and nose
acnames=rownames(eegmean)
idx=c(which(acnames=="X"),which(acnames=="Y"),which(acnames=="nd"))
eegmean=eegmean[-idx,]

# get spatial coordinates (for plotting)
data(eegcoord)
cidx=match(rownames(eegmean),rownames(eegcoord))

# temporal ICA with 4 components
icatime=eegica(eegmean,4)
#x11(width=5,height=10)
par(mfrow=c(4,2))
tseq=(0:255)*1000/255
for(j in 1:4){
  par(mar=c(5.1,4.6,4.1,2.1))
  sptitle=bquote("VAF:  "*.(round(icatime$vafs[j],4)))
  eegtime(tseq,icatime$S[,j],main=bquote("Component  "*.(j)),cex.main=1.5)
  eegspace(eegcoord[cidx,4:5],icatime$M[,j],main=sptitle)
}

# spatial ICA with 4 components
icaspace=eegica(eegmean,4,type="space")
#x11(width=5,height=10)
par(mfrow=c(4,2))
tseq=(0:255)*1000/255
for(j in 1:4){
  par(mar=c(5.1,4.6,4.1,2.1))
  sptitle=bquote("VAF:  "*.(round(icaspace$vafs[j],4)))
  eegtime(tseq,icaspace$M[,j],main=bquote("Component  "*.(j)),cex.main=1.5)
  eegspace(eegcoord[cidx,4:5],icaspace$S[,j],main=sptitle)
}
```

---

eegmesh *EEG cap for dense coordinates*

Description

Contains mesh3d object of eegdense, which is used in the plotting function eegspace.

Usage

```
data(eegmesh)
```

Format

mesh3d object

Author(s)

Nathaniel E. Helwig <helwig@umn.edu>

Source

Created by Nathaniel E. Helwig (2014) using:

Adler, D., Murdoch, D., and others (2014). *rgl: 3D visualization device system (OpenGL)*. http://CRAN.R-project.org/package=rgl

Oostenveld, R., and Praamstra, P. (2001). The Five percent electrode system for high-resolution EEG and ERP measurements. *Clinical Neurophysiology, 112*, 713-719.

Schlager, S. & authors of VCGLIB. (2014). Rvcg: Manipulations of triangular meshes (smoothing, quadric edge collapse decimation, im- and export of various mesh file-formats, cleaning, etc.) based on the VCGLIB API. R packge version 0.7.1. http://CRAN.R-project.org/package=Rvcg.

Examples

```
##########   EXAMPLE   ##########

data(eegmesh)
wire3d(eegmesh)
eegmesh$material$color=rep("red",length(eegmesh$material$color))
shade3d(eegmesh)
```

---

eegsmooth *Spatial and/or temporal smoothing of EEG data*

Description

Smooths single- or multi-channel electroencephalography (EEG) with respect to space and/or time. Uses the bigspline, bigtps, and bigssa functions (from bigsplines package) for smoothing.

Usage

```
eegsmooth(voltage,space=NULL,time=NULL,nknots=NULL,rparm=NULL,
          lambdas=NULL,skip.iter=TRUE,se.fit=FALSE,rseed=1234)
```

Arguments

voltage	Vector of recorded EEG voltage at each row in space.
space	Matrix of electrode coordinates (in three-dimensions) at which EEG was recorded. If space=NULL, data are temporally smoothed only.
time	Vector of time points at which EEG was recorded. If time=NULL, data are spatially smoothed only.
nknots	Number of knots to sample for smoothing. Positive integer.
rparm	Rounding parameter(s) to use for smoothing. See Notes and Examples.
lambdas	Smoothing parameter(s) to use for smoothing.
skip.iter	If FALSE, iterative spatial-temporal smoothing is skipped. Ignored if space=NULL or time=NULL.
se.fit	If TRUE, standard errors of smoothed values are calculated.
rseed	Random seed to use for knot selection. Set rseed=NULL to obtain different knots each time, or set rseed to any positive integer to use a different random seed.

Value

For temporal smoothing only: an object of class "css" (see bigspline).

For spatial smoothing only: an object of class "tps" (see bigtps).

For spatial-temporal smoothing: an object of class "ssa" (see bigssa).

Note

For temporal smoothing only (i.e., space=NULL), the input rparm should be a positive scalar less than 1. Larger values produce faster (but less accurate) approximations. Default is 0.01, which I recommend for temporal smoothing; rparm=0.005 may be needed for particuarly rough signals, and rparm=0.02 could work for smoother signals.

For spatial smoothing only (i.e., time=NULL), the input rparm should be a positive scalar giving the rounding unit for the spatial coordinates. For example, rparm=0.1 rounds each coordinate to the nearest 0.1 (same as round(space,1)).

For spatial-temporal smoothing (i.e., both space and time are non-null), the input rparm should be a list of the form rparm=list(space=0.1,time=0.01), where the 0.1 and 0.01 can be replaced by your desired rounding parameters.

Setting rparm=NA will use the full data solution; this is more computationally expensive, and typically produces a solution very similar to using rparm=0.01 (see references).

Author(s)

Nathaniel E. Helwig <helwig@umn.edu>

References

Helwig, N. E. (2013). *Fast and stable smoothing spline analysis of variance models for large samples with applications to electroencephalography data analysis*. Unpublished doctoral dissertation. University of Illinois at Urbana-Champaign.

Helwig, N. E. and Ma, P. (in prep). Nonparametric Gaussian regression for ultra large samples: Scalable computation via rounding parameters.

Helwig, N. E. and Ma, P. (in press). Fast and stable multiple smoothing parameter selection in smoothing spline analysis of variance models with large samples. *Journal of Computational and Graphical Statistics*.

Examples

```
##########   EXAMPLE 1: Temporal    ##########

# get "PZ" electrode of "c" subjects in "eegdata" data
data(eegdata)
idx=which(eegdata$channel=="PZ" & eegdata$group=="c")
eegdata=eegdata[idx,]

# temporal smoothing
eegmod=eegsmooth(eegdata$voltage,time=eegdata$time)

# define data for prediction
time=seq(min(eegdata$time),max(eegdata$time),length.out=100)
yhat=predict(eegmod,newdata=time,se.fit=TRUE)

# plot results using eegtime
eegtime(time,yhat[[1]],vcol="blue",voltageSE=yhat[[2]],
        scol="blue",ylim=c(-4,4),main="Pz")



##########   EXAMPLE 2: Spatial   ##########

# get time point 65 (approx 250 ms) of "c" subjects in "eegdata" data
data(eegdata)
idx=which(eegdata$time==65L & eegdata$group=="c")
eegdata=eegdata[idx,]

# remove ears and nose
idx=c(which(eegdata$channel=="X"),which(eegdata$channel=="Y"),which(eegdata$channel=="nd"))
eegdata=eegdata[-idx,]

# match to eeg coordinates
data(eegcoord)
cidx=match(eegdata$channel,rownames(eegcoord))

# spatial smoothing
eegmod=eegsmooth(eegdata$voltage,space=eegcoord[cidx,1:3])

# use dense cap for prediction
```

```
data(eegdense)
space=eegdense[,1:3]
yhat=predict(eegmod,newdata=space)

# plot results using eegspace
eegspace(space,yhat)



##########   EXAMPLE 3: Spatial-Temporal (not run)   ##########

# # get "c" subjects of "eegdata" data
# data(eegdata)
# idx=which(eegdata$group=="c")
# eegdata=eegdata[idx,]

# # remove ears and nose
# idx=c(which(eegdata$channel=="X"),which(eegdata$channel=="Y"),which(eegdata$channel=="nd"))
# eegdata=eegdata[-idx,]

# # match to eeg coordinates
# data(eegcoord)
# cidx=match(eegdata$channel,rownames(eegcoord))

# # spatial-temporal smoothing
# eegmod=eegsmooth(eegdata$voltage,space=eegcoord[cidx,1:3],time=eegdata$time)

# # time main effect
# newdata=list(time=seq(min(eegdata$time),max(eegdata$time),length.out=100))
# yhat=predict(eegmod,newdata=newdata,se.fit=TRUE,include="time")
# eegtime(newdata$time,yhat[[1]],vcol="blue",voltageSE=yhat[[2]],
#         scol="blue",ylim=c(-2,4),main="Time Main Effect")

# # space main effect
# data(eegdense)
# newdata=list(space=eegdense[,1:3])
# yhat=predict(eegmod,newdata=newdata,include="space")
# eegspace(newdata$space,yhat)

# # interaction effect (spatial map at time point 65)
# data(eegdense)
# newdata=list(space=eegdense[,1:3],time=rep(65,nrow(eegdense)))
# yhat=predict(eegmod,newdata=newdata,include="space:time")
# eegspace(newdata$space,yhat)

# # full prediction (spatial map at time point 65)
# newdata=list(space=eegdense[,1:3],time=rep(65,nrow(eegdense)))
# yhat=predict(eegmod,newdata=newdata)
# eegspace(newdata$space,yhat)
```

eegspace *Plots multi-channel EEG spatial map*

Description

Creates plot of multi-channel electroencephalography (EEG) spatial map. User can control the plot type (2d or 3d), the colormap, color, etc.

Usage

```
eegspace(space,voltage,vlim=NULL,mycolors=NULL,ncolor=100,
         colorbar=TRUE,nctick=5,rtick=1,cex.axis=1,barloc=NULL,
         colorlab=NULL,cex.lab=1,plotaxes=FALSE,main="",
         xyzlab=NULL,cex.point=1,cex.main=1,nose=TRUE,ears=TRUE,
         head=TRUE,col.head="AntiqueWhite",mar=NULL,...)
```

Arguments

space	Matrix of input electrode coordinates (3d or 2d).
voltage	Vector of recorded EEG voltage at each row in space.
vlim	Two-element vector giving the limits to use when mapping voltage to colors in mycolors. Default is vlim=range(voltage).
mycolors	Character vector of colors to use for color mapping (such that length(mycolors)<=ncolor). Default uses a reverse rainbow color scheme: mycolors=rev(rainbow(ncolor,end=3/4)).
ncolor	Number of colors to use in mapping (positive integer).
colorbar	If TRUE, colorbar is plotted.
nctick	Approximate number of ticks for colorbar. Ignored if colorbar=FALSE.
rtick	Round tick labels to given decimal. Ignored if colorbar=FALSE.
cex.axis	Cex of axis ticks for colorbar. Ignored if colorbar=FALSE.
barloc	Character vector giving location of color bar. See Notes.
colorlab	Character vector giving label for color bar. Ignored if colorbar=FALSE.
cex.lab	Cex of axis labels for colorbar. Ignored if colorbar=FALSE.
plotaxes	If TRUE, axes labels are plotted. Ignored for 3d plots.
main	Plot title. Default is no title.
xyzlab	Axis labels to use for plot. If type="2d", then xyzlab should be two-element character vector giving x and y axis labels. If type="3d", then xyzlab should be three-element character vector giving x, y, and z axis labels.
cex.point	Cex for plotted electrodes.
cex.main	Cex for plot title. Ignored if main=NULL.
nose	If TRUE, triangle is plotted to represent the subject's nose. Ignored if ncol(space)==3.
ears	If TRUE, ovals are plotted to represent the subject's ears. Ignored if ncol(space)==3.

head	If TRUE, head is plotted. Ignored if type="2d".
col.head	Color for dummy head in 3d plot. Ignored if type="2d".
mar	Margins to use for plot (see par).
...	Optional inputs for plot or lines function.

Value

Produces plot of EEG spatial map with NULL return value.

Note

For 3d plots, barloc can be one of four options: "backright", "backleft", "frontright", or "frontleft". For 2d plots, barloc can be either "right" or "left".

Currently supports spatial maps registered to the 84-channel cap produced by eegcap and eegcoord.

Author(s)

Nathaniel E. Helwig <helwig@umn.edu>

References

Bache, K. & Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Begleiter, H. *Neurodynamics Laboratory*. State University of New York Health Center at Brooklyn.

Ingber, L. (1997). Statistical mechanics of neocortical interactions: Canonical momenta indicatros of electroencephalography. *Physical Review E, 55*, 4578-4593.

Ingber, L. (1998). Statistical mechanics of neocortical interactions: Training and testing canonical momenta indicators of EEG. *Mathematical Computer Modelling, 27*, 33-64.

Examples

```
##########   EXAMPLE   ##########

# get time point 65 (approx 250 ms) from "eegdata" data
data(eegdata)
idx=which(eegdata$time==65L)
eegdata=eegdata[idx,]

# get average spatial map
eegmean=tapply(eegdata$voltage,list(eegdata$channel,eegdata$group),mean)

# remove ears and nose
acnames=rownames(eegmean)
idx=c(which(acnames=="X"),which(acnames=="Y"),which(acnames=="nd"))
eegmean=eegmean[-idx,]

# match to eeg coordinates
data(eegcoord)
cidx=match(rownames(eegmean),rownames(eegcoord))
```

```
# plot average control voltage in 3d
open3d()
eegspace(eegcoord[cidx,1:3],eegmean[,2])

# plot average control voltage in 2d
eegspace(eegcoord[cidx,4:5],eegmean[,2])

# change 3d bar location and use play3d to rotate (not run)
open3d()
par3d(windowRect=c(0,0,600,600))
eegspace(eegcoord[cidx,1:3],eegmean[,2],barloc="frontleft")
#play3d(spin3d(axis=c(0,0,1),rpm=5),duration=20)

# change 2d bar location
eegspace(eegcoord[cidx,4:5],eegmean[,2],barloc="left")
```

`eegtime` *Plots single-channel EEG time course*

Description

Creates plot of single-channel electroencephalography (EEG) time course with optional confidence interval. User can control the plot orientation, line types, line colors, etc.

Usage

```
eegtime(time,voltage,flipvoltage=TRUE,vlty=1,vlwd=2,vcol="black",
        voltageSE=NULL,slty=3,slwd=1,scol="black",conflevel=0.95,
        plotzero=TRUE,zlty=1,zlwd=0.5,zcol="black",xlim=NULL,ylim=NULL,
        xlab=NULL,ylab=NULL,nxtick=6,nytick=6,add=FALSE,...)
```

Arguments

`time`	Vector of time points at which EEG was recorded.
`voltage`	Vector of recorded EEG voltage at each point in `time`.
`flipvoltage`	If `TRUE`, negative voltages are plotted upwards.
`vlty`	Line type for `voltage`.
`vlwd`	Line width for `voltage`.
`vcol`	Line color for `voltage`.
`voltageSE`	Vector of standard errors of EEG voltage at each point in `time`.
`slty`	Line type for `voltageSE`.
`slwd`	Line width for `voltageSE`.
`scol`	Line color for `voltageSE`.

conflevel	Confidence level to use for confidence intervals. Default forms 95% CI.
plotzero	If TRUE, horizontal reference line is plotted at 0 volts.
zlty	Line type for reference line. Ignored if plotzero=FALSE.
zlwd	Line width for reference line. Ignored if plotzero=FALSE.
zcol	Line color for reference line. Ignored if plotzero=FALSE.
xlim	Plot limits for time.
ylim	Plot limits for voltage.
xlab	Plot label for time.
ylab	Plot label for voltage.
nxtick	Approximate number of axis ticks for time.
nytick	Approximate number of axis ticks voltage.
add	If TRUE, lines are added to current plot; otherwise a new plot is created.
...	Optional inputs for plot or lines function.

Value

Produces plot of EEG time course with NULL return value.

Note

Confidence intervals are formed using the normal (Gaussian) distribution.

Author(s)

Nathaniel E. Helwig <helwig@umn.edu>

References

Bache, K. & Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Begleiter, H. *Neurodynamics Laboratory*. State University of New York Health Center at Brooklyn.

Ingber, L. (1997). Statistical mechanics of neocortical interactions: Canonical momenta indicatros of electroencephalography. *Physical Review E, 55*, 4578-4593.

Ingber, L. (1998). Statistical mechanics of neocortical interactions: Training and testing canonical momenta indicators of EEG. *Mathematical Computer Modelling, 27*, 33-64.

Examples

```
##########   EXAMPLE   ##########

# get "PZ" electrode from "eegdata" data
data(eegdata)
idx=which(eegdata$channel=="PZ")
eegdata=eegdata[idx,]

# get average and standard error (note se=sd/sqrt(n))
```

```
eegmean=tapply(eegdata$voltage,list(eegdata$time,eegdata$group),mean)
eegse=tapply(eegdata$voltage,list(eegdata$time,eegdata$group),sd)/sqrt(50)

# plot results with legend
tseq=(0:255)*1000/255
eegtime(tseq,eegmean[,2],vcol="blue",voltageSE=eegse[,2],
        scol="blue",ylim=c(-10,6),main="Pz")
eegtime(tseq,eegmean[,1],vlty=2,vcol="red",voltageSE=eegse[,1],
        scol="red",add=TRUE)
legend("bottomright",c("controls","alcoholics"),lty=c(1,2),
       lwd=c(2,2),col=c("blue","red"),bty="n")
```

Index

∗Topic **datasets**
eegcoord, 6
eegdense, 7
eeghead, 8
eegmesh, 11
∗Topic **package**
eegkit-package, 2

bigspline, *12, 13*
bigssa, *12, 13*
bigtps, *12, 13*

eegcap, *2*, 3, *8*, *17*
eegcoord, *3, 4*, 6, *7*, *17*
eegdense, 7, *12*
eeghead, 8
eegica, *2*, 9
eegkit *(*eegkit-package*)*, 2
eegkit-package, 2
eegmesh, 11
eegsmooth, *2*, 12
eegspace, *2*, *8*, *12*, 16
eegtime, *2*, 18

icafast, *9*
icaimax, *9*
icajade, *9*

plot3d, *3*

rgl.postscript, *4*