Package ‘BayesBrainMap’

May 6, 2026

Type Package

Title Estimate Brain Networks and Connectivity with Population-Derived Priors

Version 0.2.0

Maintainer Amanda Mejia <`mandy.mejia@gmail.com`>

Description Implements Bayesian brain mapping with population-derived priors, including the original model described in Mejia et al. (2020) <doi:10.1080/01621459.2019.1679638>, the model with spatial priors described in Mejia et al. (2022) <doi:10.1080/10618600.2022.2104289>, and the model with population-derived priors on functional connectivity described in Mejia et al. (2025) <doi:10.1093/biostatistics/kxaf022>. Population-derived priors are based on templates representing established brain network maps, for example derived from independent component analysis (ICA), parcellations, or other methods. Model estimation is based on expectation-maximization or variational Bayes algorithms. Includes direct support for 'CIFTI', 'GIFTI', and 'NIFTI' neuroimaging file formats.

Depends R (>= 3.6.0)

License GPL-3

Additional_repositories `https://inla.r-inla-download.org/R/testing`

Encoding UTF-8

Imports abind, fMRItools (>= 0.7.1), fMRIscrub (>= 0.14.5), foreach, Matrix, matrixStats, methods, pesel, SQUAREM, stats, utils

Suggests ciftiTools (>= 0.13.2), excursions, RNifti, oro.nifti, gifti, ggplot2, parallel, doParallel, knitr, rmarkdown, INLA, testthat (>= 3.0.0)

RoxygenNote 7.3.3

URL `https://github.com/mandymejia/BayesBrainMap`

BugReports `https://github.com/mandymejia/BayesBrainMap/issues`

NeedsCompilation no

Author Amanda Mejia [aut, cre],
Damon Pham [aut] (ORCID: <https://orcid.org/0000-0001-7563-4727>),
Nohelia Da Silva [ctb]

Repository CRAN

Date/Publication 2026-02-03 09:00:14 UTC

Contents

`bdiag_m2` *Bdiag m2*

Description

Bdiag m2

Usage

```
bdiag_m2(mat, N)
```

Arguments

`mat`	a k x k 'matrix'
`N`	how many times to repeat `mat`

Value

a sparse (N*k x N*k) matrix of class `"dgCMatrix"`.

`BrainMap` *BrainMap*

Description

Deprecated: This function is deprecated. Use `fit_BBM` instead.

Usage

```
BrainMap(...)
```

Arguments

`...`	See `fit_BBM`

`Chol_samp_fun` *Cholesky-based FC sampling*

Description

Cholesky-based FC sampling

Usage

```
Chol_samp_fun(Chol_vals, p, M, chol_diag, chol_offdiag, Chol_mat_blank)
```

Arguments

`Chol_vals`	Matrix of Cholesky factorizations (upper triangular values) for all prior sessions (nN*nM x nChol)
`p`	Pivot/reordering applied to FC matrices prior to Cholesky factorization
`M`	Number of samples to draw
`chol_diag`	Indices of diagonal upper triangular elements
`chol_offdiag`	Indices of off-diagonal upper triangular elements
`Chol_mat_blank`	A nLxnL matrix indexing the upper triangular elements

`EM_BBM.spatial` *EM Algorithms for Bayesian brain maps*

Description

EM Algorithms for Bayesian brain maps

Usage

```
EM_BBM.spatial(
  prior_mean,
  prior_var,
  meshes,
  BOLD,
  theta0,
  C_diag,
  H,
  Hinv,
  maxiter = 100,
  usePar = FALSE,
  epsilon = 0.001,
  reduce_dim = TRUE,
  verbose = FALSE
```

```
)

EM_BBM.independent(
  prior_mean,
  prior_var,
  BOLD,
  theta0,
  C_diag,
  H,
  Hinv,
  maxiter = 100,
  epsilon = 0.001,
  reduce_dim = FALSE,
  usePar = FALSE,
  verbose
)
```

Arguments

`prior_mean`	($V \times Q$ matrix) mean maps for each network in the prior, where Q is the number of networks, $V = nvox$ is the number of data locations.
`prior_var`	($V \times Q$ matrix) between-subject variance maps for each network in prior
`meshes`	`NULL` for spatial independence model, otherwise a list of objects of class "BBM_mesh" containing the triangular mesh (see `make_mesh`) for each brain structure.
`BOLD`	($V \times Q$ matrix) dimension-reduced fMRI data
`theta0`	(list) initial guess at parameter values: A (QxQ mixing matrix), nu0_sq (residual variance from first level) and (for spatial model only) kappa (SPDE smoothness parameter for each network map)
`C_diag`	($Qx1$) diagonal elements of matrix proportional to residual variance.
`H`, `Hinv`	For dimension reduction of the spatial Bayesian brain map model, which assumes that all networks have the same smoothness parameter, κ
`maxiter`	Maximum number of EM iterations. Default: 100.
`usePar`	Parallelize the computation? Default: `FALSE`. Can be the number of cores to use or `TRUE`, which will use the number available minus two. Not yet implemented for spatial Bayesian brain map.
`epsilon`	Smallest proportion change between iterations. Default: 0.001.
`reduce_dim`	Reduce the temporal dimension of the data using PCA? Default: `TRUE` for the spatial EM algorithm, and `FALSE` for the independent EM algorithm.
`verbose`	If `TRUE`, display progress of algorithm. Default: `FALSE`.

Details

`EM_BBM.spatial` implements the expectation-maximization (EM) algorithm described in Mejia et al. (2019+) for estimating the subject-level networks and unknown parameters in the Bayesian brain map model with spatial priors on subject effects.

In both models, if original fMRI timeseries has covariance $\sigma^2 I_T$, the prewhitened timeseries achieved by premultiplying by (QxT) matrix H from PCA has diagonal covariance $\sigma^2 HH'$, so C_diag is $diag(HH')$.

Value

A list: theta (list of final parameter estimates), subjNet_mean (estimates of subject-level networks), subjNet_var (variance of subject-level networks, for non-spatial model) or subjNet_cov (covariance matrix of subject-level networks, for spatial model – note that only diagonal and values for neighbors are computed), and success (flag indicating convergence (`TRUE`) or not (`FALSE`))

`engagements` *engagements*

Description

Deprecated: This function is deprecated. Use `id_engagements` instead.

Usage

```
engagements(...)
```

Arguments

`...`	See `id_engagements`

`estimate.ESS` *Estimation of effective sample size*

Description

Estimation of effective sample size

Usage

```
estimate.ESS(mesh, Y, ind = NULL, trace = FALSE)
```

Arguments

`mesh`	INLA mesh
`Y`	data
`ind`	index of the data locations in the mesh
`trace`	If `TRUE`, a formula based on the trace is used, otherwise the inverse correlation is used

Details

The functions computes the effective sample size as $trace(Q^{-1})/trace(Q * Q)$ as in Bretherton et al. (1999), Journal of Climate.

Value

Estimate of the effective sample size

`estimate_nu` *Universally estimate IW dof parameter nu based on method of moments, so that no empirical variance is under-estimated*

Description

Universally estimate IW dof parameter nu based on method of moments, so that no empirical variance is under-estimated

Usage

```
estimate_nu(var_FC, mean_FC)
```

Arguments

`var_FC`	Empirical between-subject variance of covariance matrices (QxQ)
`mean_FC`	Empirical mean of covariance matrices (QxQ)

Value

estimate for nu

`estimate_nu_matrix` *Estimate IW dof parameter nu based on method of moments*

Description

Estimate IW dof parameter nu based on method of moments

Usage

```
estimate_nu_matrix(var_FC, mean_FC)
```

Arguments

`var_FC`	Empirical between-subject variance of covariance matrices (QxQ)
`mean_FC`	Empirical mean of covariance matrices (QxQ)

Value

QxQ matrix of estimates for nu

`estimate_prior` *Estimate prior*

Description

Estimate prior for Bayesian brain mapping based on fMRI data

Usage

```
estimate_prior(
  BOLD,
  BOLD2 = NULL,
  template,
  mask = NULL,
  inds = NULL,
  scale = c("local", "global", "none"),
  scale_sm_surfL = NULL,
  scale_sm_surfR = NULL,
  scale_sm_FWHM = 2,
  nuisance = NULL,
  scrub = NULL,
  drop_first = 0,
  hpf = 0,
  TR = NULL,
  GSR = FALSE,
  Q2 = 0,
  Q2_max = NULL,
  covariates = NULL,
  brainstructures = "all",
  resamp_res = NULL,
  keep_S = FALSE,
  keep_FC = FALSE,
  FC = TRUE,
  FC_nPivots = 100,
  FC_nSamp = 50000,
  FC_updateA = FALSE,
  varTol = 1e-06,
  maskTol = 0.1,
  missingTol = 0.1,
  usePar = FALSE,
  wb_path = NULL,
  verbose = TRUE
)
```

```
estimate_prior.cifti(
  BOLD,
  BOLD2 = NULL,
  template,
  inds = NULL,
  scale = c("local", "global", "none"),
  scale_sm_surfL = NULL,
  scale_sm_surfR = NULL,
  scale_sm_FWHM = 2,
  nuisance = NULL,
  scrub = NULL,
  drop_first = 0,
  hpf = 0,
  TR = NULL,
  GSR = FALSE,
  Q2 = 0,
  Q2_max = NULL,
  brainstructures = "all",
  resamp_res = resamp_res,
  keep_S = FALSE,
  keep_FC = FALSE,
  FC = TRUE,
  varTol = 1e-06,
  maskTol = 0.1,
  missingTol = 0.1,
  usePar = FALSE,
  wb_path = NULL,
  verbose = TRUE
)

estimate_prior.gifti(
  BOLD,
  BOLD2 = NULL,
  template,
  inds = NULL,
  scale = c("local", "global", "none"),
  scale_sm_surfL = NULL,
  scale_sm_surfR = NULL,
  scale_sm_FWHM = 2,
  nuisance = NULL,
  scrub = NULL,
  drop_first = 0,
  hpf = 0,
  TR = NULL,
  GSR = FALSE,
  Q2 = 0,
  Q2_max = NULL,
```

```
  brainstructures = "all",
  keep_S = FALSE,
  keep_FC = FALSE,
  FC = TRUE,
  varTol = 1e-06,
  maskTol = 0.1,
  missingTol = 0.1,
  usePar = FALSE,
  wb_path = NULL,
  verbose = TRUE
)

estimate_prior.nifti(
  BOLD,
  BOLD2 = NULL,
  template,
  inds = NULL,
  scale = c("local", "global", "none"),
  nuisance = NULL,
  scrub = NULL,
  drop_first = 0,
  hpf = 0,
  TR = NULL,
  GSR = FALSE,
  Q2 = 0,
  Q2_max = NULL,
  mask = NULL,
  keep_S = FALSE,
  keep_FC = FALSE,
  FC = TRUE,
  varTol = 1e-06,
  maskTol = 0.1,
  missingTol = 0.1,
  usePar = FALSE,
  wb_path = NULL,
  verbose = TRUE
)
```

Arguments

`BOLD, BOLD2`	Vector of subject-level fMRI data in one of the following formats: CIFTI file paths, `"xifti"` objects, GIFTI file paths, `"gifti"` objects, NIFTI file paths, `"nifti"` objects, or $V \times T$ numeric matrices, where V is the number of data locations and T is the number of timepoints. If `BOLD2` is provided it must be in the same format as `BOLD`; `BOLD` will be the test data and `BOLD2` will be the retest data. `BOLD2` should be the same length as `BOLD` and have the same subjects in the same order. If `BOLD2` is not provided, `BOLD` will be split in half; the first half will be the test data and the second half will be the retest data.

`template`	Group-level template: either a group ICA (GICA), or a parcellation. A GICA should be provided as a format compatible with `BOLD`, or a (vectorized) numeric matrix ($V \times Q$). Its columns will be centered. A parcellation must be in CIFTI format for use with CIFTI BOLD data (other formats to be implemented in the future). The parcellation should have the same locations as the BOLD and one column, with integer values indicating the parcel to which each location belongs to. Each parcel is modeled as a brain map; instead of the first step of dual regression, the medial timecourse of each parcel is used.
`mask`	Required if `BOLD` are NIFTI file paths or `"nifti"` objects, and optional for other formats. For NIFTI data, this is a logical array of the same spatial dimensions as the fMRI data, with `TRUE` corresponding to in-mask voxels. For other data, this is a logical vector with the same length as the number of locations in `template`, with `TRUE` corresponding to in-mask locations.
`inds`	Numeric indices of the networks in `template` to include in the prior. If `NULL`, use all of the original networks (default). If `inds` is provided, the networks not included will be removed after calculating dual regression, not before. This is because removing the networks prior to dual regression would leave unmodelled signals in the data, which could bias the priors.
`scale`	`"local"` (default), `"global"`, or `"none"`. Local scaling will divide each data location's time series by its estimated standard deviation. Global scaling will divide the entire data matrix by the mean image standard deviation (`mean(sqrt(rowVars(BOLD)))`).
`scale_sm_surfL`, `scale_sm_surfR`, `scale_sm_FWHM`	Only applies if `scale=="local"` and `BOLD` represents surface data (CIFTI or GIFTI). To smooth the standard deviation estimates used for local scaling, provide the surface geometries along which to smooth as GIFTI geometry files or `"surf"` objects, as well as the smoothing FWHM (default: `2`). If `scale_sm_FWHM==0`, no smoothing of the local standard deviation estimates will be performed. If `scale_sm_FWHM>0` but `scale_sm_surfL` and `scale_sm_surfR` are not provided, the default inflated surfaces from the HCP will be used. To create a `"surf"` object from data, see `make_surf`. The surfaces must be in the same resolution as the `BOLD` data.
`nuisance`	(Optional) Nuisance matrices to regress from the BOLD data. Should be a list of matrices, with time along the rows and nuisance signals along the columns, where each entry corresponds to a `BOLD` session; or, if `BOLD2` is provided, should be a length-2 list of such lists with the first sublist corresponding to `BOLD` and the second sublist corresponding to `BOLD2`. If `NULL`, do not remove any nuisance signals. Nuisance regression is performed in a simultaneous regression with any spike regressors from `scrub` and DCT bases from `hpf`. Note that the nuisance matrices should be provided with timepoints matching the original `BOLD` and `BOLD2` irregardless of `drop_first`. Nuisance matrices will be truncated automatically if `drop_first>0`.

`scrub`	(Optional) Numeric vectors of integers giving the indices of volumes to scrub from the BOLD data. (List the volumes to remove, not the ones to keep.) Should be a list of such vectors, where each entry corresponds to a `BOLD` session; or, if `BOLD2` is provided, should be a length-2 list of such lists with the first sublist corresponding to `BOLD` and the second sublist corresponding to `BOLD2`. If `NULL` (default), do not scrub. Scrubbing is performed within a nuisance regression by adding a spike regressor to the nuisance design matrix for each volume to scrub. Note that indices are counted beginning with the first index in the `BOLD` session irregardless of `drop_first`. The indices will be adjusted automatically if `drop_first>0`.
`drop_first`	(Optional) Number of volumes to drop from the start of each BOLD session. Default: `0`.
`hpf`	The frequency at which to apply a highpass filter to the data during pre-processing, in Hertz. Default: `0` Hz (disabled). If the data has not already been highpass filtered, a recommended filter value is `.01` Hz. The highpass filter serves to detrend the data, since low-frequency variance is associated with noise. Highpass filtering is accomplished by nuisance regression of discrete cosine transform (DCT) bases. Note the `TR` argument is required for highpass filtering. If `TR` is not provided, `hpf` will be ignored.
`TR`	The temporal resolution of the data, i.e. the time between volumes, in seconds. `TR` is required for detrending with `hpf`.
`GSR`	Center BOLD across columns (each image)? This is equivalent to performing global signal regression. Default: `FALSE`.
`Q2, Q2_max`	Obtain dual regression estimates after denoising? Denoising is based on modeling and removing nuisance ICs. It may result in a cleaner estimate for smaller datasets, but it may be unnecessary (and time-consuming) for larger datasets. Set `Q2` to control denoising: use a positive integer to specify the number of nuisance ICs, `NULL` to have the number of nuisance ICs estimated by PESEL, or zero (default) to skip denoising. If `is.null(Q2)`, use `Q2_max` to specify the maximum number of nuisance ICs that should be estimated by PESEL. `Q2_max` must be less than $T * .75 - Q$ where T is the minimum number of timepoints in each fMRI scan and Q is the number of networks in the `template`. If `NULL` (default), `Q2_max` will be set to $T * .50 - Q$, rounded.
`covariates`	Subjects by variables numeric matrix of covariates to take into account for model estimation. Column names should give the name of each variable. Default: `NULL` (no covariates). NOTE: Not implemented yet.
`brainstructures`	Only applies if the entries of `BOLD` are CIFTI file paths. This is a character vector indicating which brain structure(s) to obtain: `"left"` (left cortical surface), `"right"` (right cortical surface) and/or `"subcortical"` (subcortical and cerebellar gray matter). Can also be `"all"` (obtain all three brain structures). Default: `c("all")`.

`resamp_res`	Only applies if the entries of `BOLD` are CIFTI file paths. Resample the data upon reading it in? Default: `NULL` (no resampling).
`keep_S`	Keep the DR estimates of S? If `FALSE` (default), do not save the DR estimates and only return the priors. If `TRUE`, the DR estimates of S are returned too. If a single file path, save the DR estimates as an RDS file at that location rather than returning them.
`keep_FC`	Keep the DR estimates of the FC cor(A)? If `FALSE` (default), do not save the DR estimates and only return the priors. If `TRUE`, the DR estimates of cor(A) and its Cholesky factor are returned too. If a single file path, save the DR estimates as an RDS file at that location rather than returning them.
`FC`	Include the functional connectivity prior? Default: `TRUE`.
`FC_nPivots`	Number of pivots to use in Cholesky-based FC prior estimation. Set to zero to skip Cholesky-based FC prior estimation. Default: 100.
`FC_nSamp`	Number of FC matrix samples to generate across all pivots. This should be a multiple of FC_nPivots.
`FC_updateA`	Update the timecourses before computing FC? Default: `FALSE`. Only applies if `FC`.
`varTol`	Tolerance for variance of each data location. For each scan, locations which do not meet this threshold are masked out of the analysis. Default: `1e-6`. Variance is calculated on the original data, before any normalization.
`maskTol`	For computing the dual regression results for each subject: tolerance for number of locations masked out due to low variance or missing values. If more than this many locations are masked out, a subject is skipped without calculating dual regression. `maskTol` can be specified either as a proportion of the number of locations (between zero and one), or as a number of locations (integers greater than one). Default: `.1`, i.e. up to 10 percent of locations can be masked out. If `BOLD2` is provided, masks are calculated for both scans and then the intersection of the masks is used, for each subject.
`missingTol`	For computing the variance decomposition across all subjects: tolerance for number of subjects masked out due to low variance or missing values at a given location. If more than this many subjects are masked out, the location's value will be `NA` in the priors. `missingTol` can be specified either as a proportion of the number of locations (between zero and one), or as a number of locations (integers greater than one). Default: `.1`, i.e. up to 10 percent of subjects can be masked out at a given location.
`usePar`, `wb_path`	Parallelize the DR computations over subjects? Default: `FALSE`. Can be the number of cores to use or `TRUE`, which will use the number on the PC minus two. If the input data is in CIFTI format, the `wb_path` must also be provided.
`verbose`	Display progress updates? Default: `TRUE`.

Details

All fMRI data (entries in `BOLD` and `BOLD2`, and `template`) must be in the same spatial resolution.

Value

A list: the `prior` and `var_decomp` with entries in matrix format; the `mask` of locations without prior values due to too many low variance or missing values; the function `params` such as the type of scaling and detrending performed; the `dat_struct` which can be used to convert `prior` and `var_decomp` to `"xifti"` or `"nifti"` objects if the `BOLD` format was CIFTI or NIFTI data; and DR results if `isTRUE(keep_S)` and/or `isTRUE(keep_FC)`.

Use `summary` to print a description of the prior results, and for CIFTI-format data use `plot` to plot the prior mean and variance estimates. Use `export_prior` to save the priors to individual RDS, CIFTI, or NIFTI files (depending on the `BOLD` format).

Examples

```
nT <- 21
nV <- 140
nQ <- 6
mU <- matrix(rnorm(nV*nQ), nrow=nV)
mS <- mU %*% diag(seq(nQ, 1)) %*% matrix(rnorm(nQ*nT), nrow=nQ)
BOLD <- list(B1=mS, B2=mS, B3=mS)
BOLD <- lapply(BOLD, function(x){x + rnorm(nV*nT, sd=.05)})
template <- mU
estimate_prior(BOLD=BOLD, template=mU, FC_nSamp=2000, usePar=FALSE)

## Not run:
 estimate_prior(
   run1_cifti_fnames, run2_cifti_fnames,
   gICA_cifti_fname, brainstructures="all",
   scale="global", TR=0.71, Q2=NULL, varTol=10,
   usePar=FALSE
 )

## End(Not run)
```

`export_prior` *Export prior*

Description

Export the priors (mean, variance, and FC) as separate files for visualization or processing outside of `BayesBrainMap`.

Usage

```
export_prior(x, out_fname = NULL, var_method = c("non-negative", "unbiased"))
```

Arguments

`x`	The result of `estimate_prior`
`out_fname`	Use `NULL` (default) to just return the prior objects directly. Otherwise, use a character vector of length 3 or 4 of file path(s) to save the output to: the mean prior, the variance prior, the variance decomposition, and the FC prior if present, in that order. If one file name is provided, it will be appended with `"_mean.[file_ext]"` for the prior mean map, `"_var.[file_ext]"` for the prior variance map, `"_varDecomp.rds"` for the variance decomposition, and `"_FC.rds"` where `[file_ext]` will be `"dscalar.nii"` for CIFTI input, `"nii"` for NIFTI input, and `"rds"` for data input.
`var_method`	`"non-negative"` (default) or `"unbiased"`

Value

If `is.null(out_fname)`, the priors in data matrix, `"xifti"`, or `"nifti"` format, to match the format of the original BOLD data. Otherwise, the paths to the new files specified by `out_fname`. If prior includes functional connectivity components, the FC prior and its mean and variance will be included.

Examples

```
## Not run:
 tm <- estimate_prior(cii1_fnames, cii2_fnames, gICA_fname, usePar=FALSE)
 export_prior(tm, out_fname="my_prior", var_method="unbiased")

## End(Not run)
```

`fit_BBM` *Bayesian brain mapping*

Description

Fit Bayesian brain mapping model using variational Bayes (VB) or expectation-maximization (EM).

Usage

```
fit_BBM(
  BOLD,
  prior,
  var_method = c("non-negative", "unbiased"),
  scale = c("prior", "global", "local", "none"),
  scale_sm_surfL = NULL,
  scale_sm_surfR = NULL,
  scale_sm_FWHM = "prior",
  nuisance = NULL,
  scrub = NULL,
```

```
  drop_first = 0,
  hpf = "prior",
  TR = NULL,
  GSR = "prior",
  Q2 = "prior",
  Q2_max = "prior",
  covariates = NULL,
  brainstructures = "prior",
  mask = NULL,
  varTol = "prior",
  spatial_model = NULL,
  resamp_res = NULL,
  rm_mwall = TRUE,
  reduce_dim = FALSE,
  method_FC = c("VB1", "VB2", "none"),
  maxiter = 100,
  miniter = 3,
  epsilon = 0.001,
  kappa_init = 0.2,
  usePar = TRUE,
  PW = FALSE,
  seed = 1234,
  verbose = TRUE
)
```

Arguments

`BOLD`	Vector of subject-level fMRI data in one of the following formats: CIFTI file paths, `"xifti"` objects, NIFTI file paths, `"nifti"` objects, or $V \times T$ numeric matrices, where V is the number of data locations and T is the number of timepoints. If multiple BOLD data are provided, they will be independently centered, scaled, detrended (if applicable), and denoised (if applicable). Then they will be concatenated together followed by computing the initial dual regression estimate.
`prior`	Prior estimates in a format compatible with `BOLD`, from `estimate_prior`.
`var_method`	Which calculation of the prior variance to use: `"non-negative"` (default) or `"unbiased"`. The unbiased prior variance is based on the assumed mixed effects/ANOVA model, whereas the non-negative prior variance adds to it to account for greater potential between-subjects variation. (The prior mean is the same for either choice of `var_method`.)
`scale`	`"global"`, `"local"`, or `"none"`. Global scaling will divide the entire data matrix by the mean image standard deviation (`mean(sqrt(rowVars(BOLD)))`). Local scaling will divide each data location's time series by its estimated standard deviation. Default: `"prior"`, to use the same option used for estimation of the `prior`.
`scale_sm_surfL`, `scale_sm_surfR`, `scale_sm_FWHM`	Only applies if `scale=="local"` and `BOLD` represents CIFTI-format data. To smooth the standard deviation estimates used for local scaling, provide the sur-

	face geometries along which to smooth as GIFTI geometry files or `"surf"` objects, as well as the smoothing FWHM (default: `"prior"` to use the same option used for estimation of the `prior`). If `scale_sm_FWHM==0`, no smoothing of the local standard deviation estimates will be performed. If `scale_sm_FWHM>0` but `scale_sm_surfL` and `scale_sm_surfR` are not provided, the default inflated surfaces from the HCP will be used. To create a `"surf"` object from data, see `make_surf`. The surfaces must be in the same resolution as the `BOLD` data.
`nuisance`	(Optional) Signals to regress from the data, given as a numeric matrix with the same number of rows as there are volumes in the `BOLD` data. If multiple `BOLD` sessions are provided, this argument can be a list to use different nuisance regressors for different sessions. Nuisance regression is performed as a first step, before centering, scaling, and denoising. An intercept column will automatically be added to `nuisance`. If `NULL`, no extra nuisance signals will be regressed from the data, but a nuisance regression will still be used if warranted by `scrub` or `hpf`.
`scrub`	(Optional) A numeric vector of integers giving the indices of volumes to scrub from the BOLD data. (List the volumes to remove, not the ones to keep.) If multiple `BOLD` sessions are provided, this argument can be a list to remove different volumes for different sessions. Scrubbing is performed within nuisance regression by adding a spike regressor to the nuisance design matrix for each volume to scrub. If `NULL` (default), do not scrub.
`drop_first`	(Optional) Number of volumes to drop from the start of each BOLD session. Default: `0`.
`TR`, `hpf`	These arguments control detrending. `TR` is the temporal resolution of the data, i.e. the time between volumes, in seconds; `hpf` is the frequency of the high-pass filter, in Hertz. Detrending is performed via nuisance regression of DCT bases. Default: `"prior"` to use the values from the prior. Set `hpf` to `0` to disable the high-pass filter. Be sure to set the correct `TR` if it's different for the new data compared to the data used in `estimate_prior`. Note that if multiple `BOLD` sessions are provided, their `TR` and `hpf` must be the same; both arguments accept only one value.
`GSR`	Center BOLD across columns (each image)? This is equivalent to performing global signal regression. Default: `"prior"`, to use the same option used for estimation of the `prior`.
`Q2`, `Q2_max`	Denoise the BOLD data? Denoising is based on modeling and removing nuisance ICs. It may result in a cleaner estimate for smaller datasets, but it may be unnecessary (and time-consuming) for larger datasets. Set `Q2` to control denoising: use a positive integer to specify the number of nuisance ICs, `NULL` to have the number of nuisance ICs estimated by PESEL, or zero to skip denoising. If `is.null(Q2)`, use `Q2_max` to specify the maximum number of nuisance ICs that should be estimated by PESEL. `Q2_max` must be less than $T*.75-Q$ where

T is the number of timepoints in BOLD and Q is the number of networks in the prior. If `NULL`, `Q2_max` will be set to $T * .50 - Q$, rounded.

The defaults for both arguments is `"prior"`, to use the same option used for estimation of the `prior`.

`covariates`	Numeric vector of covariates to take into account for model estimation. Names should give the name of each variable. The covariates must match those of the prior. Default: `NULL` (no covariates). NOTE: Not implemented yet.
`brainstructures`	Only applies if the entries of `BOLD` are CIFTI file paths. This is a character vector indicating which brain structure(s) to obtain: `"left"` (left cortical surface), `"right"` (right cortical surface) and/or `"subcortical"` (subcortical and cerebellar gray matter). Can also be `"all"` (obtain all three brain structures). Default: `"prior"` to use the same brainstructures present in the `prior`).
`mask`	Required only if the entries of `BOLD` are NIFTI file paths or `"nifti"` objects. This is a binary array of the same spatial dimensions as the fMRI data, with `TRUE` corresponding to in-mask voxels.
`varTol`	Tolerance for variance of each data location. For each scan, locations which do not meet this threshold are masked out of the analysis. Default: `"prior"` to use the same brainstructures present in the `prior`). Variance is calculated on the original data, before any normalization. Set to `0` to avoid removing locations due to low variance.
`spatial_model`	Should spatial modeling be performed? If `NULL`, assume spatial independence. Otherwise, provide meshes specifying the spatial priors assumed on each independent component. Each should represent a brain structure, between which spatial independence can be assumed. If `BOLD` represents CIFTI-format data, `spatial_model` should give the left and right cortex surface geometries (whichever one(s) are being used) as `"surf"` objects or GIFTI surface geometry file paths. Spatial modeling is not yet available for the subcortex. This argument can also be `TRUE`, in which case spatial modeling will be performed with the surfaces included in the first entry of `BOLD` if it is a `"xifti"` object, or if those are not present available, the default inflated surfaces from `ciftiTools`. If `BOLD` represents NIFTI-format data, spatial modeling is not yet available. If `BOLD` is a numeric matrix, `spatial_model` should be a list of meshes (see `make_mesh`).
`resamp_res`	Only applies if `BOLD` represents CIFTI-format data. The target resolution for resampling (number of cortical surface vertices per hemisphere). For spatial modelling, a value less than 10000 is recommended for computational feasibility. If `NULL` (default), do not perform resampling.
`rm_mwall`	Only applies if `BOLD` represents CIFTI-format data. Should medial wall (missing data) locations be removed from the mesh? If `TRUE`, faster computation but less accurate estimates at the boundary of wall.
`reduce_dim`	Reduce the temporal dimension of the data using PCA? Default: `TRUE`. Skipping dimension reduction will slow the model estimation, but may result in more accurate results. Ignored for FC prior ICA

`method_FC`	Variational Bayes (VB) method for FC prior ICA model: `"VB1"` (default) uses a conjugate Inverse-Wishart prior for the cor(A); `"VB2"` draws samples from p(cor(A)) to emulate the population distribution using a combination of Cholesky, SVD, and random pivoting. `"none"` Uses standard prior ICA without FC prior
`maxiter`	Maximum number of EM or VB iterations. Default: `100`.
`miniter`	Minimum number of EM or VB iterations. Default: `3`.
`epsilon`	Smallest proportion change between iterations. Default: `.001`.
`kappa_init`	Starting value for kappa. Default: `0.2`.
`usePar`	Parallelize the computation? Default: `TRUE`. Can be the number of cores to use or `TRUE`, which will use the number available minus two.
`PW`	Prewhiten to account for residual autocorrelation? Default: `FALSE`. Only affects FC prior ICA models.
`seed`	(Only applicable for FC calculation and if `PW`) Seed to use for computing temporal effective sample size (ESS) to correct sums over t. If `NULL`, do not change the seed. Default: `1234`.
`verbose`	If `TRUE`, display progress of algorithm

Value

A (spatial) prior ICA object, which is a list containing: `subjNet_mean`, the $V \times L$ estimated independent components **S**; `subjNet_se`, the standard errors of **S**; the `mask` of locations without prior values due to too many low variance or missing values; the `nuisance` design matrix or matrices if applicable; and the function `params` such as the type of scaling and detrending performed.

If `BOLD` represented CIFTI or NIFTI data, `subjNet_mean` and `subjNet_se` will be formatted as `"xifti"` or `"nifti"` objects, respectively.

Examples

```
## Not run:
 tm <- estimate_prior(cii1_fnames, cii2_fnames, gICA_fname, usePar=FALSE)
 fit_BBM(newcii_fname, tm, spatial_model=TRUE, resamp_res=2000, usePar=FALSE)

## End(Not run)
```

`id_engagements` *Engagements of (spatial) Bayesian brain mapping*

Description

Identify areas of engagement in each network from the result of (spatial) Bayesian brain mapping.

Usage

```
id_engagements(
  bMap,
  u = NULL,
  z = 2,
  alpha = 0.01,
  type = c(">", "abs >", "<", "!="),
  method_p = c("BH", "bonferroni"),
  verbose = FALSE,
  which.nets = NULL,
  deviation = FALSE
)
```

Arguments

`bMap`	Fitted (spatial) Bayesian brain map from `fit_BBM`.
`u, z`	Set a threshold value for engagement? A threshold value can be specified directly with `u`, or a z-score-like threshold in terms of standard deviations (the SD of values in the mean prior) can be specified with `z`. Only one type of threshold can be used. Default: `z=2`. Set both `u` and `z` to `NULL` to not use a threshold. Either `u` or `z` can also be a vector to test multiple thresholds at once, as long as `type` is not `"!="` (to ensure the engagement regions are successive subsets).
`alpha`	Significance level for hypothesis testing. Default: `0.01`.
`type`	Type of region: `">"` (default), `"abs >"`, `"<"`, or `"!="`. `"abs >"` tests for magnitude by taking the absolute value and then testing if they are greater than... .
`method_p`	If the input is a `"bMap.[format]"` model object, the type of multiple comparisons correction to use for p-values, or `NULL` for no correction. Default: `"BH"` (Benjamini & Hochberg, i.e. the false discovery rate). Another option is `"bonferroni"` correction. See `help(p.adjust)` for the full list of options. Note that multiple comparisons will account for data locations, but not networks.
`verbose`	If `TRUE`, display progress of algorithm. Default: `FALSE`.
`which.nets`	Indices of networks for which to identify engagements. If `NULL` (default), use all networks.
`deviation`	If `TRUE` identify significant deviations from the prior mean, rather than significant areas of engagement. Default: `FALSE`.

Value

A list containing engagement maps for each network, the joint and marginal PPMs for each network, masks of regions meeting the threshold(s) if provided, and the parameters used for computing engagement. If the input represented CIFTI- or NIFTI-format data, then the engagements maps will be formatted accordingly.

Use `summary` to obtain information about the engagements results. For CIFTI-format engagements, use `plot` to visualize the engagement maps.

Examples

```
## Not run:
 id_engagements(fit_BBM_result, alpha=.05, deviation=TRUE)

## End(Not run)
```

`IW_var` *Compute theoretical Inverse-Wishart variance of covariance matrix elements*

Description

Compute theoretical Inverse-Wishart variance of covariance matrix elements

Usage

```
IW_var(nu, p, xbar_ij, xbar_ii, xbar_jj)
```

Arguments

`nu`	Inverse Wishart degrees of freedom parameter
`p`	Matrix dimension for IW distribution
`xbar_ij`	Empirical mean of covariance matrices at element (i,j)
`xbar_ii`	Empirical mean of covariance matrices at the ith diagonal element
`xbar_jj`	Empirical mean of covariance matrices at the jth diagonal element

Value

Theoretical IW variance for covariance element (i,j)

`IW_var_cor` *Compute theoretical Inverse-Wishart variance of correlation matrix elements*

Description

Compute theoretical Inverse-Wishart variance of correlation matrix elements

Usage

```
IW_var_cor(nu, p, xbar_ij)
```

Arguments

`nu`	Inverse Wishart degrees of freedom parameter
`p`	Matrix dimension for IW distribution
`xbar_ij`	Empirical mean of covariance matrices at element (i,j)

Value

Theoretical IW variance for correlation element (i,j)

`lik` *Compute likelihood in SPDE model for ESS estimation*

Description

Compute likelihood in SPDE model for ESS estimation

Usage

```
lik(theta, Y, G, C, ind = NULL)
```

Arguments

`theta`	Value of hyperparameters
`Y`	Data vector
`G`	SPDE G matrix
`C`	SPDE C matrix
`ind`	Indices of data locations in the mesh

Value

Log likelihood value

`plot.bMap.cifti` *Plot fit_BBM estiamte*

Description

Plot fit_BBM estiamte

Usage

```
## S3 method for class 'bMap.cifti'
plot(x, what = c("maps", "FC"), stat = c("mean", "se"), ...)
```

Arguments

`x`	The result of `fit_BBM` with CIFTI data
`what`	The `"maps"` (default) on the brain, or the `"FC"` matrix. If both are desired, use two separate `plot` calls to first plot the maps and then plot the FC. If `"FC"`, the default color scale will be from blue (-1) to red (1). This can be changed with the `colFUN` argument to `plot_FC_gg`.
`stat`	`"mean"` (default) or `"se"`. Note that for the FC, only the mean estimate is available, not the SE.
`...`	Additional arguments to `view_xifti` if `what=="maps"`, or `plot_FC_gg` if `what=="FC"`.

Value

The plot

`plot.bMap.matrix` *Plot prior*

Description

This feature is not supported yet.

Usage

```
## S3 method for class 'bMap.matrix'
plot(x, ...)
```

Arguments

`x`	The result of `fit_BBM` with NIFTI data
`...`	Additional arguments

Value

Nothing, because an error is raised.

`plot.bMap.nifti` *Plot prior*

Description

Plot prior

Usage

```
## S3 method for class 'bMap.nifti'
plot(
  x,
  stat = c("mean", "se"),
  plane = c("axial", "sagittal", "coronal"),
  n_slices = 9,
  slices = NULL,
  ...
)
```

Arguments

`x`	The result of `fit_BBM` with NIFTI data
`stat`	`"mean"` (default), `"se"`
`plane, n_slices, slices`	Anatomical plane and which slice indices to show. Default: 9 axial slices.
`...`	Additional arguments

Value

The plot

`plot.bMap_eng.cifti` *Plot engagements*

Description

Plot engagements

Usage

```
## S3 method for class 'bMap_eng.cifti'
plot(
  x,
  stat = c("engaged", "pvals", "pvals_adj", "tstats", "se", "thresholded"),
  test_level = NULL,
  ...
)
```

Arguments

`x`	The engagements from `id_engagements.cifti`
`stat`	`"engaged"` (default), `"pvals"`, `"pvals_adj"`, `"tstats"`, `"vars"`, or `"thresholded"`.
`test_level`	If `stat` is not `"engaged"` or `thresholded`, which test level should be used? See `dimnames(x$engaged_mat)[[3]]` for the test levels. If `NULL` (default), the first test level will be used.
`...`	Additional arguments to `view_xifti`

Value

The engagements plot

`plot.prior.cifti` *Plot prior*

Description

Plot prior

Usage

```
## S3 method for class 'prior.cifti'
plot(
  x,
  what = c("maps", "FC"),
  stat = c("mean", "sd", "var"),
  var_method = c("non-negative", "unbiased"),
  FC_method = c("empirical", "IW", "Chol", "none"),
  ...
)
```

Arguments

`x`	The prior from `estimate_prior.cifti`
`what`	The `"maps"` (default) on the brain, or the `"FC"` matrix. If both are desired, use two separate `plot` calls to first plot the maps and then plot the FC. If `"FC"`, the default color scale will be from blue (-1) to red (1). This can be changed with the `colFUN` argument to `plot_FC_gg`.
`stat`	Which prior statistic to plot: the `"mean"` (default), `"sd"` for the square root of the variance template, or `"var"` for the variance template.
`var_method`	`"non-negative"` (default) or `"unbiased"`, for the variance estimate of the maps. Note that FC variance estimates are always non-negative.
`FC_method`	If `what=="FC"`: empirical (`"emp"`) (default), Inverse-Wishart (`"IW"`), or Cholesky (`"Chol"`).
`...`	Additional arguments to `view_xifti` if `what=="maps"`, or `plot_FC_gg` if `what=="FC"`.

Value

The plot

`plot.prior.gifti` *Plot prior*

Description

Plot prior

Usage

```
## S3 method for class 'prior.gifti'
plot(
  x,
  what = c("maps", "FC"),
  stat = c("mean", "sd", "var"),
  var_method = c("non-negative", "unbiased"),
  FC_method = c("empirical", "IW", "Chol", "none"),
  ...
)
```

Arguments

`x`	The prior from `estimate_prior.gifti`
`what`	The `"maps"` (default) on the brain, or the `"FC"` matrix. If both are desired, use two separate `plot` calls to first plot the maps and then plot the FC. If `"FC"`, the default color scale will be from blue (-1) to red (1). This can be changed with the `colFUN` argument to `plot_FC_gg`.
`stat`	Which prior statistic to plot: the `"mean"` (default), `"sd"` for the square root of the variance template, or `"var"` for the variance template.
`var_method`	`"non-negative"` (default) or `"unbiased"`, for the variance estimate of the maps. Note that FC variance estimates are always non-negative.
`FC_method`	If `what=="FC"`: empirical (`"empirical"`) (default), Inverse-Wishart (`"IW"`), or Cholesky (`"Chol"`).
`...`	Additional arguments to `view_xifti` if `what=="maps"`, or `plot_FC_gg` if `what=="FC"`.

Value

The plot

`plot.prior.matrix` *Plot prior*

Description

Plot prior

Usage

```
## S3 method for class 'prior.matrix'
plot(x, ...)
```

Arguments

`x`	The prior from `estimate_prior.matrix`
`...`	Additional arguments

Value

The plot

`plot.prior.nifti` *Plot prior*

Description

Based on `oro.nifti::image`.

Usage

```
## S3 method for class 'prior.nifti'
plot(
  x,
  what = c("maps", "FC"),
  stat = c("mean", "sd", "var"),
  FC_method = c("empirical", "IW", "Chol", "none"),
  var_method = c("non-negative", "unbiased"),
  plane = c("axial", "sagittal", "coronal"),
  n_slices = 9,
  slices = NULL,
  ...
)
```

Arguments

`x`	The prior from `estimate_prior.nifti`
`what`	The `"maps"` (default) on the brain, or the `"FC"` matrix. If both are desired, use two separate `plot` calls to first plot the maps and then plot the FC. If `"FC"`, the default color scale will be from blue (-1) to red (1). This can be changed with the `colFUN` argument to `plot_FC_gg`.
`stat`	Which prior statistic to plot: the `"mean"` (default), `"sd"` for the square root of the variance template, or `"var"` for the variance template.
`FC_method`	If `what=="FC"`: empirical (`"empirical"`) (default), Inverse-Wishart (`"IW"`), or Cholesky (`"Chol"`).
`var_method`	`"non-negative"` (default) or `"unbiased"`, for the variance estimate of the maps. Note that FC variance estimates are always non-negative.
`plane`, `n_slices`, `slices`	Anatomical plane and which slice indices to show. Default: 9 axial slices.
`...`	Additional arguments to `oro.nifti::image` if `what=="maps"`, or `plot_FC_gg` if `what=="FC"`.

Details

Consider using `struct_prior` to obtain the 3D volumes to plot with a different viewer function (e.g. from `oro.nifti`) if desired.

Value

The plot

`pw_estimate` *Estimate residual autocorrelation for prewhitening*

Description

Estimate residual autocorrelation for prewhitening

Usage

```
pw_estimate(A, ar_order, aic = FALSE)
```

Arguments

`A`	Estimated A matrix (T x Q)
`ar_order`, `aic`	Order of the AR model used to prewhiten the data at each location. If `!aic` (default), the order will be exactly `ar_order`. If `aic`, the order will be between zero and `ar_order`, as determined by the AIC.

Value

Estimated AR coefficients and residual variance at every vertex

---

`summary.bMap.cifti` *Summarize a* `"bMap.cifti"` *object*

Description

Summary method for class `"bMap.cifti"`

Usage

```
## S3 method for class 'bMap.cifti'
summary(object, ...)

## S3 method for class 'summary.bMap.cifti'
print(x, ...)

## S3 method for class 'bMap.cifti'
print(x, ...)
```

Arguments

`object`	Object of class `"bMap.cifti"`.
`...`	further arguments passed to or from other methods.
`x`	The result of `fit_BBM` with CIFTI data

Value

A list summarizing of the results of the BrainMap analysis.

Nothing, invisibly.

Nothing, invisibly.

`summary.bMap.matrix` *Summarize a* `"bMap.matrix"` *object*

Description

Summary method for class `"bMap.matrix"`

Usage

```
## S3 method for class 'bMap.matrix'
summary(object, ...)

## S3 method for class 'summary.bMap.matrix'
print(x, ...)

## S3 method for class 'bMap.matrix'
print(x, ...)
```

Arguments

`object`	Object of class `"bMap.matrix"`.
`...`	further arguments passed to or from other methods.
`x`	The prior from `estimate_prior.cifti`

Value

A list summarizing of the results of the BrainMap analysis.

Nothing, invisibly.

Nothing, invisibly.

`summary.bMap.nifti` *Summarize a* `"bMap.nifti"` *object*

Description

Summary method for class `"bMap.nifti"`

Usage

```
## S3 method for class 'bMap.nifti'
summary(object, ...)

## S3 method for class 'summary.bMap.nifti'
print(x, ...)

## S3 method for class 'bMap.nifti'
print(x, ...)
```

Arguments

`object`	Object of class `"bMap.nifti"`.
`...`	further arguments passed to or from other methods.
`x`	The prior from `estimate_prior.cifti`

Value

A list summarizing of the results of the BrainMap analysis.

Nothing, invisibly.

Nothing, invisibly.

`summary.bMap_eng.cifti` *Summarize a* "`bMap_eng.cifti`" *object*

Description

Summary method for class "`bMap_eng.cifti`"

Usage

```
## S3 method for class 'bMap_eng.cifti'
summary(object, ...)

## S3 method for class 'summary.bMap_eng.cifti'
print(x, ...)

## S3 method for class 'bMap_eng.cifti'
print(x, ...)
```

Arguments

`object`	Object of class "`bMap_eng.cifti`".
`...`	further arguments passed to or from other methods.
`x`	The engagements from `id_engagements.cifti`

Value

A list summarizing the data and results for the engagements analysis.

Nothing, invisibly.

Nothing, invisibly.

`summary.bMap_eng.matrix` *Summarize a* "`bMap_eng.matrix`" *object*

Description

Summary method for class "`bMap_eng.matrix`"

Usage

```
## S3 method for class 'bMap_eng.matrix'
summary(object, ...)

## S3 method for class 'summary.bMap_eng.matrix'
print(x, ...)

## S3 method for class 'bMap_eng.matrix'
print(x, ...)
```

Arguments

`object`	Object of class "`bMap_eng.matrix`".
`...`	further arguments passed to or from other methods.
`x`	The engagements from `id_engagements`

Value

A list summarizing the data and results for the engagements analysis.

Nothing, invisibly.

Nothing, invisibly.

`summary.bMap_eng.nifti`

Summarize a "`bMap_eng.nifti`" *object*

Description

Summary method for class "`bMap_eng.nifti`"

Usage

```
## S3 method for class 'bMap_eng.nifti'
summary(object, ...)

## S3 method for class 'summary.bMap_eng.nifti'
print(x, ...)

## S3 method for class 'bMap_eng.nifti'
print(x, ...)
```

Arguments

`object`	Object of class "`bMap_eng.nifti`".
`...`	further arguments passed to or from other methods.
`x`	The engagements from `id_engagements`

Value

A list summarizing the data and results for the engagements analysis.

Nothing, invisibly.

Nothing, invisibly.

`summary.prior.cifti` *Summarize a* `"prior.cifti"` *object*

Description

Summary method for class `"prior.cifti"`

Usage

```
## S3 method for class 'prior.cifti'
summary(object, ...)

## S3 method for class 'summary.prior.cifti'
print(x, ...)

## S3 method for class 'prior.cifti'
print(x, ...)
```

Arguments

`object`	Object of class `"prior.cifti"`.
`...`	further arguments passed to or from other methods.
`x`	The prior from `estimate_prior.cifti`

Value

A list summarizing the prior: data dimensions, options used for prior estimation, etc.

Nothing, invisibly.

Nothing, invisibly.

`summary.prior.gifti` *Summarize a* "`prior.gifti`" *object*

Description

Summary method for class "`prior.gifti`"

Usage

```
## S3 method for class 'prior.gifti'
summary(object, ...)

## S3 method for class 'summary.prior.gifti'
print(x, ...)

## S3 method for class 'prior.gifti'
print(x, ...)
```

Arguments

`object`	Object of class "`prior.gifti`".
`...`	further arguments passed to or from other methods.
`x`	The prior from `estimate_prior.gifti`

Value

A list summarizing the prior: data dimensions, options used for prior estimation, etc.

Nothing, invisibly.

Nothing, invisibly.

`summary.prior.matrix` *Summarize a* "`prior.matrix`" *object*

Description

Summary method for class "`prior.matrix`"

Usage

```
## S3 method for class 'prior.matrix'
summary(object, ...)

## S3 method for class 'summary.prior.matrix'
print(x, ...)

## S3 method for class 'prior.matrix'
print(x, ...)
```

Arguments

`object`	Object of class `"prior.matrix"`.
`...`	further arguments passed to or from other methods.
`x`	The prior from `estimate_prior.cifti`

Value

A list summarizing the prior: data dimensions, options used for prior estimation, etc.

Nothing, invisibly.

Nothing, invisibly.

`summary.prior.nifti` *Summarize a* `"prior.nifti"` *object*

Description

Summary method for class `"prior.nifti"`

Usage

```
## S3 method for class 'prior.nifti'
summary(object, ...)

## S3 method for class 'summary.prior.nifti'
print(x, ...)

## S3 method for class 'prior.nifti'
print(x, ...)
```

Arguments

`object`	Object of class `"prior.nifti"`.
`...`	further arguments passed to or from other methods.
`x`	The prior from `estimate_prior.nifti`

Value

A list summarizing the prior: data dimensions, options used for prior estimation, etc.

Nothing, invisibly.

Nothing, invisibly.

`UpdateTheta_BBM` *Parameter Estimates in EM Algorithm for Bayesian brain map*

Description

Parameter Estimates in EM Algorithm for Bayesian brain map

Usage

```
UpdateTheta_BBM(
  prior_mean,
  prior_var,
  meshes,
  BOLD,
  theta,
  C_diag,
  H,
  Hinv,
  s0_vec,
  D,
  Dinv_s0,
  verbose = FALSE,
  return_MAP = FALSE,
  update = c("all", "kappa", "A")
)

UpdateTheta_iBM(
  prior_mean,
  prior_var,
  BOLD,
  theta,
  C_diag,
  H,
  Hinv,
  update_nu0sq = TRUE,
  return_MAP = FALSE,
  verbose = TRUE
)
```

Arguments

`prior_mean`	($V \times Q$ matrix) mean maps for each network in prior
`prior_var`	($V \times Q$ matrix) between-subject variance maps for each network in prior
`meshes`	`NULL` for spatial independence model, otherwise a list of objects of class "BBM_mesh" containing the triangular mesh (see `make_mesh`) for each brain structure.
`BOLD`	($V \times Q$ matrix) dimension-reduced fMRI data

`theta`	(list) current parameter estimates
`C_diag`	$(Qx1)$ diagonal elements of residual covariance after dimension reduction
`H`, `Hinv`	For dimension reduction
`s0_vec`	Vectorized prior means
`D`	Sparse diagonal matrix of prior standard deviations
`Dinv_s0`	The inverse of D times s0_vec
`verbose`	If `TRUE`, display progress of algorithm. Default: `FALSE`.
`return_MAP`	If `TRUE`. return the posterior mean and precision of the latent fields instead of the parameter estimates. Default: `FALSE`.
`update`	Which parameters to update. Either `"all"`, `"A"` or `"kappa"`.
`update_nu0sq`	For non-spatial model: updating `nu0sq` is recommended if dimension reduction was not performed, and is not recommended if it was.

Value

An updated list of parameter estimates, theta, OR if `return_MAP=TRUE`, the posterior mean and precision of the latent fields

---

`var_sq_err` *Compute the error between empirical and theoretical variance of covariance matrix elements*

Description

Compute the error between empirical and theoretical variance of covariance matrix elements

Usage

```
var_sq_err(nu, p, var_ij, xbar_ij, xbar_ii, xbar_jj)
```

Arguments

`nu`	Inverse Wishart degrees of freedom parameter
`p`	Matrix dimension for IW distribution
`var_ij`	Empirical between-subject variance of covariance matrices at element (i,j)
`xbar_ij`	Empirical mean of covariance matrices at element (i,j)
`xbar_ii`	Empirical mean of covariance matrices at the ith diagonal element
`xbar_jj`	Empirical mean of covariance matrices at the jth diagonal element

Value

Squared difference between the empirical and theoretical IW variance of covariance matrices at element (i,j)

`var_sq_err_constrained` *Compute the overall error between empirical and theoretical variance of CORRELATION matrix elements*

Description

Compute the overall error between empirical and theoretical variance of CORRELATION matrix elements

Usage

```
var_sq_err_constrained(nu, p, var, xbar, M = 10000)
```

Arguments

`nu`	Inverse Wishart degrees of freedom parameter
`p`	Matrix dimension for IW distribution
`var`	Empirical between-subject variances of CORRELATION matrix (upper triangle)
`xbar`	Empirical mean of CORRELATION matrix (upper triangle)
`M`	Penalty to assign if theoretical variance is smaller than empirical variance

Value

Sum of squared difference between the empirical and theoretical IW variance of CORRELATION matrix, but with constraint that theoretical variances must not be smaller than empirical variances

Index