Package 'BSTFA'

May 14, 2025

Title Fast Computation for Bayesian Spatio-temporal Factor Analysis Model

Description This package fits a spatio-temporal factor analysis model to a spatio-temporal data set and provides plotting functionality for inference and prediction.

Version 0.1.0

License `use_gpl3_license()`

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

VignetteBuilder knitr

Depends R (>= 2.10)

LinkingTo Rcpp, RcppArmadillo

LazyData true

Imports Matrix, MASS, RColorBrewer, ggplot2, ggpubr, mgcv, MCMCpack, coda, npreg, matrixcalc, scatterplot3d, sf, Rcpp, RcppArmadillo

Suggests knitr, rmarkdown

NeedsCompilation yes

Author Adam Simpson [aut], Candace Berrett [aut, cre] (ORCID: <https://orcid.org/0000-0003-4721-3065>)

Maintainer Candace Berrett <cberrett@stat.byu.edu>

Contents

BSTFA *Reduced BSTFA function*

Description

This function uses MCMC to draw from posterior distributions of a Bayesian spatio-temporal factor analysis model. All spatial processes use one of Fourier, thin plate spline, or multiresolution basis functions. The temporally-dependent factors use Fourier bases. The default values are chosen to work well for many data sets. Thus, it is possible to use this function using only three arguments: `ymat`, `dates`, and `coords`. The default number of MCMC iterations is 10000 (saving 5000); however, depending on the number of observations and processes modeled, it may need more draws than this to ensure the posterior draws are representative of the entire posterior distribution space.

Usage

```
BSTFA(
  ymat,
  dates,
  coords,
  iters = 10000,
  n.times = nrow(ymat),
  n.locs = ncol(ymat),
  x = NULL,
  mean = FALSE,
  linear = TRUE,
  seasonal = TRUE,
  factors = TRUE,
  n.seasn.knots = min(7, ceiling(length(unique(yday(dates)))/3)),
  spatial.style = "fourier",
  n.spatial.bases = min(8, ceiling(n.locs/3)),
  knot.levels = 2,
  max.knot.dist = mean(dist(coords)),
  premade.knots = NULL,
  plot.knots = FALSE,
  n.factors = min(4, ceiling(n.locs/20)),
  factors.fixed = NULL,
  plot.factors = FALSE,
  load.style = "fourier",
  n.load.bases = min(6, ceiling(dim(coords)[1]/3)),
  freq.lon = 4 * diff(range(coords[, 1])),
  freq.lat = 4 * diff(range(coords[, 2])),
  n.temp.bases = ifelse(floor(n.times * 0.1)%%2 == 1, floor(n.times * 0.1) - 1,
    floor(n.times * 0.1)),
  freq.temp = n.times,
  alpha.prec = 1/1e+05,
  tau2.gamma = 2,
  tau2.phi = 1e-07,
```

```
  sig2.gamma = 2,
  sig2.phi = 1e-05,
  sig2 = NULL,
  beta = NULL,
  xi = NULL,
  Fmat = matrix(0, nrow = n.times, ncol = n.factors),
  Lambda = matrix(0, nrow = n.locs, n.factors),
  thin = 1,
  burn = iters * 0.5,
  verbose = TRUE,
  filename = "BSTFA.Rdata",
  save.missing = TRUE,
  save.output = FALSE
)
```

Arguments

ymat	Data matrix of size n.times by n.locs. Any missing data should be marked by NA. The model works best if the data are zero-centered for each location.
dates	n.times length vector of class "Date" corresponding to each date of the observed data. For now, the dates should be regularly spaced (e.g., daily).
coords	n.locs by 2 matrix or data frame of coordinates for the locations of the observed data. If using longitude and latitude, longitude is assumed to be the first coordinate.
iters	Number of MCMC iterations to draw. Default value is 10000. Function only saves (iters-burn)/thin drawn values.
n.times	Number of observations for each location. Default is nrow(ymat).
n.locs	Number of observed locations. Default is ncol(ymat).
x	Optional n.locs by p matrix of covariates for each location. If there are no covariates, set to NULL (default).
mean	Logical scalar. If TRUE, the model will fit a spatially-dependent mean for each location. Otherwise, the model will assume the means are zero at each location (default).
linear	Logical scalar. If TRUE (default), the model will fit a spatially-dependent linear increase/decrease (or "slope") in time. Otherwise, the model will assume a zero change in slope across time.
seasonal	Logical scalar. If TRUE (default), the model will use circular b-splines to model a spatially-dependent annual process. Otherwise, the model will assume there is no seasonal (annual) process.
factors	Logical scalar. If TRUE (default), the model will fit a spatio-temporal factor analysis model with temporally-dependent factors and spatially-dependent loadings.
n.seasn.knots	Numeric scalar indicating the number of knots to use for the seasonal basis components. The default value is min(7, ceiling(length(unique(yday(dates)))/3)), where 7 will capture approximately 2 peaks during the year.
spatial.style	Character scalar indicating the style of bases to use for the linear and seasonal components. Style options are 'fourier' (default), 'tps' for thin plate splines, and 'grid' for multiresolution bisquare bases using knots from a grid across the space.

`n.spatial.bases`
: Numeric scalar indicating the number of spatial bases to use when `spatial.style` is either `'fourier'` or `'tps'`. Default value is `min(8, ceiling(n.locs/3))`.

`knot.levels`
: Numeric scalar indicating the number of resolutions to use for when `spatial.style='grid'` and/or `load.style='grid'`. Default is 2.

`max.knot.dist`
: Numeric scalar indicating the maximum distance at which a basis value is greater than zero when `spatial.style='grid'` and/or `load.style='grid'`. Default value is `mean(dist(coords))`.

`premade.knots`
: Optional list of length `knot.levels` with each list element containing a matrix of longitude-latitude coordinates of the knots to use for each resolution when `spatial.style='grid'` and/or `load.style='grid'`. Otherwise, when `premade.knots = NULL` (default), the knots are determined by using the standard multiresolution grids across the space.

`plot.knots`
: Logical scalar indicating whether to plot the knots used when `spatial.style='grid'` and/or `load.style='grid'`. Default is `FALSE`.

`n.factors`
: Numeric scalar indicating how many factors to use in the model. Default is `min(4,ceiling(n.locs/20))`.

`factors.fixed`
: Numeric vector of length `n.factors` indicating the locations to use for the fixed loadings. This is needed for model identifiability. If `factors.fixed=NULL` (default), the code will select locations with less than 20% missing data and that are far apart in the space.

`plot.factors`
: Logical scalar indicating whether to plot the fixed factor locations. Default is `FALSE`.

`load.style`
: Character scalar indicating the style of spatial bases to use for the spatially-dependent loadings. Options are `'fourier'` (default) for the Fourier bases, `'tps'` for thin plate splines, and `'grid'` for multiresolution bases. This can be the same as or different than `spatial.style`.

`n.load.bases`
: Numeric scalar indicating the number of bases to use for the spatially-dependent loadings when `load.style` is either `'fouier'` or `'tps'`. This can be the same as or different than `n.spatial.bases`. Default is `min(6, ceiling(dim(coords)[1]/3))`.

`freq.lon`
: Numeric scalar indicating the frequency to use for the first column of `coords` (assumed to be longitude) for the Fourier bases when `spatial.style='fourier'` and/or `load.style='fourier'`. Default value is `4*diff(range(coords[,1]))`.

`freq.lat`
: Numeric scalar indicating the frequency to use for the second column of `coords` (assumed to be latitude) for the Fourier bases when `spatial.style='fourier'` and/or `load.style='fourier'`. Default value is `4*diff(range(coords[,2]))`.

`n.temp.bases`
: Numeric scalar indicating the number of Fourier bases to use for the temporally-dependent factors. The default value is 10% of `n.times`.

`freq.temp`
: Numeric scalar indicating the frequency to use for the Fourier bases of the temporally-dependent factors. The default value is `n.times`.

`alpha.prec`
: Numeric scalar indicating the prior precision for all model process coefficients. Default value is `1/100000`.

`tau2.gamma`
: Numeric scalar indicating the prior shape for the precision of the model coefficients. Default value is 2.

`tau2.phi`
: Numeric scalar indicating the prior rate for the precision of the model coefficients. Default value is `1e-07`.

`sig2.gamma`
: Numeric scalar indicating the prior shape for the residual precision. Default value is 2.

sig2.phi	Numeric scalar indicating the prior rate for the residual precision. Default value is 1e-05.
sig2	Numeric scalar indicating the starting value for the residual variance. If NULL (default), the function will select a reasonable starting value.
beta	Numeric vector of length n.locs + p indicating starting values for the slopes. If NULL (default), the function will select reasonable starting values.
xi	Numeric vector of length (n.locs + p)*n.seasn.knots indicating starting values for the coefficients of the seasonal component. If NULL (default), the function will select reasonable starting values.
Fmat	Numeric matrix of size n.times by n.factors indicating starting values for the factors. Default value is to start all factor values at 0.
Lambda	Numeric matrix of size n.locs by n.factors indicating starting values for the loadings. Default value is to start all loadings at 0.
thin	Numeric scalar indicating how many MCMC iterations to thin by. Default value is 1, indicating no thinning.
burn	Numeric scalar indicating how many MCMC iterations to burn before saving. Default value is one-half of iters.
verbose	Logical scalar indicating whether or not to print the status of the MCMC process. If TRUE (default), the function will print every time an additional 10% of the MCMC process is completed.
filename	Character scalar indicating the filename to use to save the MCMC output. Default value is 'BSTFA.Rdata'.
save.missing	Logical scalar indicating whether or not to save the MCMC draws for the missing observations. If TRUE (default), the function will save an additional MCMC object containing the MCMC draws for each missing observation. Use FALSE to save file space and memory.
save.output	Logical scalar indicating whether to save the output object to filename. Default value is FALSE.

Value

A list containing the following elements (any elements that are the same as in the function input are removed here for brevity):

mu An mcmc object of size draws by n.locs containing posterior draws for the mean of each location. If mean=FALSE (default), the values will all be zero.

alpha.mu An mcmc object of size draws by n.spatial.bases + p containing posterior draws for the coefficients modeling the mean process. If mean=FALSE (default), the values will all be zero.

tau2.mu An mcmc object of size draws by 1 containing the posterior draws for the variance of the mean process. If mean=FALSE (default), the values will all be zero.

beta An mcmc object of size draws by n.locs containing the posterior draws for the increase/decrease (slope) across time for each location.

alpha.beta An mcmc object of size draws by n.spatial.bases + p containing posterior draws for the coefficients modeling the slope.

tau2.beta An mcmc object of size draws by 1 containing posterior draws of the variance of the slopes.

xi An mcmc object of size `draws` by `n.seasn.knots*n.locs` containing posterior draws for the coefficients of the seasonal process.

alpha.xi An mcmc object of size `draws` by `(n.spatial.bases + p)*n.seasn.knots` containing posterior draws for the coefficients modeling each coefficient of the seasonal process.

tau2.xi An mcmc object of size `draws` by `1` containing posterior draws of the variance of the coefficients of the seasonal process.

F.tilde An mcmc object of size `draws` by `n.times*n.factors` containing posterior draws of the residual factors.

alphaT An mcmc object of size `draws` by `n.factors*n.temp.bases` containing posterior draws of the coefficients for the factor temporally-dependent process.

Lambda.tilde An mcmc object of size `draws` by `n.factors*n.locs` containing posterior draws of the loadings for each location.

alphaS An mcmc object of size `draws` by `n.factors*n.load.bases` containing posterior draws of the coefficients for the loadings spatial process.

tau2.lambda An mcmc object of size `draws` by `1` indicating the residual variance of the loadings spatial process.

sig2 An mcmc object of size `draws` by `1` containing posterior draws of the residual variance of the data.

y.missing If `save.missing=TRUE`, a matrix of size `sum(missing)` by `draws` containing posterior draws of the missing observations. Otherwise, the object is `NULL`.

time.data A data frame of size `iters` by 6 containing the time it took to sample each parameter for every iteration.

setup.time An object containing the time the model setup took.

model.matrices A list containing the matrices used for each modeling process. `newS` is the matrix of spatial basis coefficients for the mean, linear, and seasonal process coefficients. `linear.Tsub` is the matrix used to enforce a linear increase/increase (slope) across time. `seasonal.bs.basis` is the matrix containing the circular b-splines of the seasonal process. `confoundingPmat.prime` is the matrix that enforces orthogonality of the factors from the mean, linear, and seasonal processes. `QT` contains the fourier bases used to model the temporal factors. `QS` contains the bases used to model the spatial loadings.

factors.fixed A vector of length `n.factors` giving the location indices of the fixed loadings.

iters A scalar returning the number of MCMC iterations.

y An `n.times*n.locs` vector of the observations.

missing A logical vector indicating whether that element's observation was missing or not.

doy A numeric vector of length `n.times` containing the day of year for each element in the original `dates`.

knots.spatial For `spatial.style='grid'`, a list of length `knot.levels` containing the coordinates for all knots at each resolution.

knots.load For `load.style='grid'`, a list of length `knot.levels` containing the coordinates for all knots at each resolution.

draws The number of saved MCMC iterations after removing the burn-in and thinning.

Author(s)

Adam Simpson and Candace Berrett

Examples

```
data(utahDataList)
out <- BSTFA(ymat=TemperatureVals, dates=Dates, coords=Coords)
```

BSTFAfull *Full BSTFA function*

Description

This function uses MCMC to draw from posterior distributions of a Bayesian spatio-temporal factor analysis model. The spatial processes for the mean, linear, and seasonal behavior use one of Fourier, thin plate spline, or multiresolution basis functions. The temporal dependence of the factors is modeled using a vector autoregressive model. The spatially-dependent loadings are modeled using a mean-zero Gaussian process with an exponential covariance structure. The default values are chosen to work well for many data sets. Thus, it is possible to use this function using only three arguments: `ymat`, `dates`, and `coords`. The default number of MCMC iterations is 10000 (saving 5000); however, depending on the number of observations and processes modeled, it may need more draws than this to ensure the posterior draws are representative of the entire posterior distribution space.

Usage

```
BSTFAfull(
  ymat,
  dates,
  coords,
  iters = 10000,
  n.times = nrow(ymat),
  n.locs = ncol(ymat),
  x = NULL,
  mean = FALSE,
  linear = TRUE,
  seasonal = TRUE,
  factors = TRUE,
  n.seasn.knots = min(7, ceiling(length(unique(lubridate::yday(dates)))/3)),
  spatial.style = "grid",
  n.spatial.bases = ceiling(n.locs/2),
  knot.levels = 2,
  max.knot.dist = n.locs * 0.05,
  premade.knots = NULL,
  plot.knots = FALSE,
  freq.lon = 4 * diff(range(coords[, 1])),
  freq.lat = 4 * diff(range(coords[, 2])),
  n.factors = min(4, ceiling(n.locs/20)),
  factors.fixed = NULL,
  plot.factors = FALSE,
  alpha.prec = 1/1e+05,
  tau2.gamma = 2,
  tau2.phi = 1e-07,
  sig2.gamma = 2,
  sig2.phi = 1e-05,
```

```
  omega.ii.mean = 1,
  omega.ii.var = 1,
  omega.ij.mean = 0,
  omega.ij.var = 2,
  S.F = diag(1, n.factors),
  nu.F = n.factors,
  phi.gamma = 3,
  phi.phi = 0.5,
  sig2 = NULL,
  beta = NULL,
  xi = NULL,
  Fmat = matrix(0, nrow = n.times, ncol = n.factors),
  Omega = diag(1, n.factors),
  Sigma.F = diag(1, n.factors),
  Lambda = matrix(0, nrow = n.locs, n.factors),
  phi.lambda = rep(1, n.factors),
  thin = 1,
  burn = floor(iters * 0.5),
  c.omega = matrix(0.001, n.factors, n.factors),
  c.phi.lambda = rep(0.001, n.factors),
  adapt.iter = (burn + 10),
  adapt.epsilon = 1e-20,
  verbose = TRUE,
  filename = "STFA.Rdata",
  save.missing = TRUE,
  save.output = FALSE
)
```

Arguments

ymat	Data matrix of size n.times by n.locs. Any missing data should be marked by NA. The model works best if the data are zero-centered for each location.
dates	n.times length vector of class "Date" corresponding to each date of the observed data. For now, the dates should be regularly spaced (e.g., daily).
coords	n.locs by 2 matrix or data frame of coordinates for the locations of the observed data. If using longitude and latitude, longitude is assumed to be the first coordinate.
iters	Number of MCMC iterations to draw. Default value is 10000. Function only saves (iters-burn)/thin drawn values.
n.times	Number of observations for each location. Default is nrow(ymat).
n.locs	Number of observed locations. Default is ncol(ymat).
x	Optional n.locs by p matrix of covariates for each location. If there are no covariates, set to NULL (default).
mean	Logical scalar. If TRUE, the model will fit a spatially-dependent mean for each location. Otherwise, the model will assume the means are zero at each location (default).
linear	Logical scalar. If TRUE (default), the model will fit a spatially-dependent linear increase/decrease (or "slope") in time. Otherwise, the model will assume a zero change in slope across time.

`seasonal` Logical scalar. If `TRUE` (default), the model will use circular b-splines to model a spatially-dependent annual process. Otherwise, the model will assume there is no seasonal (annual) process.

`factors` Logical scalar. If `TRUE` (default), the model will fit a spatio-temporal factor analysis model with temporally-dependent factors and spatially-dependent loadings.

`n.seasn.knots` Numeric scalar indicating the number of knots to use for the seasonal basis components. The default value is `min(7, ceiling(length(unique(yday(dates)))/3))`, where 7 will capture approximately 2 peaks during the year.

`spatial.style` Character scalar indicating the style of bases to use for the mean, linear, and seasonal components. Style options are `'fourier'`, `'tps'` for thin plate splines, and `'grid'` (default) for multiresolution bisquare bases using knots from a grid across the space.

`n.spatial.bases` Numeric scalar indicating the number of spatial bases to use when `spatial.style` is either `'fourier'` or `'tps'`. Default value is `min(8, ceiling(n.locs/3))`.

`knot.levels` Numeric scalar indicating the number of resolutions to use for when `spatial.style='grid'`. Default is 2.

`max.knot.dist` Numeric scalar indicating the maximum distance at which a basis value is greater than zero when `spatial.style='grid'`. Default value is `mean(dist(coords))`.

`premade.knots` Optional list of length `knot.levels` with each list element containing a matrix of longitude-latitude coordinates of the knots to use for each resolution when `spatial.style='grid'`. Otherwise, when `premade.knots = NULL` (default), the knots are determined by using the standard multiresolution grids across the space.

`plot.knots` Logical scalar indicating whether to plot the knots used when `spatial.style='grid'`. Default is `FALSE`.

`freq.lon` Numeric scalar indicating the frequency to use for the first column of `coords` (assumed to be longitude) for the Fourier bases when `spatial.style='fourier'`. Default value is `4*diff(range(coords[,1]))`.

`freq.lat` Numeric scalar indicating the frequency to use for the second column of `coords` (assumed to be latitude) for the Fourier bases when `spatial.style='fourier'`. Default value is `4*diff(range(coords[,2]))`.

`n.factors` Numeric scalar indicating how many factors to use in the model. Default is `min(4,ceiling(n.locs/20))`.

`factors.fixed` Numeric vector of length `n.factors` indicating the locations to use for the fixed loadings. This is needed for model identifiability. If `factors.fixed=NULL` (default), the code will select locations with less than 20% missing data and that are far apart in the space.

`plot.factors` Logical scalar indicating whether to plot the fixed factor locations. Default is `FALSE`.

`alpha.prec` Numeric scalar indicating the prior precision for all model process coefficients. Default value is `1/100000`.

`tau2.gamma` Numeric scalar indicating the prior shape for the precision of the model coefficients. Default value is `2`.

`tau2.phi` Numeric scalar indicating the prior rate for the precision of the model coefficients. Default value is `1e-07`.

`sig2.gamma` Numeric scalar indicating the prior shape for the residual precision. Default value is 2.

sig2.phi	Numeric scalar indicating the prior rate for the residual precision. Default value is 1e-05.
omega.ii.mean	Numeric scalar indicating the prior mean for the diagonal elements of the autoregressive correlation matrix of the factors. Default is 1.
omega.ii.var	Numeric scalar indicating the prior variance for the diagonal elements of the autoregressive correlation matrix of the factors. Default is 1.
omega.ij.mean	Numeric scalar indicating the prior mean for the off-diagonal elements of the autoregressive correlation matrix of the factors. Default is 0.
omega.ij.var	Numeric scalar indicating the prior variance for the off-diagonal elements of the autoregressive correlation matrix of the factors. Default is 2.
S.F	Numeric matrix of size n.factors by n.factors indicating the prior residual covariance matrix for the factors. Default is diag(1,n.factors).
nu.F	Numeric scalar indicating the prior degrees of freedom for the residual covariance matrix of the factors. Default is n.factors; must be greater than or equal to n.factors.
phi.gamma	Numeric scalar indicating the prior shape of the spatial range parameter for the spatially-dependent loadings. Default value is 3.
phi.phi	Numeric scalar indicating the prior rate of the spatial range parameter for the spatially-dependent loadings. Default is 0.5.
sig2	Numeric scalar indicating the starting value for the residual variance. If NULL (default), the function will select a reasonable starting value.
beta	Numeric vector of length n.locs + p indicating starting values for the slopes. If NULL (default), the function will select reasonable starting values.
xi	Numeric vector of length (n.locs + p)*n.seasn.knots indicating starting values for the coefficients of the seasonal component. If NULL (default), the function will select reasonable starting values.
Fmat	Numeric matrix of size n.times by n.factors indicating starting values for the factors. Default value is to start all factor values at 0.
Omega	Numeric matrix of size n.factors by n.factors indicating the starting value for the autoregressive correlation of the factors. Default value is the identity matrix.
Sigma.F	Numeric matrix of size n.factors by n.factors indicating the starting value for the residual covariance matrix of the factors. Default value is the identity matrix.
Lambda	Numeric matrix of size n.locs by n.factors indicating starting values for the loadings. Default value is to start all loadings at 0.
phi.lambda	Numeric vector of length n.factors indicating the starting values for the spatial range parameters for each loading. Default value is a vector of 1's.
thin	Numeric scalar indicating how many MCMC iterations to thin by. Default value is 1, indicating no thinning.
burn	Numeric scalar indicating how many MCMC iterations to burn before saving. Default value is one-half of iters.
c.omega	Numeric matrix of starting values for the proposal standard deviations (for the Metropolis random walk algorithm) for sampling proposal values of the autoregressive correlation matrix for the factors. Default is matrix(0.001, n.factors, n.factors).

c.phi.lambda	Numeric vector of starting values for the proposal standard deviations (for the Metropolis random walk algorithm) for sampling proposal values of the range of the spatially-dependent loadings. Default is rep(0.001, n.factors).
adapt.iter	Numeric scalar indicating the number of iterations to start adjusting the proposal standard deviations for the Metropolis random walk algorithms. Value must be at least 2 larger than burn. Default value is burn+10.
adapt.epsilon	Numeric scalar indicating the small value to add to the proposal standard deviations when using the adaptive Metropolis random walk algorithms. Default is 1e-20.
verbose	Logical scalar indicating whether or not to print the status of the MCMC process. If TRUE (default), the function will print every time an additional 10% of the MCMC process is completed.
filename	Character scalar indicating the filename to use to save the MCMC output. Default value is 'BSTFA.Rdata'.
save.missing	Logical scalar indicating whether or not to save the MCMC draws for the missing observations. If TRUE (default), the function will save an additional MCMC object containing the MCMC draws for each missing observation. Use FALSE to save file space and memory.
save.output	Logical scalar indicating whether to save the output object to filename. Default value is FALSE.

Value

A list containing the following elements (any elements that are the same as in the function input are removed here for brevity):

mu An mcmc object of size draws by n.locs containing posterior draws for the mean of each location. If mean=FALSE (default), the values will all be zero.

alpha.mu An mcmc object of size draws by n.spatial.bases + p containing posterior draws for the coefficients modeling the mean process. If mean=FALSE (default), the values will all be zero.

tau2.mu An mcmc object of size draws by 1 containing the posterior draws for the variance of the mean process. If mean=FALSE (default), the values will all be zero.

beta An mcmc object of size draws by n.locs containing the posterior draws for the increase/decrease (slope) across time for each location.

alpha.beta An mcmc object of size draws by n.spatial.bases + p containing posterior draws for the coefficients modeling the slope.

tau2.beta An mcmc object of size draws by 1 containing posterior draws of the variance of the slopes.

xi An mcmc object of size draws by n.seasn.knots*n.locs containing posterior draws for the coefficients of the seasonal process.

alpha.xi An mcmc object of size draws by (n.spatial.bases + p)*n.seasn.knots containing posterior draws for the coefficients modeling each coefficient of the seasonal process.

tau2.xi An mcmc object of size draws by 1 containing posterior draws of the variance of the coefficients of the seasonal process.

F.tilde An mcmc object of size draws by n.times*n.factors containing posterior draws of the residual factors.

alphaT An mcmc object of size draws by n.factors*n.temp.bases containing posterior draws of the coefficients for the factor temporally-dependent process.

Lambda.tilde An mcmc object of size `draws` by `n.factors*n.locs` containing posterior draws of the loadings for each location.

alphaS An mcmc object of size `draws` by `n.factors*n.load.bases` containing posterior draws of the coefficients for the loadings spatial process.

tau2.lambda An mcmc object of size `draws` by `1` indicating the residual variance of the loadings spatial process.

sig2 An mcmc object of size `draws` by `1` containing posterior draws of the residual variance of the data.

y.missing If `save.missing=TRUE`, a matrix of size `sum(missing)` by `draws` containing posterior draws of the missing observations. Otherwise, the object is `NULL`.

time.data A data frame of size `iters` by 6 containing the time it took to sample each parameter for every iteration.

setup.time An object containing the time the model setup took.

model.matrices A list containing the matrices used for each modeling process. `newS` is the matrix of spatial basis coefficients for the mean, linear, and seasonal process coefficients. `linear.Tsub` is the matrix used to enforce a linear increase/increase (slope) across time. `seasonal.bs.basis` is the matrix containing the circular b-splines of the seasonal process. `confoundingPmat.prime` is the matrix that enforces orthogonality of the factors from the mean, linear, and seasonal processes. `QT` contains the fourier bases used to model the temporal factors. `QS` contains the bases used to model the spatial loadings.

factors.fixed A vector of length `n.factors` giving the location indices of the fixed loadings.

iters A scalar returning the number of MCMC iterations.

y An `n.times*n.locs` vector of the observations.

missing A logical vector indicating whether that element's observation was missing or not.

doy A numeric vector of length `n.times` containing the day of year for each element in the original `dates`.

knots.spatial For `spatial.style='grid'`, a list of length `knot.levels` containing the coordinates for all knots at each resolution.

draws The number of saved MCMC iterations after removing the burn-in and thinning.

Author(s)

Candace Berrett and Adam Simpson

Examples

```
data(utahDataList)
out <- BSTFA.full(ymat=TemperatureVals, dates=Dates, coords=Coords)
```

---

check.convergence *Check effective sample size and geweke diagnostic*

Description

Check effective sample size and geweke diagnostic

Usage

```
check.convergence(
  out,
  type = "eSS",
  cutoff = ifelse(type == "eSS", 100, 0.001)
)
```

Arguments

out	output from STFA or STFAfull

---

computation.summary *Print computation summary*

Description

Print computation summary

Usage

```
computation.summary(out)
```

Arguments

out	output from STFA or STFAfull

---

computeLogLik *Compute log-likelihood*

Description

Compute log-likelihood

Usage

```
computeLogLik(out, verbose = FALSE, addthin = 1)
```

Arguments

out	output from BSTFA or BSTFAfull

plot.annual *Plot annual curve*

Description

Plot annual curve

Usage

```
## S3 method for class 'annual'
plot(
  out,
  location,
  add = F,
  years = "one",
  interval = 0.95,
  yrange = NULL,
  new_x = NULL
)
```

Arguments

out	output from STFA or STFAfull

plot.factor *Plot the factors*

Description

Plot the factors

Usage

```
## S3 method for class 'factor'
plot(
  out,
  factor = 1,
  together = FALSE,
  include.legend = TRUE,
  type = "mean",
  uncertainty = T,
  ci.level = c(0.025, 0.975),
  xrange = NULL
)
```

Arguments

out	output from STFA or STFAfull

plot.fourier.bases *Visualize fourier bases*

Description

Visualize fourier bases

Usage

```
## S3 method for class 'fourier.bases'
plot(
  coords,
  R,
  fine = 100,
  plot.3d = FALSE,
  freq.lon = diff(range(coords[, 1]))^2,
  freq.lat = diff(range(coords[, 2]))^2,
  par.mfrow = c(2, 3)
)
```

Arguments

out	output from STFA or STFAfull

plot.grid *Plot on a grid*

Description

Plot on a grid

Usage

```
## S3 method for class 'grid'
plot(
  out,
  parameter,
  loadings = 1,
  type = "mean",
  ci.level = c(0.025, 0.975),
  yearscale = TRUE,
 color.gradient = colorRampPalette(rev(RColorBrewer::brewer.pal(9, name = "RdBu")))(50)
)
```

Arguments

out	output from STFA or STFAfull

plot.location *Plot a location*

Description

Plot a location

Usage

```
## S3 method for class 'location'
plot(
  out,
  location,
  new_x = NULL,
  type = "mean",
  par.mfrow = c(1, 1),
  pred.int = TRUE,
  ci.level = c(0.025, 0.975),
  uncertainty = TRUE,
  xrange = NULL,
  truth = FALSE,
  ylim = NULL
)
```

Arguments

out	output from STFA or STFAfull

plot.map *Plot on a map*

Description

Plot on a map

Usage

```
## S3 method for class 'map'
plot(
  out,
  parameter = "slope",
  yearscale = TRUE,
  new_x = NULL,
  type = "mean",
  ci.level = c(0.025, 0.975),
  fine = 100,
  color.gradient = colorRampPalette(rev(RColorBrewer::brewer.pal(9, name =
    "RdBu")))(fine),
  with.uncertainty = FALSE,
```

```
  map = FALSE,
  state = FALSE,
  location = NULL,
  loading = 1,
  addthin = 1
)
```

Arguments

out	output from STFA or STFAfull

---

plot.trace *Plot trace plots*

Description

Plot trace plots

Usage

```
## S3 method for class 'trace'
plot(out, parameter, param.range = NULL, par.mfrow = c(1, 1), density = TRUE)
```

Arguments

out	output from STFA or STFAfull

---

predictBSTFA *Prediction*

Description

Prediction

Usage

```
predictBSTFA(
  out,
  location = NULL,
  type = "mean",
  ci.level = c(0.025, 0.975),
  new_x = NULL,
  pred.int = TRUE
)
```

Arguments

out	output from STFA or STFAfull

Index