Package 'skipTrack'

January 27, 2025

Title A Bayesian Hierarchical Model that Controls for Non-Adherence in Mobile Menstrual Cycle Tracking

Version 0.1.2

Description Implements a Bayesian hierarchical model designed to identify skips in mobile menstrual cycle self-tracking on mobile apps. Future developments will allow for the inclusion of covariates affecting cycle mean and regularity, as well as extra information regarding tracking non-adherence. Main methods to be outlined in a forthcoming paper, with alternative models from Li et al. (2022) <doi:10.1093/jamia/ocab182>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.1

URL https://github.com/LukeDuttweiler/skipTrack

BugReports https://github.com/LukeDuttweiler/skipTrack/issues

Imports doParallel (>= 1.0.0), foreach (>= 1.5.0), genMCMCDiag (>= 0.2.0), ggplot2 (>= 3.4.0), ggtext (>= 0.1.0), glmnet (>= 4.1.0), gridExtra (>= 2.0), LaplacesDemon (>= 16.0.0), lifecycle, mvtnorm (>= 1.2.0), optimg (>= 0.1.2), parallel (>= 4.0.0), stats (>= 4.0.0), utils (>= 4.0.0)

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Luke Duttweiler [aut, cre, cph] (<https://orcid.org/0000-0002-0467-995X>)

Maintainer Luke Duttweiler <lduttweiler@hsph.harvard.edu>

Repository CRAN

Date/Publication 2025-01-27 21:50:02 UTC

Contents

`gibbsStepLi` *Gibbs Step Li - One MCMC step for the Li Model*

Description

Gibbs Step Li - One MCMC step for the Li Model

Usage

```
gibbsStepLi(ijDat, iDat, kappa, gamma, alpha, beta, S, indFirst)
```

Arguments

ijDat	A data.frame with parameters at the individual-observation level: Individual, ys, lambdais, piis, ss.
iDat	A data.frame with parameters at the individual level: Individual, lambdas, pis.
kappa	Fixed value of hyperparameter kappa.
gamma	Fixed value of hyperparameter gamma.
alpha	Fixed value of hyperparameter alpha.
beta	Fixed value of hyperparamter beta.
S	Fixed input value S.
indFirst	A logical vector indicating the first occurrence of each individual.

Value

A list containing one MCMC draws for each parameter. Elements are:

ijDat A data.frame with updated parameters at the individual-observation level: Individual, ys, lambdais, piis, ss.

iDat A data.frame with updated parameters at the individual level: Individual, lambdas, pis.

kappa Fixed value of hyperparameter kappa.

gamma Fixed value of hyperparameter gamma.

alpha Fixed value of hyperparameter alpha.

beta Fixed value of hyperparamter beta.

S Fixed input value S.

indFirst A logical vector indicating the first occurrence of each individual.

References

Li, Kathy, et al. "A predictive model for next cycle start date that accounts for adherence in menstrual self-tracking." Journal of the American Medical Informatics Association 29.1 (2022): 3-11.

`liInference` *Perform hyperparameter inference assuming the model given in Li et al. (2022) on a cycle length dataset.*

Description

This function performs hyperparameter inference on a given dataset of individuals and their tracked cycles, assuming the model specified in Li et al. (2022). Default starting values for hyperparameters and optimization tuning parameters are those given in Li et al.

Usage

```
liInference(
  Y,
  cluster,
  S = 10,
  startingParams = c(kappa = 180, gamma = 6, alpha = 2, beta = 20)
)
```

Arguments

Y	A vector of observed cycle lengths.
cluster	A vector indicating the individual cluster/group membership for each observation Y.
S	Maximum number of possible skipped cycles (see Li et al. for details).
startingParams	A vector of starting values for hyperparameters (default values from Li et al.).

Value

A list containing the results of hyperparameter inference.

References

Li, Kathy, et al. "A predictive model for next cycle start date that accounts for adherence in menstrual self-tracking." Journal of the American Medical Informatics Association 29.1 (2022): 3-11.

---

likVec *Monte Carlo estimate of negative marginal log-likelihood of Li model*

Description

This function calculates a Monte Carlo estimate of the negative marginal log-likelihood of the given hyperparameters for the generative model from Li et al. (2022). It samples M instances of the parameters from the given distributions and averages the the likelihoods, giving a marginal likelihood for the hyperparameters.

Usage

```
likVec(
  pars = c(kappa = 180, gamma = 6, alpha = 2, beta = 20),
  S = 10,
  M = 1000,
  cycleDat,
  verbose = FALSE,
  ...
)
```

Arguments

pars	Named numeric vector of hyperparameters containing the elements: kappa, gamma, alpha, beta. NOTE: MUST BE IN CORRECT ORDER. • kappa: Numeric value, shape parameter of Gamma distribution for Lambda_i. • gamma: Numeric value, rate parameter of Gamma distribution for Lambda_i. • alpha: Numeric value, shape1 parameter of Beta distribution for Pi_i. • beta: Numeric value, shape2 parameter of Beta distribution for Pi_i.
S	Integer, maximum number of allowed skips in the model.
M	Integer specifying the number of Monte Carlo iterations.
cycleDat	Data frame containing information about individuals and their tracked cycles.
verbose	Logical with default FALSE. If true, prints extra info while running.
...	Does nothing.

Value

Numeric value representing the Monte Carlo estimate of the negative marginal log-likelihood.

References

Li, Kathy, et al. "A predictive model for next cycle start date that accounts for adherence in menstrual self-tracking." Journal of the American Medical Informatics Association 29.1 (2022): 3-11.

---

liMCMC *Runs MCMC algorithm for performing inference using the model from Li et al. (2022)*

Description

This function performs inference on cycle length data, assuming the model from Li et al. (2022). It is important to note that Li et al. does not actually use this algorithm as they target a particular analytic posterior predictive distribution, and solve directly. However, we are targeting a different posterior and thus use this MCMC to perform inference.

Usage

```
liMCMC(
  Y,
  cluster,
  S,
  hyperparams = c(kappa = 180, gamma = 6, alpha = 2, beta = 20),
  initialParams = list(pi = c(1/3, 1/3, 1/3), lambdais = rep(30,
    length(unique(cycleDat$Individual))), piis = rep(0.2,
   length(unique(cycleDat$Individual))), ss = sample(0:S, nrow(cycleDat), replace =
    TRUE)),
  reps = 1000,
  ...
)
```

Arguments

`Y`	A vector of observed cycle lengths.
`cluster`	A vector indicating the individual cluster/group membership for each observation Y.
`S`	Integer. The maximum number of skips to consider possible.
`hyperparams`	Named numeric vector of hyperparameters containing the elements: kappa, gamma, alpha, beta. NOTE: MUST BE IN CORRECT ORDER. • `kappa`: Numeric value, shape parameter of Gamma distribution for Lambda_i. • `gamma`: Numeric value, rate parameter of Gamma distribution for Lambda_i. • `alpha`: Numeric value, shape1 parameter of Beta distribution for Pi_i. • `beta`: Numeric value, shape2 parameter of Beta distribution for Pi_i.
`initialParams`	A list of initial parameter values for the MCMC algorithm. Default values are provided for pi, lambdais, piis, ss.
`reps`	The number of MCMC iterations (steps) to perform. Default is 1000.
`...`	For catching unused arguments (like li = TRUE)

Value

A list containing the MCMC draws for each parameter at each iteration. Each element in the list is itself a list containing:

ijDat A data.frame with updated parameters at the individual-observation level: Individual, ys, lambdais, piis, ss.

iDat A data.frame with updated parameters at the individual level: Individual, lambdas, pis.

kappa Fixed value of hyperparameter kappa.

gamma Fixed value of hyperparameter gamma.

alpha Fixed value of hyperparameter alpha.

beta Fixed value of hyperparamter beta.

S Fixed input value S.

indFirst A logical vector indicating the first occurrence of each individual.

References

Li, Kathy, et al. "A predictive model for next cycle start date that accounts for adherence in menstrual self-tracking." Journal of the American Medical Informatics Association 29.1 (2022): 3-11.

See Also

`gibbsStepLi`

`liSim` *Simulate user tracked menstrual cycle data for an individual using the li model.*

Description

This function generates synthetic data for user tracked menstrual cycles for a single individual using the li model. For this model Beta0 = log(30), and Gamma0 doesn't really make sense.

Usage

```
liSim(i, skipProb, maxCycles, trueBetas, trueGammas = NULL, avgCyclesPer)
```

Arguments

`i`	Individual identifier. Character, numeric or integer.
`skipProb`	Vector, ignored for this model.
`maxCycles`	Integer, Maximum possible number of true cycles per tracked cycle.
`trueBetas`	Optional. True values for generated mean regression coefficients.
`trueGammas`	NULL, left for consistency. Will throw error if specified.
`avgCyclesPer`	Average number of cycles contributed by each individual. Actual number is drawn from Poisson for each person. Default is 7.

Value

'Individual' Individual identifiers.

'TrackedCycles' Tracked cycles.

'NumTrue' Number of true values.

'SkipProb' Individual's probability of skipping tracking a cycle

'Mean' Individual's mean values.

'Beta0' Beta0 true value.

'Gamma0 NA

'Z0' 1

'X0',...,'XN' Covariate matrix for Mean, where N is the length of trueBetas.

See Also

`skipTrack.simulate`

`mixSim` *Simulate user tracked menstrual cycle data for an individual using the mixture model.*

Description

This function generates synthetic data for user tracked menstrual cycles for a single individual using the mixture model. For this model Beta_0 is set to log(30) and Gamma_0 is set to 15, although for the skipTrack model this lacks interpretation.

Usage

```
mixSim(i, skipProb, maxCycles, trueBetas, trueGammas, overlap, avgCyclesPer)
```

Arguments

`i`	Individual identifier. Character, numeric, or integer.
`skipProb`	Vector of probabilities for the number of true cycles per tracked cycle. For example, (.7, .2, .1) means that 70% of observed cycles will contain one true cycle, 20% will contain 2 true cycles, and 10% will contain 3 true cycles.
`maxCycles`	Maximum number of true cycles per tracked cycle.
`trueBetas`	Optional. True values for generated mean regression coefficients.
`trueGammas`	Optional. True values for the generated precision regression coefficients.
`overlap`	Optional. Number of (non-intercept) columns shared between X and Z. Columns are shared from left to right.
`avgCyclesPer`	Average number of cycles contributed by each individual. Actual number is drawn from Poisson for each person. Default is 7.

Value

'Individual' Individual identifiers.

'TrackedCycles' Tracked cycles.

'NumTrue' Number of true values.

'Mean' Individual's mean values.

'Beta0' Beta0 true value.

'Gamma0' Gamma0 true value.

'X0',...,'XN' Covariate matrix for Mean, where N is the length of trueBetas.

'Z0',...,'ZM' Covariate matrix for precision, where M is the length of trueGammas.

See Also

`skipTrack.simulate`

plot.skipTrack.model *Plot skipTrack.model objects*

Description

Plot skipTrack.model objects

Usage

```
## S3 method for class 'skipTrack.model'
plot(x, ...)
```

Arguments

x	skipTrack.model object from the function skipTrack.fit
...	Needed for S3 consistency

Value

Invisible NULL. Prints plots from skipTrack.visualize

Examples

```
#Simulated data
simDat <- skipTrack.simulate(n = 100, skipProb = c(.7, .2, .1))

#Run model fit (should typically run with much more than 50 reps)
modFit <- skipTrack.fit(Y = simDat$Y, cluster = simDat$cluster, chains = 2, reps = 50)
plot(modFit)
```

postBeta *Draw from Posterior Distribution for Beta Parameters*

Description

In our model mui follows a normal distribution with mean Xi^T %*% beta and precision rho. Additionally we assume that beta follows a mvnormal prior with mean 0 and precision (rho_Beta) * I.This function draws from the posterior distribution of beta under these assumptions.

Usage

```
postBeta(rhoBeta = 0.01, rho, Xi, muI)
```

Arguments

rhoBeta	A scalar representing the prior precision parameter for beta.
rho	A scalar representing the precision parameter.
Xi	A matrix of covariates, where each row represents an individual and each column represents a covariate.
muI	A vector where each element is the mean for individual i.

Details

This function assumes that Xi is a (num Individuals) x (dimension of beta) matrix of covariates.

Value

A vector representing a draw from the posterior distribution of beta parameters.

---

postCij *Sample a vector of values from the full conditional posterior of the c_ij vector*

Description

In our model the data are drawn from LogN(mu_i + log(c_ij), tau_i) The prior for c_ij is a categorical prior with category probabilities pi1, ..., pik, and c_ij can take values 1, ..., k where k is the length of pi. This function samples from the full conditional posterior of all c_ijs, given vectors of equal length yijs, muis, tauis

Usage

```
postCij(yijs, pi, muis, tauis)
```

Arguments

yijs	Numeric Vector, cycle lengths
pi	Numeric vector, must sum to 1. Sampled probabilities for c_ijs
muis	Numeric vector, log of sampled mean for all individuals yijs
tauis	Numeric vector > 0, sampled precision for all individuals yijs

Value

Integer vector

postGamma — *Perform a Metropolis-Hastings Step for Drawing a New Gamma*

Description

Our model assumes that tau_i ~ Gamma(alpha_i, phi) where alpha_i/phi = theta_i and g(theta_i) = Z_i^T Gamma, where g is the log-link function. Because of this GLM formulation we cannot simply draw from a posterior here and instead use a Metropolis-Hastings step with proposal distribution propGamma ~ MVNormal(currentGamma, rhoGamma*I)

Usage

```
postGamma(taui, Zi, currentGamma, phi = 1, rhoGamma = 1000)
```

Arguments

taui	A vector of length num_Individuals representing precision parameters for individuals.
Zi	A matrix of covariates, where each row represents an individual and each column represents a covariate.
currentGamma	A vector (or matrix with 1 row) representing the current Gamma value.
phi	A scalar, the prior rate for tau_i.
rhoGamma	A scalar representing the proposal distribution precision parameter.

Details

This draw step assumes a log-link function for the Gamma GLM that we are fitting.

Value

A list containing the new Gamma value and the corresponding thetai values.

postLambdai — *Compute random draw from full conditional posterior for lambda_i in Li algorithm.*

Description

This function calculates a random draw from the full conditional posterior distribution for lambda_i in the Li algorithm, given the observed values y_ij, the indicators s_ij, and the prior hyperparameters priorK and priorG.

Usage

```
postLambdai(yij, sij, priorK, priorG)
```

Arguments

yij	Vector of observed values for individual i.
sij	Vector of cycle skip indicators for individual i.
priorK	Prior hyperparameter kappa.
priorG	Prior hyperparameter gamma.

Value

A random draw from the posterior distribution of lambda_i.

---

postMui *Sample a value from the full conditional posterior of mu_i*

Description

In our model the data are drawn from LogN(mu_i + log(c_ij), tau_i). The prior for mu_i is given as N(x_i^T %*% beta, rho). This function draws from the conditional posterior of mu_i.

Usage

```
postMui(yij, cij, taui, xib, rho)
```

Arguments

yij	Numeric vector, cycle lengths for a single individual
cij	Positive Integer vector, a sampled vector of length(yij) where the corresponding values in cij indicate a sampled number of TRUE cycles in each cycle length given by yij
taui	Numeric > 0, A sampled precision for the yijs
xib	Numeric, result of multiplying x_i^T %*% beta (single value, not vector)
rho	Numeric > 0, sampled prior precision of mu_i

Details

Additionally, note that in order to vectorize the remainder of the MCMC algorithm this function returns the sampled value repeated for length(yij)

Value

Numeric vector, repeated sampled value of length(yij)

postPhi *Metropolis-Hastings step to draw a new value for phi.*

Description

In our model the data are drawn from LogN(mu_i + log(c_ij), tau_i). The prior for tau_i is given as Gamma(thetai*phi, phi). This function uses a MH step to draw a new sample of phi. Proposal distribution is Gamma(currentPhi*rhoPhi, rhoPhi). Note that we parameterize with RATE, not SCALE.

Usage

```
postPhi(taui, thetai, currentPhi, rhoPhi = 1000)
```

Arguments

taui	Numeric vector, individuals precisions.
thetai	Numeric vector. individuals precisions means (estimate)
currentPhi	Previous draw of phi
rhoPhi	Proposal rate for gamma distribution that draws proposal for phi, default is 1000.

Value

Numeric, new draw of phi

postPi *Sample a value from the full conditional posterior of pi*

Description

In our model the data are drawn from LogN(mu_i + log(c_ij), tau_i) The prior for c_ij is a categorical prior with category probabilities pi1, ..., pik, and c_ij can take values 1, ..., k where k is the length of pi. This function samples from the posterior of pi = pi1, ..., pik, assuming that pi follows Dirichlet(priorAlphas)

Usage

```
postPi(ci, priorAlphas)
```

Arguments

ci	Integer vector, all of the sampled cij values for all individuals
priorAlphas	Numeric vector, prior dirichlet parameters for pi

Value

Numeric vector

postPii *Compute M-H draw for pi_i in Li algorithm*

Description

This performs a Metropolis-Hastings draw for pi_i, assuming s_ij follows a truncated geometric distribution with parameters pi_i and S. The proposal distribution for pi_i is Beta(alpha, beta).

Usage

```
postPii(sij, currentPii, priorA, priorB, S)
```

Arguments

sij	Vector of cycle skip indicators for individual i
currentPii	Current value of pi_i
priorA	Hyperparameter alpha.
priorB	Hyperparameter beta.
S	Maximum number of skips allowed in algorithm

Value

Draw for pi_i, repeated for the number of observations from individual i

postRho *Sample a value from the full conditional posterior of rho*

Description

In our model the data are drawn from LogN(mu_i + log(c_ij), tau_i). The prior for mu_i is given as N(mu, rho). This function draws from the conditional posterior of rho, given that the prior on rho is a uniform prior on the standard deviation.

Usage

```
postRho(muI, xib)
```

Arguments

muI	Numeric vector, log of individuals mean values.
xib	Numeric vector, result of X %*% Beta, same length as muI.

Value

Numeric

postSij — *Compute random draw from full conditional posterior for s_ij in Li algorithm.*

Description

This function calculates a random draw from the full conditional posterior distribution for s_ij in the Li algorithm, given the observed values yijs, the parameter pi, the lambda_i value, and the truncation level S.

Usage

```
postSij(yijs, pii, lambdai, S)
```

Arguments

yijs	Vector of observed values for s_ij.
pii	Probability parameter pi.
lambdai	Value of lambda_i.
S	Truncation level.

Value

A random draw from the posterior distribution of s_ij.

postTaui — *Sample a value from the full conditional posterior of tau_i*

Description

In our model the data are drawn from LogN(mu_i + log(c_ij), tau_i). The prior for tau_i is given as Gamma(thetai*phi, phi). This function draws from the conditional posterior of tau_i. Note that we parameterize with RATE, not SCALE.

Usage

```
postTaui(yij, cij, mui, thetai, phi = 1)
```

Arguments

yij	Numeric vector, cycle lengths for a single individual
cij	Positive Integer vector, a sampled vector of length(yij) where the corresponding values in cij indicate a sampled number of TRUE cycles in each cycle length given by yij
mui	Numeric, log of sampled mean of this individual's yijs
thetai	Numeric, mean of prior (gamma) distribution on taui
phi	Numeric, rate for Taui prior

Details

Additionally, note that in order to vectorize the remainder of the MCMC algorithm this function returns the sampled value repeated for length(yij)

Value

Numeric vector, repeated sampled value of length(yij)

`print.skipTrack.model` *Print skipTrack.model to console*

Description

Print skipTrack.model to console

Usage

```
## S3 method for class 'skipTrack.model'
print(x, ...)
```

Arguments

x	skipTrack.model object from the function skipTrack.fit
...	Needed for S3 consistency

Value

Invisible NULL. Prints info about skipTrack.model object

Examples

```
#Simulated data
simDat <- skipTrack.simulate(n = 100, skipProb = c(.7, .2, .1))

#Run model fit (should typically run with much more than 50 reps)
modFit <- skipTrack.fit(Y = simDat$Y, cluster = simDat$cluster, chains = 2, reps = 50)
print(modFit)
```

`sampleStep` *Perform a single step of the MCMC sampling process for skipTrack*

Description

This function performs a single step of the Markov Chain Monte Carlo (MCMC) algorithm to update parameters in a hierarchical model used for identifying skips in menstrual cycle tracking.

Usage

```
sampleStep(
  ijDat,
  iDat,
  rho,
  pi,
  Xi,
  Zi,
  Beta,
  Gamma,
  priorAlphas,
  indFirst,
  rhoBeta,
  rhoGamma,
  phi,
  rhoPhi,
  fixedSkips
)
```

Arguments

`ijDat`	A data.frame with individual-observation level parameters: Individual, ys, cijs, muis, tauis.
`iDat`	A data.frame with individual level parameters: Individual, mus, taus, thetas.
`rho`	Updated value of the global parameter rho.
`pi`	Updated value of the global parameter pi.
`Xi`	A matrix (numIndividuals x length(Beta)) of covariates for cycle length mean. Default is a vector of 1's. NOTE THE DIFFERENCE (from skipTrack.MCMC) IN EXPECTED DIMENSION OF X
`Zi`	A matrix (numIndividuals x length(Gamma)) of covariates for cycle length precision. Default is a vector of 1's. NOTE THE DIFFERENCE (from skipTrack.MCMC) IN EXPECTED DIMENSION OF Z
`Beta`	Matrix (1 x length(Beta)) of coefficients for cycle length mean.
`Gamma`	Matrix of (1 x length(Gamma)) coefficients for cycle length precision.
`priorAlphas`	Vector of prior alpha values for updating pi.

indFirst	A logical vector indicating the first occurrence of each individual.
rhoBeta	Updated value of the global parameter rhoBeta.
rhoGamma	Value of the proposal parameter rhoGamma.
phi	Value of the parameter phi.
rhoPhi	Value of the proposal parameter rhoPhi.
fixedSkips	Logical. If TRUE cycle skip information (cijs) is not updated in sample steps and the inputs are instead assumed to be true.

Value

A list containing updated parameters after performing a single MCMC step. The list includes:

ijDat A data.frame with updated parameters at the individual-observation level: Individual, ys, cijs, muis, tauis.

iDat A data.frame with updated parameters at the individual level: Individual, mus, taus, thetas.

rho Updated value of the global parameter rho.

pi Updated value of the global parameter pi.

Xi Matrix of covariates for cycle length mean.

Zi Matrix of covariates for cycle length precision.

Beta Updated matrix of coefficients for cycle length mean.

Gamma Updated matrix of coefficients for cycle length precision.

priorAlphas Vector of prior alpha values for updating pi.

indFirst A logical vector indicating the first occurrence of each individual.

rhoBeta Hyperprior parameter rhoBeta, used to update Beta.

rhoGamma Value of the proposal parameter rhoGamma.

phi Updated value of the parameter phi.

rhoPhi Value of the proposal parameter rhoPhi.

fixedSkips Logical. Indicates if skips were fixed.

---

skipTrack.diagnostics *skipTrack MCMC Diagnostics*

Description

Takes model results from skipTrack.fit and uses genMCMCDiag to get generalized mcmc diagnostics

Usage

```
skipTrack.diagnostics(
  stFit,
  param = c("rho", "phi", "Betas", "Gammas", "muis", "tauis", "cijs"),
  proximityMap = NULL,
  ...
)
```

Arguments

stFit	A list of MCMC results from the skipTrack.fit function.
param	A character string specifying the parameter for which diagnostics are to be calculated. Must be one of: 'rho', 'phi', 'Betas', 'Gammas', 'muis', 'tauis', or 'cijs'.
proximityMap	An optional parameter specifying the proximity-map for calculating diagnostics. See package genMCMCDiag for details. Default is NULL.
...	Arguments passed on to genMCMCDiag::genDiagnostic
	diagnostics A character vector or list of diagnostic functions to be evaluated. Options include 'traceplot', 'ess', 'psrf', or custom functions. See details.
	distance Function for evaluating distance between MCMC draws if required by 'method'. This should be a pairwise distance function that operates on elements of the chains from mhDraws. Note that the lanfear and ts proximityMaps ALWAYS require a distance function.
	verbose If TRUE, informative messages are displayed.

Details

If the parameter is 'rho' or 'phi' (the univariate parameters), the function extracts the specified parameter from the MCMC results and calculates diagnostics using the genDiagnostic function with the standard proximityMap. If the parameter is any of the other available options, the function extracts the corresponding values and calculates diagnostics using the genDiagnostic function with the specified or default proximityMap ('lanfear') and hammingDist as the distance function.

Details on genDiagnostic can be found in the genMCMCDiag package.

Value

A mcmcDiag object of MCMC diagnostics for the specified parameter

See Also

genDiagnostic, skipTrack.fit

Examples

```
#Simulated data
simDat <- skipTrack.simulate(n = 100, skipProb = c(.7, .2, .1))

#Run model fit (should typically run with much more than 50 reps)
modFit <- skipTrack.fit(Y = simDat$Y, cluster = simDat$cluster, chains = 2, reps = 50)

#Get diagnostics for cijs
skipTrack.diagnostics(modFit, 'cijs')
```

`skipTrack.fit` *Fits the skipTrack Model using 1 or more MCMC chains*

Description

This function fits the model using multiple instances of skipTrack.MCMC, either in parallel or sequentially.

Usage

```
skipTrack.fit(
  Y,
  cluster,
  X = matrix(1, nrow = length(cluster)),
  Z = matrix(1, nrow = length(cluster)),
  numSkips = 10,
  reps = 1000,
  chains,
  useParallel = FALSE,
  ...
)
```

Arguments

`Y`	A vector of observed cycle lengths.
`cluster`	A vector indicating the individual cluster/group membership for each observation Y.
`X`	A matrix (length(Y) x length(Beta)) of covariates for cycle length mean. Default is a vector of 1's.
`Z`	A matrix (length(Y) x length(Gamma)) of covariates for cycle length precision. Default is a vector of 1's.
`numSkips`	The maximum number of skips to allow. Default is 10.
`reps`	The number of MCMC iterations (steps) to perform. Default is 1000.
`chains`	Number of chains to run.
`useParallel`	Logical, indicating whether to use parallel processing, as supported by doParallel. Default is FALSE.
`...`	Arguments passed on to `skipTrack.MCMC` `fixedSkips` If TRUE cycle skip information (cijs) is not updated in sample steps and the inputs are instead assumed to be true. `initialParams` A list of initial parameter values for the MCMC algorithm. Default values are provided for pi, muis, tauis, rho, cijs, alphas, Beta, Gamma, phi, rhoBeta, rhoGamma, and rhoPhi. `verbose` logical. If true progress bars and additional info are printed to the console.

Value

A list containing the results of skipTrack.MCMC for each chain.

See Also

skipTrack.MCMC

Examples

```
#Simulated data
simDat <- skipTrack.simulate(n = 100, skipProb = c(.7, .2, .1))

#Run model fit (should typically run with much more than 50 reps)
modFit <- skipTrack.fit(Y = simDat$Y, cluster = simDat$cluster, chains = 2, reps = 50)
modFit
```

---

skipTrack.MCMC *Perform one chain of MCMC sampling for the skipTrack model.*

Description

This function runs a single Markov Chain Monte Carlo (MCMC) chain to update parameters in the skipTrack hierarchical model.

Usage

```
skipTrack.MCMC(
  Y,
  cluster,
  X = matrix(1, nrow = length(cluster)),
  Z = matrix(1, nrow = length(cluster)),
  numSkips = 10,
  reps = 1000,
  fixedSkips = FALSE,
 initialParams = list(pi = rep(1/(numSkips + 1), numSkips + 1), muis = rep(log(30),
  length(unique(cluster))), tauis = rep(5, length(unique(cluster))), rho = 1, cijs =
   sample(1:3, length(Y), replace = TRUE), alphas = rep(1, numSkips + 1), Beta =
  matrix(rep(0, ncol(as.matrix(X))), 1), Gamma = matrix(rep(0, ncol(as.matrix(Z))), 1),
   rhoBeta = 0.01, rhoGamma = 1000, phi = 0.01, rhoPhi = 1000),
  verbose = FALSE
)
```

Arguments

Y	A vector of observed cycle lengths.
cluster	A vector indicating the individual cluster/group membership for each observation Y.
X	A matrix (length(Y) x length(Beta)) of covariates for cycle length mean. Default is a vector of 1's.
Z	A matrix (length(Y) x length(Gamma)) of covariates for cycle length precision. Default is a vector of 1's.
numSkips	The maximum number of skips to allow. Default is 10.
reps	The number of MCMC iterations (steps) to perform. Default is 1000.
fixedSkips	If TRUE cycle skip information (cijs) is not updated in sample steps and the inputs are instead assumed to be true.
initialParams	A list of initial parameter values for the MCMC algorithm. Default values are provided for pi, muis, tauis, rho, cijs, alphas, Beta, Gamma, phi, rhoBeta, rhoGamma, and rhoPhi.
verbose	logical. If true progress bars and additional info are printed to the console.

Value

A list containing the MCMC draws for each parameter at each iteration. Each element in the list is itself a list containing:

ijDat A data.frame with updated parameters at the individual-observation level: Individual, ys, cijs, muis, tauis.

iDat A data.frame with updated parameters at the individual level: Individual, mus, taus, thetas.

rho Updated value of the global parameter rho.

pi Updated value of the global parameter pi.

Xi Matrix of covariates for cycle length mean.

Zi Matrix of covariates for cycle length precision.

Beta Updated matrix of coefficients for cycle length mean.

Gamma Updated matrix of coefficients for cycle length precision.

priorAlphas Vector of prior alpha values for updating pi.

indFirst A logical vector indicating the first occurrence of each individual.

rhoBeta Hyperprior parameter rhoBeta, used to update Beta.

rhoGamma Value of the proposal parameter rhoGamma.

phi Updated value of the parameter phi.

rhoPhi Value of the proposal parameter rhoPhi.

fixedSkips Logical. Indicates if skips were fixed.

See Also

sampleStep

skipTrack.results *Get tables of Inference results from skipTrack.fit*

Description

This function calculates inference results on Betas, Gammas, and cijs based on the provided MCMC results. It returns summaries such as credible intervals for Betas, Gammas, wald-type confidence intervals for cijs, and Gelman-Rubin PSRF diagnostics for all 3. Note that true values and converage are included in the output if trueVals is supplied, but otherwise not.

Usage

```
skipTrack.results(stFit, trueVals = NULL, burnIn = 750)
```

Arguments

stFit	Object result of skipTrack.fit function.
trueVals	Optional named list containing true values for Betas, Gammas, and cijs. (Also can use output of skipTrack.simulate)
burnIn	Number of MCMC iterations to discard as burn-in per chain.

Value

A list containing the following elements:

Betas	data.frame with 95% credible intervals and (if trueVals is supplied) true values for Betas and Coverage tag.
Gammas	data.frame with 95% credible intervals and (if trueVals is supplied) true values for Gammas and Coverage tag.
cijs	data.frame with Wald-type 95% confidence intervals and (if trueVals is supplied) true values for cijs and Coverage tags.
Diagnostics	data.frame with ess and gelman-rubin diagnostics from genMCMCDiag package, for parameter sets 'Betas', 'Gammas' and 'cijs'.

Examples

```
#Simulated data
simDat <- skipTrack.simulate(n = 100, skipProb = c(.7, .2, .1))

#Run model fit (should typically run with much more than 50 reps)
modFit <- skipTrack.fit(Y = simDat$Y, cluster = simDat$cluster, chains = 2, reps = 50)
modFit

# If using simulated data (which includes access to ground truth):
#
skipTrack.results(modFit, trueVals = simDat, burnIn = 25)
#Recommended burnIn with real data is at least 750
```

```
#
# If not using simulated data:
#
skipTrack.results(modFit, burnIn = 25)
#Recommended burnIn with real data is at least 750
```

---

skipTrack.simulate *Simulate user-tracked menstrual cycle data for multiple individuals*

Description

This function generates synthetic data for user-tracked menstrual cycles given a generative model, skip probabilities, maximum cycles and covariates (depending on the model). It supports built-in models ('skipTrack', 'li', 'mixture') and custom models written as functions.

Usage

```
skipTrack.simulate(
  n,
  model = c("skipTrack", "li", "mixture"),
  skipProb = NULL,
  maxCycles = length(skipProb),
  trueBetas = NULL,
  trueGammas = NULL,
  overlap = 0,
  avgCyclesPer = 7
)
```

Arguments

n	Number of individuals to simulate data for.
model	model for data simulation. Can be a character ('skipTrack', 'li', 'mixture') or a custom function.
skipProb	Vector of probabilities for number of true cycles per tracked cycle. For example, (.7, .2, .1) means that 70% of observed cycles will contain one true cycle, 20% will contain 2 true cycles and 10% will contain 3 true cycles. Default is NULL. If model == 'li', skipProb values are set and user input will be ignored.
maxCycles	Maximum number of cycles for generating skip cycles. Default is the length of skipProb. If model == 'li', this must be specified; if model == 'skipTrack' or 'mixture', leave as default.
trueBetas	Optional. True values for the mean regression coefficients (not counting intercept which is automatic based on the model).
trueGammas	Optional. True values for the precision regression coefficients (not counting intercept which is automatic based on the model). Precision covariates not available for model == 'li'.

`overlap`	Optional. Number of (non-intercept) columns shared between X and Z. Columns are shared from left to right.
`avgCyclesPer`	Average number of cycles contributed by each individual. Actual number is drawn from Poisson for each person. Default is 7.

Value

A list containing:

'Y' Tracked cycles from the simulated data.

'cluster' Individual identifiers from the simulated data.

'X' Covariate matrix for Betas (mean cycle length).

'Z' Covariate matrix for Gammas (regularity).

'Beta' True beta coefficients.

'Gamma' True gamma coefficients.

'NumTrue' Number of true cycles in each tracked cycle. Order matches Y.

'Underlying' Subset of the simulated data containing individual level information. For 'skipTrack' - individual mean and precision for log(cycle lengths), for 'li' - individual mean for cycle lengths, for 'mixture' - individual mean for cycle lengths

References

Li, Kathy, et al. "A predictive model for next cycle start date that accounts for adherence in menstrual self-tracking." Journal of the American Medical Informatics Association 29.1 (2022): 3-11.

See Also

`stSim`, `liSim`, `mixSim`

Examples

```
# Example simulation from the SkipTrack model
resultSt <- skipTrack.simulate(1000, model = 'skipTrack', skipProb = c(.7, .2, .1))
hist(resultSt$Y, breaks = 5:200)

# Example simulation from the Li model
resultLi <- skipTrack.simulate(1000, model = 'li', maxCycles = 3)
hist(resultLi$Y, breaks = 5:200)

#Example simulation from the mixture model
resultMix <- skipTrack.simulate(1000, model = 'mixture', skipProb = c(.7, .2, .1))
hist(resultMix$Y, breaks = 5:200)
```

`skipTrack.visualize` *Visualize Results from skipTrack.fit*

Description

This function takes results from skipTrack.fit and produces several helpful visualizations.

Usage

```
skipTrack.visualize(stFit)
```

Arguments

`stFit`	A list containing MCMC results obtained from skipTrack.fit.

Value

A list of three ggplot2 objects:

- cijOverLength - Scatter plot of estimated Cij values against reported cycle length.
- cijOverTaus - Scatter plot of estimated Cij values against estimated individual precisions, colored by cycle length.
- cijDens - Density plot of Y values overlayed with a density plot of Y values separated by estimated cij value.

See Also

`skipTrack.fit` for generating MCMC results.

Examples

```
#Simulated data
simDat <- skipTrack.simulate(n = 100, skipProb = c(.7, .2, .1))

#Run model fit (should typically run with much more than 50 reps)
modFit <- skipTrack.fit(Y = simDat$Y, cluster = simDat$cluster, chains = 2, reps = 50)

#Visualize results
skipTrack.visualize(modFit)
```

str.skipTrack.model *Report skipTrack.model structure to console*

Description

Report skipTrack.model structure to console

Usage

```
## S3 method for class 'skipTrack.model'
str(object, ...)
```

Arguments

object	skipTrack.model object from the function skipTrack.fit
...	To match other str calls

Value

Invisible NULL. Prints structure of skipTrack.model object

Examples

```
#Simulated data
simDat <- skipTrack.simulate(n = 100, skipProb = c(.7, .2, .1))

#Run model fit (should typically run with much more than 50 reps)
modFit <- skipTrack.fit(Y = simDat$Y, cluster = simDat$cluster, chains = 2, reps = 50)
str(modFit)
```

stSim *Simulate user tracked menstrual cycle data for an individual, based on the skipTrack model.*

Description

This function generates synthetic data for user tracked menstrual cycles for a single individual. For this model Beta_0 = log(30), Gamma_0 = 5.5, and phi = .01.

Usage

```
stSim(i, skipProb, maxCycles, trueBetas, trueGammas, overlap, avgCyclesPer)
```

Arguments

i	Individual identifier. Character, numeric or integer.
skipProb	Vector of probabilities for number of true cycles per tracked cycle. For example, (.7, .2, .1) means that 70% of observed cycles will contain one true cycle, 20% will contain 2 true cycles and 10% will contain 3 true cycles.
maxCycles	Maximum number of true cycles per tracked cycle. Ignored for this model.
trueBetas	Optional. True values for the mean regression coefficients (not counting intercept which is automatic based on the model).
trueGammas	Optional. True values for the precision regression coefficients (not counting intercept which is automatic based on the model).
overlap	Optional. Number of (non-intercept) columns shared between X and Z. Columns are shared from left to right.
avgCyclesPer	Average number of cycles contributed by each individual. Actual number is drawn from Poisson for each person. Default is 7.

Value

'Individual' Individual identifiers.

'TrackedCycles' Tracked cycles.

'NumTrue' Number of true values.

'LogMean' Individual's mean of log(Y).

'LogPrec' Individual's precision of log(Y)

'Beta0' Beta0 true value.

'Gamma0' Gamma0 true value.

'X0',...,'XN' Covariate matrix for Mean, where N is the length of trueBetas.

'Z0',...,'ZM' Covariate matrix for precision, where M is the length of trueGammas.

See Also

skipTrack.simulate

---

summary.skipTrack.model

Report skipTrack.model results to console

Description

Report skipTrack.model results to console

Usage

```
## S3 method for class 'skipTrack.model'
summary(object, ...)
```

Arguments

object	skipTrack.model object from the function skipTrack.fit
...	arguments passed to skipTrack.results

Value

Invisible skipTrack.results. Prints info about skipTrack.model object

Examples

```
#Simulated data
simDat <- skipTrack.simulate(n = 100, skipProb = c(.7, .2, .1))

#Run model fit (should typically run with much more than 50 reps)
modFit <- skipTrack.fit(Y = simDat$Y, cluster = simDat$cluster, chains = 2, reps = 50)
summary(modFit, burnIn = 25) #Recommended burnIn with real data is at least 750
```

Index