Package ‘hbmem’

August 22, 2023

Type Package
Title Hierarchical Bayesian Analysis of Recognition Memory
Version 0.3-4
Date 2023-08-21
Author Michael S. Pratte
Maintainer Mike Pratte <prattems@gmail.com>
Depends R (>= 1.8.0), methods
Description Contains functions for fitting hierarchical versions of EVSD, UVSD, DPSD, DPSD with d’ restricted to be positive, and our gamma signal detection model to recognition memory confidence-ratings data.
License LGPL (>= 2.0)
LazyLoad yes
URL https://pcn.psychology.msstate.edu/
Repository CRAN
Date/Publication 2023-08-22 18:30:06 UTC
NeedsCompilation yes

R topics documented:

hbmem-package ........................................... 2
dpsdSample ............................................. 4
dpsdSim .................................................... 6
gammaLikeSample ..................................... 8
gammaSample .......................................... 10
gammaSim ............................................... 13
prm09 ..................................................... 14
rtgamma .................................................. 15
sampleGamma .......................................... 16
uvsdSample .......................................... 18
uvsdSim ............................................... 21

Index 24
Description

Contains functions for fitting hierarchical versions of EVSD, UVSD, DPSD, and our gamma signal detection model to recognition memory confidence-ratings data.

Author(s)

Michael S. Pratte <prattems@gmail.com>

References

Morey, Pratte, and Rouder (2008); Pratte, Rouder, and Morey (2009); Pratte and Rouder (2012).

See Also

'uvsdSample' to fit hierarchical UVSD model, 'uvsdSim' to simulate data from the hierarchical UVSD model, 'dpsdSample' to fit the hierarchial DPSD model, 'dpsdSim' to simulate data from the hierarchial DPSD model, 'dpsdPosSim' and 'dpsdPosSample' for the DPSD model with positive sensitivity, and datasets from our publications.

Examples

#In this example data are simulated from EVSD
#They are then fit by both UVSD and DPSD

library(hbmem)
sim=uvsdSim(s2aS2=0,s2bS2=0) #Simulate data from hierarchical EVSD
dat=as.data.frame(cbind(sim@subj,sim@item,sim@Scond,sim@cond,sim@lag,sim@resp))
colnames(dat)=c("sub","item","Scond","cond","lag","resp")

M=10 #Set way low for speed
keep=2:M
#For real analysis we run 105000 iterations
#with the first 5000 serving as burnin, and
#only keep every 10th iteration for analysis,
#i.e., thinning the chains to mitigate autocorrelation.
ev=uvsdSample(dat,M=M,keep=keep,equalVar=TRUE) #Fit EVSD
uv=uvsdSample(dat,M=M,keep=keep,freeSig2=TRUE) #Fit UVSND w/1 Sigma2
dp=dpsdSample(dat,M=M,keep=keep) #Fit DPSD

#Look at available information
slotNames(uv)
slotNames(dp)

#Compare DIC; smaller is better
ev@DIC
# Effective parameters. Because there are no real effects on studied-item variance, the hierarchical models are drastically shrinking these effect parameters to zero, so that they do not count as full parameters.

```r
# PLOTS FROM UVSD FIT
par(mfrow=c(3,2), pch=19, pty='s')
# Make sure chains look OK
matplot(uvsd@blockN[,uvsd@muN], t='l', xlab='Iteration', ylab='Mu-N')
abline(h=sim@muN, col='blue')
matplot(uvsd@blockS[,uvsd@muS], t='l', xlab='Iteration', ylab='Mu-S')
abline(h=sim@muS, col='blue')
# Estimates of Alpha as function of true values
plot(uvsd@estN[uvsd@alphaN]~sim@alphaN, xlab='True Alpha-N', ylab='Est. Alpha-N'); abline(0, 1, col='blue')
plot(uvsd@estS[uvsd@alphaS]~sim@alphaS, xlab='True Alpha-S', ylab='Est. Alpha-S'); abline(0, 1, col='blue')
# Estimates of Beta as function of true values
plot(uvsd@estN[uvsd@betaN]~sim@betaN, xlab='True Beta-N', ylab='Est. Beta-N'); abline(0, 1, col='blue')
plot(uvsd@estS[uvsd@betaS]~sim@betaS, xlab='True Beta-S', ylab='Est. Beta-S'); abline(0, 1, col='blue')
```

```r
### Look at Sigma2 and Recollection from UVSD and DPSD ###
par(mfrow=c(2,3), pch=19, pty='s')
plot(sqrt(exp(uvsd@blockS2[,uvsd@muS])), t='l', ylab='Sigma', main='Grand Mean')
abline(h=1, col='blue')
hist(uvsd@blockS2[,uvsd@s2alphaS], main='Participant Effect')
hist(uvsd@blockS2[,uvsd@s2betaS], main='Item Effect')
plot(pnorm(dpsd@blockR[,dpsd@muS]), t='l', ylab='P(Recollection)', main='Grand Mean')
abline(h=0, col='blue')
hist(dpsd@blockR[,dpsd@s2alphaS], main='Participant Effect')
hist(dpsd@blockR[,dpsd@s2betaS], main='Item Effect')
```

```r
# See what DPSD does with EVSD effects
par(mfrow=c(2,3))
plot(dpsd@estN[dpsd@alphaN]~sim@alphaN, xlab='True Alpha-N', ylab='DPSD Alpha-N'); abline(0, 1, col='blue')
plot(dpsd@estS[dpsd@alphaS]~sim@alphaS, xlab='True Alpha-S', ylab='DPSD Alpha-S'); abline(0, 1, col='blue')
plot(dpsd@estR[dpsd@alphaS]~sim@alphaS, xlab='True Alpha-S', ylab='DPSD Alpha-S'); abline(0, 1, col='blue')
plot(dpsd@estR[dpsd@alphaS]~sim@alphaS, xlab='True Alpha-S', ylab='DPSD Alpha-S'); abline(0, 1, col='blue')
```
dpsdSample

Function to fit hierarchical DPSD model to data.

Description
Runs MCMC estimation for the hierarchical DPSD model.

Usage
dpsdSample(dat, M = 5000, keep = (M/10):M, getDIC = TRUE,
freeCrit=TRUE,Hier=TRUE, jump=.01)

Arguments
dat      Data frame that must include variables Scond, cond, sub, item, lag, resp. Scond indexes studied/new, whereas cond indexes conditions nested within the studied or new conditions. Indexes for Scond, cond, sub, item, and response must start at zero and have no gaps (i.e., no skipped subject numbers). Lags must be zero-centered.
M        Number of MCMC iterations.
keep     Which MCMC iterations should be included in estimates and returned. Use keep to both get rid of burn-in, and thin chains if necessary.
getDIC   Logical. Should the function compute DIC value? This takes a while if M is large.
freeCrit Logical. If true then criteria are estimated separately for each participant. Should be set to false if analyzing only one participant (e.g., if averaging over subjects).
Hier     Logical. If true then the variances of effects (e.g., item effects) are estimated from the data, i.e., effects are treated as random. If false then these variances are fixed to 2.0 (.5 for recollection effects), thus treating these effects as fixed. This option is there to allow for comparison with more traditional approaches, and to see the effects of imposing hierarchical structure. It should always be set to TRUE in real analysis, and is not even guaranteed to work if set to false.
jump     The criteria and decorrelating steps utilize Matropolis-Hastings sampling routines, which require tuning. All MCMC functions should self-tune during the burnin period (iterations before keep), and they will alert you to the success of tuning. If acceptance rates are too low, "jump" should be decreased, if they are too high, "jump" should be increased. Alternatively, or in addition to adjusting "jump", simply increase the burnin period which will allow the function more time to self-tune.
**Value**

The function returns an internally defined "uvsd" structure that includes the following components:

- **mu**: Indexes which element of blocks contain μ
- **alpha**: Indexes which element of blocks contain participant effects, α
- **beta**: Indexes which element of blocks contain item effects, β
- **s2alpha**: Indexes which element of blocks contain variance of participant effects (α).
- **s2beta**: Indexes which element of blocks contain variance of item effects (β).
- **theta**: Indexes which element of blocks contain θ, the slope of the lag effect
- **estN**: Posterior means of block parameters for new-item means
- **estS**: Posterior means of block parameters for studied-item means
- **estR**: Posterior means of block for Recollection means.
- **estCrit**: Posterior means of criteria
- **blockN**: Each iteration for each parameter in the new-item mean block. Rows index iteration, columns index parameter.
- **blockS**: Same as blockN, but for the studied-item means
- **blockR**: Same as blockN, but for the recollection-parameter means.
- **s.crit**: Samples of each criteria.
- **pD**: Number of effective parameters used in DIC. Note that this should be smaller than the actual number of parameters, as constraint from the hierarchical structure decreases the number of effective parameters.
- **DIC**: DIC value. Smaller values indicate better fits. Note that DIC is notably biased toward complexity.
- **M**: Number of MCMC iterations run
- **keep**: MCMC iterations that were used for estimation and returned
- **b0**: Metropolis-Hastings acceptance rates for decorrelating steps. These should be between .2 and .6. If they are not, the M, keep, or jump arguments need to be adjusted.
- **b0Crit**: acceptance rates for criteria.

**Author(s)**

Michael S. Pratte

**References**

See Pratte, Rouder, & Morey (2009)

**See Also**

hbmem
Examples

# In this example we generate data from EVSD, then fit it with both
# hierarchical DPSD and DPSD assuming no participant or item effects.
library(hbmem)
sim = dpsdSim(I = 30, J = 200)
dat = as.data.frame(cbind(sim@subj, sim@item, sim@cond, sim@Scond, sim@lag, sim@resp))
colnames(dat) = c("sub", "item", "cond", "Scond", "lag", "resp")
dat$lag[dat$Scond == 1] = dat$lag[dat$Scond == 1] - mean(dat$lag[dat$Scond == 1])

M = 10  # Too low for real analysis!
keep = 2:M
DPSD = dpsdSample(dat, M = M)

# Look at all parameters
par(mfrow = c(3, 3), pch = 19, pty = "s")
matplot(DPSD@blockN[, DPSD@muN], t = 'l', ylab = "muN")
abline(h = sim@muN, col = "blue")
plot(DPSD@estN[DPSD@alphaN] - sim@alphaN)
abline(0, 1, col = "blue")
plot(DPSD@estN[DPSD@betaN] - sim@betaN)
abline(0, 1, col = "blue")

matplot(DPSD@blockS[, DPSD@muS], t = 'l', ylab = "muS")
abline(h = sim@muS, col = "blue")
plot(DPSD@estS[DPSD@alphaS] - sim@alphaS)
abline(0, 1, col = "blue")
plot(DPSD@estS[DPSD@betaS] - sim@betaS)
abline(0, 1, col = "blue")

matplot(pnorm(DPSD@blockR[, DPSD@muS]), t = 'l', ylab = "P(recollection)")
abline(h = pnorm(sim@muR), col = "blue")
plot(DPSD@estR[DPSD@alphaR] - sim@alphaR)
abline(0, 1, col = "blue")
plot(DPSD@estR[DPSD@betaR] - sim@betaR)
abline(0, 1, col = "blue")

---

dpsdSim

Function dpsdSim

Description

Simulates data from a hierarchical DPSD model.
**Usage**

dpsdSim(NN=2,NS=1,I=30,J=200,K=6,muN=c(-.7,-.5),s2aN=.2,s2bN=.2, muS=0,s2aS=.2,s2bS=.2,muR=qnorm(.25),s2aR=.2,s2bR=.2, crit=matrix(rep(c(-1.6,-.5,0,.5,1.6),each=I),nrow=I,ncol=(K-1)))

**Arguments**

- **NN**: Number of new-item conditions.
- **NS**: Number of studied-item conditions.
- **I**: Number of participants.
- **J**: Number of items.
- **K**: Number of response options.
- **muN**: Mean of new-item distribution. If there are more than one new-item conditions this is a vector of means with length equal to NN.
- **s2aN**: Variance of participant effects on mean of new-item distribution.
- **s2bN**: Variance of item effects on mean of new-item distribution.
- **muS**: Mean of studied-item distribution. If there are more than new-item conditions this is a vector of means with length equal to NNone studied-item conditions this is a vector of means with length equal to NS.
- **s2aS**: Variance of participant effects on mean of studied-item distribution.
- **s2bS**: Variance of item effects on mean of studied-item distribution.
- **muR**: Mean recollection, on probit space.
- **s2aR**: Variance of participant effects recollection.
- **s2bR**: Variance of item effects on recollection.
- **crit**: Matrix of criteria (not including -Inf or Inf). Columns correspond to criteria, rows correspond to participants.

**Value**

The function returns an internally defined "dpsdSim" structure.

**Author(s)**

Michael S. Pratte

**References**

See Pratte, Rouder, & Morey (2009)

**See Also**

hbmem
Examples

```r
library(hbmem)
# Data from hiererchial model
sim=dpsdSim()
slotNames(sim)
# Scond indicates studied/new
# cond indicates which condition (e.g., deep/shallow)

table(sim@resp,sim@Scond,sim@cond)

# Usefull to make data.frame for passing to functions
dat=as.data.frame(cbind(sim@subj,sim@item,sim@Scond,sim@cond,sim@lag,sim@resp))
colnames(dat)=c("sub","item","Scond","cond","lag","resp")

table(dat$resp,dat$Scond,dat$cond)
```

---

**gammaLikeSample**

Function **gammaLikeSample**

**Description**

Runs MCMC for the hierarchical Gamma Likelihood model

**Usage**

```r
gammaLikeSample(dat, M = 10000, keep = (M/10):M, getDIC = TRUE,
shape=2,jump=.005)
```

**Arguments**

- **dat**
  Data frame that must include variables cond,sub,item,lag,resp. Indexes for cond, sub, item, and respose must start at zero and have no gapes (i.e., no skipped subject numbers). Lags must be zero-centered.

- **M**
  Number of MCMC iterations.

- **keep**
  Which MCMC iterations should be included in estimates and returned. Use keep to both get ride of burn-in, and thin chains if necessary

- **getDIC**
  Logical. should the function compute DIC value? This takes a while if M is large.

- **shape**
  Fixed shape across both new and studied distribuutions.

- **jump**
  The criteria and decorrelating steps utilize Matropolis-Hastings sampling routines, which require tuning. All MCMC functions should self tune during the burnin perior (iterations before keep), and they will alert you to the success of tuning. If acceptance rates are too low, "jump" should be decreased, if they are too hight, "jump" should be increased. Alternatively, or in addition to adjusting "jump", simply increase the burnin period which will allow the function more time to self-tune.
Value

The function returns an internally defined "uvsd" S4 class that includes the following components:

- **mu**: Indexes which element of blocks contain grand means, \( \mu \)
- **alpha**: Indexes which element of blocks contain participant effects, \( \alpha \)
- **beta**: Indexes which element of blocks contain item effects, \( \beta \)
- **s2alpha**: Indexes which element of blocks contain variance of participant effects (\( \alpha \)).
- **s2beta**: Indexes which element of blocks contain variance of item effects (\( \beta \)).
- **theta**: Indexes which element of blocks contain theta, the slope of the lag effect
- **estN**: Posterior means of block parameters for new-item means
- **estS**: Posterior means of block parameters for studied-item means
- **estS2**: Not used for gamma model.
- **estCrit**: Posterior means of criteria
- **blockN**: Each iteration for each parameter in the new-item mean block. Rows index iteration, columns index parameter.
- **blockS**: Same as blockN, but for the studied-item means
- **blockS2**: Not used for gamma model.
- **s.crit**: Samples of each criteria.
- **pD**: Number of effective parameters used in DIC. Note that this should be smaller than the actual number of parameters, as constraint from the hierarchical structure decreases the number of effective parameters.
- **DIC**: DIC value. Smaller values indicate better fits. Note that DIC is notably biased toward complexity.
- **M**: Number of MCMC iterations run
- **keep**: MCMC iterations that were used for estimation and returned
- **b0**: Metropolis-Hastings acceptance rates for new-item distribution parameters. These should be between .2 and .6. If they are not, the M, keep, or jump need to be adjusted.
- **b0S2**: Metropolis-Hastings acceptance rates for studied-item distribution parameters.
- **b0Crit**: Metropolis-Hastings acceptance rates for criteria.

Author(s)

Michael S. Pratte

See Also

hbmem
gammaSample

Function gammaSample

description
Runs MCMC for the hierarchical Gamma model

Usage

gammaSample(dat, M = 10000, keep = (M/10):M, getDIC = TRUE, freeCrit=TRUE, shape=2, jump=.005)
**Arguments**

dat
Data frame that must include variables cond, sub, item, lag, resp. Indexes for cond, sub, item, and respone must start at zero and have no gaps (i.e., no skipped subject numbers). Lags must be zero-centered.

M
Number of MCMC iterations.

keep
Which MCMC iterations should be included in estimates and returned. Use keep to both get ride of burn-in, and thin chains if necessary

getDIC
Logical. should the function compute DIC value? This takes a while if M is large.

freeCrit
Logical. If TRUE (default) individual criteria vary across people. If false, all participants have the same criteria (but note that overall response biases are still modeled in the means)

shape
Fixed shape across both new and studied distributions.

jump
The criteria and decorrelating steps utilize Matropolis-Hastings sampling routines, which require tuning. All MCMC functions should self tune during the burnin period (iterations before keep), and they will alert you to the success of tuning. If acceptance rates are too low, "jump" should be decreased, if they are too high, "jump" should be increased. Alternatively, or in addition to adjusting "jump", simply increase the burnin period which will allow the function more time to self-tune.

**Value**

The function returns an internally defined "uvsd" S4 class that includes the following components

mu
Indexes which element of blocks contain grand means, mu

alpha
Indexes which element of blocks contain participant effects, alpha

beta
Indexes which element of blocks contain item effects, beta

s2alpha
Indexes which element of blocks contain variance of participant effects (alpha).

s2beta
Indexes which element of blocks contain variance of item effects (beta).

theta
Indexes which element of blocks contain theta, the slope of the lag effect

estN
Posterior means of block parameters for new-item means

estS
Posterior means of block parameters for studied-item means

estS2
Not used for gamma model.

estCrit
Posterior means of criteria

blockN
Each iteration for each parameter in the new-item mean block. Rows index iteration, columns index parameter.

blockS
Same as blockN, but for the studied-item means

blockS2
Not used for gamma model.

s.crit
Samples of each criteria.

pD
Number of effective parameters used in DIC. Note that this should be smaller than the actual number of parameters, as constraint from the hierarchical structure decreases the number of effective parameters.
DIC value. Smaller values indicate better fits. Note that DIC is notably biased toward complexity.

M  Number of MCMC iterations run

keep  MCMC iterations that were used for estimation and returned

b0  Metropolis-Hastings acceptance rates for new-item distribution parameters. These should be between .2 and .6. If they are not, the M, keep, or jump need to be adjusted.

b0S2  Metropolis-Hastings acceptance rates for studied-item distribution parameters.

b0Crit  Metropolis-Hastings acceptance rates for criteria.

Author(s)

Michael S. Pratte

See Also

hbmem

Examples

# make data from gamma model
library(hbmem)
sim=gammaSim(I=30,J=200)
dat=as.data.frame(cbind(sim@subj,sim@item,sim@cond,sim@Scond,sim@lag,sim@resp))
colnames(dat)=c("sub","item","cond","Scond","lag","resp")

M=10  # set very small for demo speed
keep=2:M
gamma=gammaSample(dat,M=M,keep=keep,jump=.01)

par(mfrow=c(3,2),pch=19,pty='s')
# Look at chains of MuN and MuS
matplot(gamma@blockN[,gamma@muN],t='l',xlab="Iteration",ylab="Mu-N")
abline(h=sim@muN,col="blue")
matplot(gamma@blockS[,gamma@muS],t='l',xlab="Iteration",ylab="Mu-S")
abline(h=sim@muS,col="blue")

# Estimates of Alpha as function of true values
plot(gamma@estN[gamma@alphaN]~sim@alphaN,xlab="True Alpha-N",ylab="Est. Alpha-N");abline(0,1,col="blue")
plot(gamma@estS[gamma@alphaS]~sim@alphaS,xlab="True Alpha-S",ylab="Est. Alpha-S");abline(0,1,col="blue")

# Estimates of Beta as function of true values
plot(gamma@estN[gamma@betaN]~sim@betaN,xlab="True Beta-N",ylab="Est. Beta-N");abline(0,1,col="blue")
plot(gamma@estS[gamma@betaS]~sim@betaS,xlab="True Beta-S",ylab="Est. Beta-S");abline(0,1,col="blue")

gamma@estN[c(gamma@s2alphaN,gamma@s2betaN)]
gamma@estS[c(gamma@s2alphaS,gamma@s2betaS)]
#Look at some criteria
par(mfrow=c(2,2))
for(i in 1:4)
  matplot(t(gamma$s.crit[,i]),t='l')

---

**gammaSim**

*Function gammaSim*

**Description**
Simulates data from a hierarchical Gamma model.

**Usage**

```r
gammaSim(NN=1,NS=2,I=30,J=200,K=6,muN=log(.65),s2aN=.2,s2bN=.2,
muS=log(c(.8,1.2)),s2aS=.2,s2bS=.2,lagEffect=-.001,shape=2,
crit=matrix(rep(c(.3,.6,1,1.2,1.6),each=I),ncol=(K-1)))
```

**Arguments**

- **NN** Number of conditions for new words.
- **NS** Number of conditions for studied words.
- **I** Number of participants.
- **J** Number of items.
- **K** Number of response options.
- **muN** Mean of new-item distribution. If NN is greater than 1, then muN must be a vector of length NN.
- **s2aN** Variance of participant effects on mean of new-item distribution.
- **s2bN** Variance of item effects on mean of new-item distribution.
- **muS** Mean of studied-item distribution. If NS is greater than 1, then muS must be a vector of length NS.
- **s2aS** Variance of participant effects on mean of studied-item distribution.
- **s2bS** Variance of item effects on mean of studied-item distribution.
- **lagEffect** Linear slope of lag effect on log of studied-item scale.
- **shape** Common shape for both new and studied distributions.
- **crit** Matrix of criteria (not including -Inf or Inf). Columns correspond to criteria, rows correspond to participants.

**Value**
The function returns an internally defined “uvsdSim” structure.
Author(s)

Michael S. Pratte

References

See Pratte, Rouder, & Morey (2009)

See Also

hbmem

Examples

library(hbmem)
#Data from hiererchial model
sim=gammaSim()
slotNames(sim)
table(sim@resp,sim@cond,sim@Scond)

#Usefull to make data.frame for passing to model-fitting functions
dat=as.data.frame(cbind(sim@subj,sim@item,sim@cond,sim@Scond,sim@lag,sim@resp))
colnames(dat)=c("sub","item","cond","Scond","lag","resp")
table(dat$resp,dat$cond,dat$Scond)

Description

Confidence ratings data from Pratte, Rouder, and Morey (2009).

Usage

data(prm09)

Format

A flat-field data frame (each row is a trial) with the following variables

cond  0=new; 1=studied
sub  index of subject starting at 0
item  index of item starting at 0
lag  index of lag, zero-centered
resp  which response was made; 0="sure new"
Details

Participants studied a list of 240 words, and were then tested on the 240 studied and on 240 new words. At test, participants made one of six confidence ratings ranging from "sure new" to "sure studied". Note that to apply the models to these data the "Scond" variable should be set to "cond", and the "cond" variable should be all zeros. This is a backwards-compatibility issue.

Source


Examples

library(hbmem)
data(prm09)
table(prm09$resp,prm09$cond)
#Turn it into data suitable for
#analysis with HBMEM functions:
newdat=prm09
newdat$Scond=newdat$cond
newdat$cond=0
summary(newdat)

---

**rtgamma**

Function rtgamma

Description

Returns random draws from truncated gamma distribution.

Usage

rtgamma(N, shape, scale, a, b)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Number of samples.</td>
</tr>
<tr>
<td>shape</td>
<td>Shape of gamma distribution.</td>
</tr>
<tr>
<td>scale</td>
<td>Scale of gamma distribution.</td>
</tr>
<tr>
<td>a</td>
<td>Lower truncation point.</td>
</tr>
<tr>
<td>b</td>
<td>Upper truncation point.</td>
</tr>
</tbody>
</table>
Function sampleGamma

Description
Samples posterior of mean parameters of the hierarchical linear model on the log scale parameter of a gamma distribution. Usually used within an MCMC loop.

Usage
sampleGamma(sample, y, cond, subj, item, lag, N, I, J, R, ncond, nsub, nitem, s2mu, s2a, s2b, met, shape, sampLag, pos = FALSE)

Arguments
- sample: Block of linear model parameters from previous iteration.
- y: Vector of data
- cond: Vector of condition index, starting at zero.
- subj: Vector of subject index, starting at zero.
- item: Vector of item index, starting at zero.
- lag: Vector of lag index, zero-centered.
- N: Number of conditions.
- I: Number of subjects.
- J: Number of items.
- R: Total number of trials.
- ncond: Vector of length (N) containing number of trials per condition.
- nsub: Vector of length (I) containing number of trials per each subject.
- nitem: Vector of length (J) containing number of trials per each item.
- s2mu: Prior variance on the grand mean mu; usually set to some large number.
- s2a: Shape parameter of inverse gamma prior placed on effect variances.
- s2b: Rate parameter of inverse gamma prior placed on effect variances. Setting both s2a AND s2b to be small (e.g., .01, .01) makes this an uninformative prior.
- met: Vector of tuning parameter for metropolis-hastings steps. Here, all sampling (except variances of alpha and beta) and decorrelating steps utilize the M-H sampling algorithm. This should be adjusted so that .2 < b0 < .6.
- shape: Single shape of Gamma distribution.
- sampLag: Logical. Whether or not to sample the lag effect.
- pos: Logical. If true, the model on scale is 1+exp(mu + alpha + beta). That is, the scale is always greater than one.
Value

The function returns a list. The first element of the list is the newly sampled block of parameters. The second element contains a vector of 0s and 1s indicating which of the decorrelating steps were accepted.

Author(s)

Michael S. Pratte

See Also

hbmem

Examples

library(hbmem)
N=2
shape=2
I=30
J=50
R=I*J
# make some data
mu=log(c(1,2))
alpha=rnorm(I,0,.2)
beta=rnorm(J,0,.2)
theta=-.001
cond=sample(0:(N-1),R,replace=TRUE)
subj=rep(0:(I-1),each=J)
item=NULL
for(i in 1:I)
  item=c(item,sample(0:(J-1),J,replace=FALSE))
lag=rnorm(R,0,100)
lag=lag-mean(lag)
resp=1:R
for(r in 1:R)
  {
    resp[r]=rgamma(1,shape=shape,scale=scale)
  }
ncond=table(cond)
nsub=table(subj)
nitem=table(item)
M=10
keep=2:M
B=N+I+J+3
s.block=matrix(0,nrow=M,ncol=B)
met=rep(.08,B)
b0=rep(0,B)
jump=.0005
for(m in 2:M)
Function uvsdSample

Description

Runs MCMC estimation for the hierarchical UVSD model.

Usage

uvsdSample(dat, M = 10000, keep = (M/10):M, getDIC = TRUE, freeCrit=TRUE, equalVar=FALSE, freeSig2=FALSE, Hier=TRUE, jump=.0001)
Arguments

- **dat**: Data frame that must include variables Scond, cond, sub, item, lag, resp. Scond indexes studied/new, whereas cond indexes conditions nested within the studied or new conditions. Indexes for Scond, cond, sub, item, and response must start at zero and have no gaps (i.e., no skipped subject numbers). Lags must be zero-centered.

- **M**: Number of MCMC iterations.

- **keep**: Which MCMC iterations should be included in estimates and returned. Use keep to both get ride of burn-in, and thin chains if necessary.

- **getDIC**: Logical. should the function compute DIC value? This takes a while if M is large.

- **freeCrit**: Logical. If TRUE (default) individual criteria vary across people. If false, all participants have the same criteria. This should be set to false if there is only one participant, e.g., if averaging data over subjects.

- **equalVar**: Logical. If FALSE (default), unequal-variance model is fit. If TRUE, equal-variance model is fit.

- **freeSig2**: Logical. If FALSE (default), one sigma is fit for all participants and items (as in Pratte, et al., 2009). If TRUE, then an additive model is placed on the log of sigma2 (as in Pratte and Rouder 2010).

- **Hier**: Logical. If TRUE then the variances of effects (e.g., item effects) are estimated from the data, i.e., effects are treated as random. If FALSE then these variances are fixed to 2.0 (.5 for recollection effects), thus treating these effects as fixed. This option is there to allow for compairson with more traditional approaches, and to see the effects of imposing hierarcical structure. It should always be set to TRUE in real analysis, and is not even guaranteed to work if set to false.

- **jump**: The criteria and decorrelating steps utilize Matropolis-Hastings sampling routines, which require tuning. All MCMC functions should self tune during the burnin period (iterations before keep), and they will alert you to the success of tuning. If acceptance rates are too low, "jump" should be decreased, if they are too hight, "jump" should be increased. Alternatively, or in addition to adjusting "jump", simply increase the burnin period which will allow the function more time to self-tune.

Value

The function returns an internally defined "uvsd" S4 class that includes the following components:

- **mu**: Indexes which element of blocks contain grand means, mu
- **alpha**: Indexes which element of blocks contain participant effects, alpha
- **beta**: Indexes which element of blocks contain item effects, beta
- **s2alpha**: Indexes which element of blocks contain variance of participant effects (alpha).
- **s2beta**: Indexes which element of blocks contain variance of item effects (beta).
- **theta**: Indexes which element of blocks contain theta, the slope of the lag effect
- **estN**: Posterior means of block parameters for new-item means
estS  Posterior means of block parameters for studied-item means
estS2 Posterior means of block for studied-item variances.
estCrit Posterior means of criteria
blockN Each iteration for each parameter in the new-item mean block. Rows index iteration, columns index parameter.
blockS Same as blockN, but for the studied-item means
blockS2 Same as blockN, but for variances of studied-item distribution. If equalVar=TRUE, then these values are all zero. If UVSD is fit but freeSig2=FALSE, then only the first element is non-zero (mu).
s.crit Samples of each criteria.
pD Number of effective parameters used in DIC. Note that this should be smaller than the actual number of parameters, as constraint from the hierarchical structure decreases the number of effective parameters.
DIC DIC value. Smaller values indicate better fits. Note that DIC is notably biased toward complexity.
M Number of MCMC iterations run
keep MCMC iterations that were used for estimation and returned
b0 Metropolis-Hastings acceptance rates for decorrelating steps. These should be between .2 and .6. If they are not, the M, keep, or jump need to be adjusted.
b0S2 If additive model is placed on Sigma2 (i.e., freeSigma2=TRUE), then all parameters on S2 must be tuned. b0S2 are the acceptance probabilities for these parameters.

Author(s)
Michael S. Pratte

References
See Pratte, Rouder, & Morey (2009)

See Also
hbmem

Examples
#In this example we generate data from UVSD with a different muN,muS, and Sigma2 for every person and item. These data are then fit with hierarchical UVSD allowing participant or item effects on log(sigma2).

library(hbmem)
sim=uvsdSim(NN=1,muN=-.5,NS=2,muS=c(.5,1),I=30,J=300,s2aN = .2, s2bN = .2, muS=log(c(1.3,1.5)),s2aS=.2,s2bS=.2,s2aS2=.2,s2bS2=.2)
sim=

dat=as.data.frame(cbind(sim@subj,sim@item,sim@cond,sim@Scond,sim@lag,sim@resp))
colnames(dat)=c("sub","item","cond","Scond","lag","resp")
# Set parameters and data generation

```r
cats = 6;  # number of cats
print(cats)
print(nrow(data))
print(ncol(data))
M=10  #Way too low for real analysis
keep=2:M
uvsd=uvsdSample(dat,M=M,keep=keep,equalVar=FALSE,freeSig2=TRUE,jump=.0001,Hier=1)
```

# Visualize the chains

```r
par(mfrow=c(3,2),pch=19,pty='s')
#Look at chains of MuN and MuS
matplot(uvsd@blockN[,uvsd@muN],t='l',xlab="Iteration",ylab="Mu-N")
abline(h=sim@muN,col="blue")
matplot(uvsd@blockS[,uvsd@muS],t='l',xlab="Iteration",ylab="Mu-S")
abline(h=sim@muS,col="blue")
```

# Estimates of strength effects as function of true values

```r
plot(uvsd@estN[uvsd@alphaN]-sim@alphaN,xlab="True Alpha-N",ylab="Est. Alpha-N");abline(0,1,col="blue")
plot(uvsd@estS[uvsd@alphaS]-sim@alphaS,xlab="True Alpha-S",ylab="Est. Alpha-S");abline(0,1,col="blue")
plot(uvsd@estN[uvsd@betaN]-sim@betaN,xlab="True Beta-N",ylab="Est. Beta-N");abline(0,1,col="blue")
plot(uvsd@estS[uvsd@betaS]-sim@betaS,xlab="True Beta-S",ylab="Est. Beta-S");abline(0,1,col="blue")
```

# Sigma^2 effects

```r
#Note that Sigma^2 is biased high with few participants and items. This bias.
#goes away with larger sample sizes.
par(mfrow=c(2,2),pch=19,pty='s')
matplot(sqrt(exp(uvsd@blockS2[,uvsd@muS])),t='l',xlab="Iteration",ylab="Mu-Sigma2")
abline(h=sqrt(exp(sim@muS2)),col="blue")
```

# Look at some criteria

```r
par(mfrow=c(2,2))
for(i in 1:4)
matplot(t(uvsd@s.crit[i,,]),t='l')
```

---

### uvsdSim Function

**Description**

Simulates data from a hierarchical UVSD model.
uvsdSim

Usage

uvsdSim(NN = 2, NS = 1, I = 30, J = 200, K = 6, muN = c(-0.5, -0.2), s2aN = 0.2, s2bN = 0.2, muS = 0.5, s2aS = 0.2, s2bS = 0.2, muS2 = log(1), s2aS2 = 0, s2bS2 = 0, lagEffect = -0.001, crit = matrix(rep(c(-1.5,-0.5, 0, 0.5, 1.5), each = I), ncol = (K - 1)))

Arguments

NN  Number of conditions for new words.
NS  Number of conditions for studied words.
I   Number of participants.
J   Number of items.
K   Number of response options.
muN Mean of new-item distribution. If NN is greater than 1, then muN must be a vector of length NN.
s2aN Variance of participant effects on mean of new-item distribution.
s2bN Variance of item effects on mean of new-item distribution.
muS Mean of studied-item distribution. If NS is greater than 1, then muS must be a vector of length NS.
s2aS Variance of participant effects on mean of studied-item distribution.
s2bS Variance of item effects on mean of studied-item distribution.
lagEffect Magnitude of linear lag effect on both studied-item distribution and log(sigma2).
muS2 Mean variance of studied-item distribution, sigma2
s2aS2 Variance of participant effects sigma2.
s2bS2 Variance of item effects on sigma2.
crit Matrix of criteria (not including -Inf or Inf). Columns correspond to criteria, rows correspond to participants.

Value

The function returns an internally defined "uvsdSim" structure.

Author(s)

Michael S. Pratte

References

See Pratte, Rouder, & Morey (2009)

See Also

hbmem
Examples

library(hbmem)
#Data from hiererchial model
sim=uvsdSim()
slotNames(sim)
table(sim@resp,sim@Scond,sim@cond)

#Usefull to make data.frame for passing to model-fitting functions
dat=as.data.frame(cbind(sim@subj,sim@item,sim@cond,sim@Scond,sim@lag,sim@resp))
colnames(dat)=c("sub","item","cond","Scond","lag","resp")

table(dat$resp,dat$Scond,dat$cond)
Index

* datasets
  prm09, 14

* models
  dpsdSample, 4
dpsdSim, 6
gammaSample, 10

* package
  hbmem-package, 2
dpsdSample, 4
dpsdSim, 6

gammaLikeSample, 8
gammaSample, 10
gammaSim, 13

hbmem (hbmem-package), 2
hbmem-package, 2

prm09, 14
r tgamma, 15

sampleGamma, 16

uvxsdSample, 18
uvxsdSim, 21