# Package ‘dina’

April 26, 2017

**Type** Package

**Title** Bayesian Estimation of DINA Model

**Version** 1.0.2

**Date** 2017-04-26

**Description** Estimate the Deterministic Input, Noisy "And" Gate (DINA) cognitive diagnostic model parameters using the Gibbs sampler described by Culpepper (2015) <doi:10.3102/1076998615595403>.

**URL** [https://github.com/tmsalab/dina](https://github.com/tmsalab/dina)

**BugReports** [https://github.com/tmsalab/dina/issues](https://github.com/tmsalab/dina/issues)

**License** GPL (>= 2)

**Imports** Rcpp (>= 0.12.10)

**LinkingTo** Rcpp (>= 0.12.10), RcppArmadillo (>= 0.7.800)

**Depends** R (>= 3.0.2)

**RoxygenNote** 6.0.1

**NeedsCompilation** yes

**Author** Steven Andrew Culpepper [aut, cph, cre]

**Maintainer** Steven Andrew Culpepper <sculpepp@illinois.edu>

**Repository** CRAN

**Date/Publication** 2017-04-26 16:26:39 UTC

---

## R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>dina-package</td>
<td>2</td>
</tr>
<tr>
<td>DINAsim</td>
<td>2</td>
</tr>
<tr>
<td>DINA_Gibbs</td>
<td>4</td>
</tr>
<tr>
<td>rDirichlet</td>
<td>7</td>
</tr>
<tr>
<td>rmultinomial</td>
<td>7</td>
</tr>
<tr>
<td>update_alpha</td>
<td>8</td>
</tr>
<tr>
<td>update_sg</td>
<td>9</td>
</tr>
</tbody>
</table>

**Index** 10
Description


Author(s)

Maintainer: Steven Andrew Culpepper <sculpepp@illinois.edu> [copyright holder]

See Also

Useful links:

- [https://github.com/tmsalab/dina](https://github.com/tmsalab/dina)
- Report bugs at [https://github.com/tmsalab/dina/issues](https://github.com/tmsalab/dina/issues)

DINAsim

Description

Simulation Responses from the DINA model

Sample responses from the DINA model for given attribute profiles, Q matrix, and item parmeters. Returns a matrix of dichotomous responses generated under DINA model.

Usage

DINAsim(alphas, Q, ss, gs)

Arguments

alphas A N by K matrix of latent attributes.
Q A N by K matrix indicating which skills are required for which items.
ss A J vector of item slipping parameters.
gs A J vector of item guessing parameters.

Value

A N by J matrix of responses from the DINA model.

Author(s)

Steven Andrew Culpepper
DINAsim

See Also

DINA_Gibbs

Examples

```r
# de la Torre (2009) Simulation Replication
N = 200
J = 30
delta0 = rep(1,2*K)

# Creating Q matrix
Q = matrix(rep(diag(K),2),2*K,K,byrow=TRUE)
for (mm in 2:K){
  temp = combn(1:K,m=mm)
  tempmat = matrix(0,ncol(temp),K)
  for (j in 1:ncol(temp)) tempmat[j,temp[,j]] = 1
  Q = rbind(Q,tempmat)
}
Q = Q[1:J,]

# Setting item parameters and generating attribute profiles
ss = gs = rep(.2,J)
PIs = rep(1/(2*K),2*K)
CLs = c((1:(2*K))%*%multinom(n=N,size=1,prob=PIs) )

# Defining matrix of possible attribute profiles
As = rep(0,K)
for (j in 1:K){
  temp = combn(1:K,m=j)
  tempmat = matrix(0,ncol(temp),K)
  for (j in 1:ncol(temp)) tempmat[j,temp[,j]] = 1
  As = rbind(As,tempmat)
}
As = as.matrix(As)

# Sample true attribute profiles
Alphas = As[CLs,]

# Simulate data under DINA model
gen = DINAsim(Alphas,Q,ss,gs)
Y_sim = gen$Y

# Execute MCMC
# NOTE: small chain length used to reduce computation time for pedagogical example.
chainLength = 200
burnin = 100
outchain <- DINA_Gibbs(Y_sim, Amat = As, Q, chain_length = chainLength)

# Summarize posterior samples for g and l-s
```
mGs = apply(outchain$GamS[,burnin:chain_length],1,mean)
sGs = apply(outchain$GamS[,burnin:chain_length],1, sd)
m1mSS = 1-apply(outchain$SigS[,burnin:chain_length],1,mean)
s1mSS = apply(outchain$SigS[,burnin:chain_length],1, sd)
output = cbind(mGs,sGs,m1mSS,s1mSS)
colnames(output) = c('g Est','g SE','1-s Est','1-s SE')
rownames(output) = paste0('Item ',1:J)
print(output,digits=3)

# Summarize marginal skill distribution using posterior samples for latent class proportions
PIoutput = cbind(apply(outchain$Pis,1,mean),apply(outchain$Pis,1, sd))
colnames(PIoutput) = c('EST','SE')
rownames(PIoutput) = apply(As,1,paste0,collapse='')
print(PIoutput,digits=3)

---

DINA_Gibbs

Generate Posterior Distribution with Gibbs sampler

Description

Function for sampling parameters from full conditional distributions. The function returns a list of arrays or matrices with parameter posterior samples. Note that the output includes the posterior samples in objects named: CLASSES = individual attribute profiles, PIs = latent class proportions, SigS = item slipping parameters, and GamS = item guessing parameters.

Usage

DINA_Gibbs(Y, Amat, Q, chain_length = 10000L)

Arguments

Y A N by J matrix of observed responses.
Amat A C by K matrix of latent classes.
Q A N by K matrix indicating which skills are required for which items.
chain_length Number of MCMC iterations.

Value

A list with samples from the posterior distribution.

Author(s)

Steven Andrew Culpepper

See Also

DINAsim
Examples

```r
## Not run:
#Tatsuoka Fraction Subtraction Data

require(CDM)
data(fraction.subtraction.data)
Y_1984 = as.matrix(fraction.subtraction.data)
Q_1984 = as.matrix(fraction.subtraction.qmatrix)
K_1984 = ncol(fraction.subtraction.qmatrix)
J_1984 = ncol(Y_1984)

# Creating matrix of possible attribute profiles
As_1984 = rep(0,K_1984)
for(j in 1:K_1984){
    temp = combn(1:K_1984,m=j)
    tempmat = matrix(0,ncol(temp),K_1984)
    for(j in 1:ncol(temp)) tempmat[temp[,j]] = 1
    As_1984 = rbind(As_1984,tempmat)
}
As_1984 = as.matrix(As_1984)

# Generate samples from posterior distribution
# May take 8 minutes
chainLength <- 5000
burnin <- 1000
outchain_1984 <- DINA_Gibbs(Y = Y_1984, Amat = As_1984,
Q_1984, chain_length = chainLength)

# Summarize posterior samples for g and l-s
mgs_1984 = apply(outchain_1984$GamS[,burnin:chainLength],1,mean)
sgs_1984 = apply(outchain_1984$GamS[,burnin:chainLength],1,sd)
mss_1984 = 1-apply(outchain_1984$SigS[,burnin:chainLength],1,mean)
sss_1984 = apply(outchain_1984$SigS[,burnin:chainLength],1,sd)
colnames(output_1984) = c('g Est','g SE','l-s Est','l-s SE')
rownames(output_1984) = colnames(Y_1984)
print(output_1984,digits=3)

# Summarize marginal skill distribution using posterior samples for latent class proportions
marg_PIs = t(outchain_1984$Pi)
PI_Est = apply(marg_PIs[,burnin:chainLength],1,mean)
PI_Sd = apply(marg_PIs[,burnin:chainLength],1,sd)
PIoutput = cbind(PI_Est,PI_Sd)
colnames(PIoutput) = c('EST','SE')
rownames(PIoutput) = paste0('Skill ',1:K_1984)
print(PIoutput,digits=3)
```

# de la Torre (2009) Simulation Replication w/ N = 200

N = 200
K = 5
J = 30
delta0 = rep(1,2*K)

#Creating Q matrix
Q = matrix(rep(diag(K),2),2*K,K,byrow=TRUE)
for(mm in 2:K){
    temp = combn(1:K,m=mm)
    tempmat = matrix(0,ncol(temp),K)
    for(j in 1:ncol(temp)) tempmat[j,temp[,j]] = 1
    Q = rbind(Q,tempmat)
}
Q = Q[1:J,]

# Setting item parameters and generating attribute profiles
ss = gs = rep(1.2,J)
PIs = rep(1/((2*K)),2*K)
CLS = c((1:(2*K))%%*%%rmultinom(n=N,size=1,prob=PIs))

# Defining matrix of possible attribute profiles
As = rep(0,K)
for(j in 1:K){
    temp = combn(1:K,m=j)
    tempmat = matrix(0,ncol(temp),K)
    for(j in 1:ncol(temp)) tempmat[j,temp[,j]] = 1
    As = rbind(As,tempmat)
}
As = as.matrix(As)

# Sample true attribute profiles
Alphas = As[CLS,]

# Simulate data under DINA model
gen = DINA_sim(Alphas,Q,ss,gs)
Y_sim = gen$Y

# Execute MCMC
# NOTE small chain length used to reduce computation time for pedagogical example.
chainLength = 200
burnin = 100

outchain <- DINA_Gibbs(Y_sim,Amat=As,Q,chain_length=chainLength)

# Summarize posterior samples for g and 1-s
mGs = apply(outchain$GamS[,burnin:chainLength],1,mean)
sGs = apply(outchain$GamS[,burnin:chainLength],1,sd)
m1mSS = 1 - apply(outchain$SigS[,burnin:chainLength],1,mean)
s1mSS = apply(outchain$SigS[,burnin:chainLength],1,sd)
output = cbind(mGs,sGs,m1mSS,s1mSS)
colnames(output) = c('g Est','g SE','1-s Est','1-s SE')
rownames(output) = paste0('Item ',1:J)
print(output, digits=3)
rDirichlet

Generate Dirichlet Random Variable

Description
Sample a Dirichlet random variable.

Usage
rDirichlet(deltas)

Arguments

deltas A vector of Dirichlet parameters.

Value
A vector from a Dirichlet.

Author(s)
Steven Andrew Culpepper

rmultinomial
Generate Multinomial Random Variable

Description
Sample a multinomial random variable for given probabilities.

Usage
rmultinomial(ps)

Arguments

ps A vector for the probability of each category.
update_alpha

Value
A vector from a multinomial with probability ps.

Author(s)
Steven Andrew Culpepper

update_alpha Update attributes and latent class probabilities

Description
Update attributes and latent class probabilities by sampling from full conditional distribution.

Usage
update_alpha(Amat, Q, ss, gs, Y, PIs, ALPHAS, delta0)

Arguments
Amat A C by K matrix of latent classes.
Q A N by K matrix indicating which skills are required for which items.
ss A J vector of item slipping parameters.
gs A J vector of item guessing parameters.
Y A N by J matrix of observed responses.
PIS A C vector of latent class probabilities.
ALPHAS A N by K matrix of latent attributes.
delta0 A J vector of Dirichlet prior parameters.

Value
A N by K matrix of attributes and a C vector of class probabilities.

Author(s)
Steven Andrew Culpepper
**Description**

Update guessing and slipping parameters from full conditional distribution.

**Usage**

```r
update_sg(Y, Q, ALPHAS, ss_old, as0, bs0, ag0, bg0)
```

**Arguments**

- `Y`: A N by J matrix of observed responses.
- `Q`: A N by K matrix indicating which skills are required for which items.
- `ALPHAS`: A N by K matrix of latent attributes.
- `ss_old`: A J vector of item slipping parameters from prior iteration.
- `as0`: Slipping prior alpha parameter for Beta distribution.
- `bs0`: Slipping prior beta parameter for Beta distribution.
- `ag0`: Guessing prior alpha parameter for Beta distribution.
- `bg0`: Guessing prior beta parameter for Beta distribution.

**Value**

A list with two J vectors of guessing and slipping parameters.

**Author(s)**

Steven Andrew Culpepper
Index

dina (dina-package), 2
dina-package, 2
DINA_Gibbs, 3, 4
DINAsim, 2, 4

rDirichlet, 7
rmultinomial, 7

update_alpha, 8
update_sg, 9