

The bayesm Package

November 10, 2005

Version 2.0-2

Date 2005-11-10

Title Bayesian Inference for Marketing/Micro-econometrics

Author Peter Rossi <peter.rossi@ChicagoGsb.edu>, Rob McCulloch
<robert.mcculloch@ChicagoGsb.edu>.

Maintainer Peter Rossi <peter.rossi@chicagosb.edu>

Depends R (>= 1.8.1)

Description bayesm covers many important models used in marketing and micro-econometrics applications. The package includes: Bayes Regression (univariate or multivariate dep var), Bayes Seemingly Unrelated Regression (SUR), Multinomial Logit (MNL) and Multinomial Probit (MNP), Multivariate Probit, Negative Binomial (Poisson) Regression, Multivariate Mixtures of Normals (including clustering), Hierarchical Linear Models with normal prior and covariates, Hierarchical Multinomial Logits with mixture of normals prior and covariates, Hierarchical Negative Binomial Regression Models, Bayesian analysis of choice-based conjoint data, Bayesian treatment of linear instrumental variables models, and Analysis of Multivariate Ordinal survey data with scale usage heterogeneity (as in Rossi et al, JASA (01)). For further reference, consult our book, Bayesian Statistics and Marketing by Rossi, Allenby and McCulloch.

License GPL (version 2 or later)

URL <http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

R topics documented:

Scotch	3
bank	4
breg	6
cgetC	8
cheese	9
clusterMix	10
condMom	12
createX	13
customerSat	14
detailing	15
eMixMargDen	17
fsh	18

ghkvec	19
init.rmultiregfp	20
llmnl	21
llmnp	22
llnhlogit	23
lndIChisq	24
lndIWishart	25
lndMvn	26
lndMvst	27
logMargDenNR	28
margarine	29
mixDen	31
mixDenBi	32
mnlHess	34
mnpProb	35
momMix	36
nmat	37
numEff	38
rbiNormGibbs	39
rbprobitGibbs	40
rdirichlet	41
rhierBinLogit	42
rhierLinearModel	44
rhierMnlRwMixture	46
rhierNegbinRw	50
rivGibbs	53
rmixGibbs	55
rmixture	56
rmnlIndepMetrop	57
rmnpGibbs	59
rmultireg	60
rmultiregfp	62
rmvpGibbs	63
rmvst	65
rnegbinRw	66
rmixGibbs	68
rscaleUsage	71
rsurGibbs	73
rtrun	75
runireg	76
runiregGibbs	77
rwishart	78
simmnl	79
simmnlwX	80
simmnp	81
simmvp	82
simnhlogit	83

Scotch

Survey Data on Brands of Scotch Consumed

Description

from Simmons Survey. Brands used in last year for those respondents who report consuming scotch.

Usage

```
data(Scotch)
```

Format

A data frame with 2218 observations on the following 21 variables. All variables are coded 1 if consumed in last year, 0 if not.

Chivas.Regal a numeric vector
Dewar.s.White.Label a numeric vector
Johnnie.Walker.Black.Label a numeric vector
J...B a numeric vector
Johnnie.Walker.Red.Label a numeric vector
Other.Brands a numeric vector
Glenlivet a numeric vector
Cutty.Sark a numeric vector
Glenfiddich a numeric vector
Pinch..Haig. a numeric vector
Clan.MacGregor a numeric vector
Ballantine a numeric vector
Macallan a numeric vector
Passport a numeric vector
Black...White a numeric vector
Scoresby.Rare a numeric vector
Grants a numeric vector
Ushers a numeric vector
White.Horse a numeric vector
Knockando a numeric vector
the.Singleton a numeric vector

Source

Edwards, Y. and G. Allenby (2003), "Multivariate Analysis of Multiple Response Data," *JMR* 40, 321-334.

References

Chapter 4, *Bayesian Statistics and Marketing* by Rossi et al.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```

data(Scotch)
cat(" Frequencies of Brands", fill=TRUE)
mat=apply(as.matrix(Scotch),2,mean)
print(mat)
##
## use Scotch data to run Multivariate Probit Model
##
if(nchar(Sys.getenv("LONG_TEST")) != 0){
##

y=as.matrix(Scotch)
p=ncol(y); n=nrow(y)
dimnames(y)=NULL
y=as.vector(t(y))
y=as.integer(y)
I_p=diag(p)
X=rep(I_p,n)
X=matrix(X,nrow=p)
X=t(X)

R=2000
Data=list(p=p,X=X,y=y)
Mcmc=list(R=R)
set.seed(66)
out=rmvpGibbs(Data=Data,Mcmc=Mcmc)

ind=(0:(p-1))*p + (1:p)
cat(" Betadraws ",fill=TRUE)
mat=apply(out$betadraw/sqrt(out$sigmadraw[,ind]),2,quantile,probs=c(.01,.05,.5,.95,.99))
print(mat)
rdraw=matrix(double((R)*p*p),ncol=p*p)
rdraw=t(apply(out$sigmadraw,1,nmat))
cat(" Draws of Correlation Matrix ",fill=TRUE)
mat=apply(rdraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
## correlation matrix too large to print -- summarize
quantile(round(mat,digits=2))

}

```

bank

Bank Card Conjoint Data of Allenby and Ginter (1995)

Description

Data from a conjoint experiment in which two partial profiles of credit cards were presented to 946 respondents. The variable `bank$choiceAtt$choice` indicates which profile was chosen. The profiles are coded as the difference in attribute levels. Thus, a "-1" means the profile coded as a choice of "0" has the attribute. A value of 0 means that the attribute was not present in the comparison.

data on age, income and gender (female=1) are also recorded in `bank$demo`

Usage

```
data(bank)
```

Format

This R object is a list of two data frames, `list(choiceAtt,demo)`.

List of 2

\$ choiceAtt: 'data.frame': 14799 obs. of 16 variables:

```
...$ id : int [1:14799] 1 1 1 1 1 1 1 1 1 1
...$ choice : int [1:14799] 1 1 1 1 1 1 1 1 0 1
...$ Med_FInt : int [1:14799] 1 1 1 0 0 0 0 0 0 0
...$ Low_FInt : int [1:14799] 0 0 0 0 0 0 0 0 0 0
...$ Med_VInt : int [1:14799] 0 0 0 0 0 0 0 0 0 0
...$ Rewrd_2 : int [1:14799] -1 1 0 0 0 0 0 1 -1 0
...$ Rewrd_3 : int [1:14799] 0 -1 1 0 0 0 0 0 1 -1
...$ Rewrd_4 : int [1:14799] 0 0 -1 0 0 0 0 0 0 1
...$ Med_Fee : int [1:14799] 0 0 0 1 1 -1 -1 0 0 0
...$ Low_Fee : int [1:14799] 0 0 0 0 0 1 1 0 0 0
...$ Bank_B : int [1:14799] 0 0 0 -1 1 -1 1 0 0 0
...$ Out_State : int [1:14799] 0 0 0 0 -1 0 -1 0 0 0
...$ Med_Rebate : int [1:14799] 0 0 0 0 0 0 0 0 0 0
...$ High_Rebate : int [1:14799] 0 0 0 0 0 0 0 0 0 0
...$ High_CredLine: int [1:14799] 0 0 0 0 0 0 0 -1 -1 -1
...$ Long_Grace : int [1:14799] 0 0 0 0 0 0 0 0 0 0
```

\$ demo : 'data.frame': 946 obs. of 4 variables:

```
...$ id : int [1:946] 1 2 3 4 6 7 8 9 10 11
...$ age : int [1:946] 60 40 75 40 30 30 50 50 50 40
...$ income: int [1:946] 20 40 30 40 30 60 50 100 50 40
...$ gender: int [1:946] 1 1 0 0 0 0 1 0 0 0
```

Details

Each respondent was presented with between 13 and 17 paired comparisons. Thus, this dataset has a panel structure.

Source

Allenby and Ginter (1995), "Using Extremes to Design Products and Segment Markets," *JMR*, 392-403.

References

Appendix A, *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```
data(bank)
cat(" table of Binary Dep Var", fill=TRUE)
print(table(bank$choiceAtt[,2]))
cat(" table of Attribute Variables", fill=TRUE)
```

```

mat=apply(as.matrix(bank$choiceAtt[,3:16]),2,table)
print(mat)
cat(" means of Demographic Variables",fill=TRUE)
mat=apply(as.matrix(bank$demo[,2:3]),2,mean)
print(mat)

## example of processing for use with rhierBinLogit
##
if(nchar(Sys.getenv("LONG_TEST")) != 0)
{
choiceAtt=bank$choiceAtt
Z=bank$demo

## center demo data so that mean of random-effects
## distribution can be interpreted as the average respondents

Z[,1]=rep(1,nrow(Z))
Z[,2]=Z[,2]-mean(Z[,2])
Z[,3]=Z[,3]-mean(Z[,3])
Z[,4]=Z[,4]-mean(Z[,4])
Z=as.matrix(Z)

hh=levels(factor(choiceAtt$id))
nhh=length(hh)
lgtdata=NULL
for (i in 1:nhh) {
  y=choiceAtt[choiceAtt[,1]==hh[i],2]
  nobs=length(y)
  X=as.matrix(choiceAtt[choiceAtt[,1]==hh[i],c(3:16)])
  lgtdata[[i]]=list(y=y,X=X)
}

cat("Finished Reading data",fill=TRUE)
fsh()

Data=list(lgtdata=lgtdata,Z=Z)
Mcmc=list(R=10000,sbeta=0.2,keep=20)
set.seed(66)
out=rhierBinLogit(Data=Data,Mcmc=Mcmc)

cat(" Deltadraws ",fill=TRUE)
mat=apply(out$Deltadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
print(mat)
cat(" Vbetadraws ",fill=TRUE)
mat=apply(out$Vbetadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
print(mat)
}

```

Description

breg makes one draw from the posterior of a univariate regression (scalar dependent variable) given the error variance = 1.0. A natural conjugate, normal prior is used.

Usage

```
breg(y, X, betabar, A)
```

Arguments

y	vector of values of dep variable.
X	n (length(y)) x k Design matrix.
betabar	k x 1 vector. Prior mean of regression coefficients.
A	Prior precision matrix.

Details

model: $y = x'\beta + e$. $e \sim N(0, 1)$.

prior: $\beta \sim N(\text{betabar}, A^{-1})$.

Value

k x 1 vector containing a draw from the posterior distribution.

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

In particular, X must be a matrix. If you have a vector for X, coerce it into a matrix with one column

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```
##

if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=1000} else {R=10}

## simulate data
set.seed(66)
n=100
X=cbind(rep(1,n),runif(n)); beta=c(1,2)
y=X%*%beta+rnorm(n)
##
## set prior
```

```

A=diag(c(.05,.05)); betabar=c(0,0)
##
## make draws from posterior
betadraw=matrix(double(R*2),ncol=2)
for (rep in 1:R) {betadraw[rep,]=breg(y,X,betabar,A)}
##
## summarize draws
mat=apply(betadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(beta,mat); rownames(mat)[1]="beta"; print(mat)

```

cgetC

Obtain A List of Cut-offs for Scale Usage Problems

Description

cgetC obtains a list of censoring points, or cut-offs, used in the ordinal multivariate probit model of Rossi et al (2001). This approach uses a quadratic parameterization of the cut-offs. The model is useful for modeling correlated ordinal data on a scale from 1, ..., k with different scale usage patterns.

Usage

```
cgetC(e, k)
```

Arguments

e	quadratic parameter (>0 and less than 1)
k	items are on a scale from 1, ..., k

Value

A vector of k+1 cut-offs.

Warning

This is a utility function which implements **no** error-checking.

Author(s)

Rob McCulloch and Peter Rossi, Graduate School of Business, University of Chicago. Peter.Rossi@ChicagoGsb.edu.

References

Rossi et al (2001), "Overcoming Scale Usage Heterogeneity," *JASA*96, 20-31.

See Also

[rscaleUsage](#)

Examples

```
##
cgetC(.1,10)
```


cheese

*Sliced Cheese Data***Description**

Panel data with sales volume for a package of Borden Sliced Cheese as well as a measure of display activity and price. Weekly data aggregated to the "key" account or retailer/market level.

Usage

```
data(cheese)
```

Format

A data frame with 5555 observations on the following 4 variables.

RETAILER a list of 88 retailers

VOLUME unit sales

DISP a measure of display activity – per cent ACV on display

PRICE in \$

Source

Boatwright et al (1999), "Account-Level Modeling for Trade Promotion," *JASA* 94, 1063-1073.

References

Chapter 3, *Bayesian Statistics and Marketing* by Rossi et al.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```
data(cheese)
cat(" Quantiles of the Variables ",fill=TRUE)
mat=apply(as.matrix(cheese[,2:4]),2,quantile)
print(mat)

##
## example of processing for use with rhierLinearModel
##
if(nchar(Sys.getenv("LONG_TEST")) != 0)
{

retailer=levels(cheese$RETAILER)
nreg=length(retailer)
nvar=3
regdata=NULL
for (reg in 1:nreg) {
  y=log(cheese$VOLUME[cheese$RETAILER==retailer[reg]])
  iota=c(rep(1,length(y)))
  X=cbind(iota,cheese$DISP[cheese$RETAILER==retailer[reg]],
          log(cheese$PRICE[cheese$RETAILER==retailer[reg]]))
  regdata[[reg]]=list(y=y,X=X)
```

```

}
Z=matrix(c(rep(1,nreg)),ncol=1)
nz=ncol(Z)
##
## run each individual regression and store results
##
lscoef=matrix(double(nreg*nvar),ncol=nvar)
for (reg in 1:nreg) {
  coef=lsfit(regdata[[reg]]$X,regdata[[reg]]$y,intercept=FALSE)$coef
  if (var(regdata[[reg]]$X[,2])==0) { lscoef[reg,1]=coef[1]; lscoef[reg,3]=coef[2]
  else {lscoef[reg,]=coef }
}

R=2000
Data=list(regdata=regdata,Z=Z)
Mcmc=list(R=R,keep=1)

betamean=array(double(nreg*nvar),dim=c(nreg,nvar))
burnin=100

set.seed(66)
out=rhierLinearModel(Data=Data,Mcmc=Mcmc)

for (k in 1:nvar) { betamean[,k]=apply(out$betadraw[,k,burnin:R],1,mean)}
print(betamean)
cat(" Deltadraws ",fill=TRUE)
mat=apply(out$Deltadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
print(mat)
cat(" Vbetadraws ",fill=TRUE)
mat=apply(out$Vbetadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
print(mat)

if(0){
coefn=c("Intercept","Display","LnPrice")
colors=c("blue","green","red","yellow")
par(mfrow=c(nvar,1),mar=c(5.1,15,4.1,13))
for (n in 1:nvar)
{
  plot(range(betamean[,n]),range(betamean[,n]),
       type="n",main=coefn[n],xlab="ls coef",ylab="post mean")
  points(lscoef[,n],betamean[,n],pch=17,col="blue",cex=1.2)
  abline(c(0,1))
}
}

}

```

clusterMix

Cluster Observations Based on Indicator MCMC Draws

Description

clusterMix uses MCMC draws of indicator variables from a normal component mixture model to cluster observations based on a similarity matrix.

Usage

```
clusterMix(zdraw, cutoff = 0.9, SILENT = FALSE)
```

Arguments

zdraw	R x nobs array of draws of indicators
cutoff	cutoff probability for similarity (def=.9)
SILENT	logical flag for silent operation (def= FALSE)

Details

define a similarity matrix, Sim, $\text{Sim}[i,j]=1$ if observations i and j are in same component. Compute the posterior mean of Sim over indicator draws.

clustering is achieved by two means:

Method A: Find the indicator draw whose similarity matrix minimizes, $\text{loss}(E[\text{Sim}]-\text{Sim}(z))$, where loss is absolute deviation.

Method B: Define a Similarity matrix by setting any element of $E[\text{Sim}] = 1$ if $E[\text{Sim}] > \text{cutoff}$. Compute the clustering scheme associated with this "windsorized" Similarity matrix.

Value

clustera	indicator function for clustering based on method A above
clusterb	indicator function for clustering based on method B above

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago (Peter.Rossi@ChicagoGsb.edu).

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch Chapter 3.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rnmixGibbs](#)

Examples

```
##
if(nchar(Sys.getenv("LONG_TEST")) != 0)
{
  ## simulate data from mixture of normals
  n=500
  pvec=c(.5,.5)
  mul=c(2,2)
```

```

mu2=c(-2,-2)
Sigma1=matrix(c(1,.5,.5,1),ncol=2)
Sigma2=matrix(c(1,.5,.5,1),ncol=2)
comps=NULL
comps[[1]]=list(mu1,backsolve(chol(Sigma1),diag(2)))
comps[[2]]=list(mu2,backsolve(chol(Sigma2),diag(2)))
dm=rmixture(n,pvec,comps)
## run MCMC on normal mixture
R=2000
Data=list(y=dm$x)
ncomp=2
Prior=list(ncomp=ncomp,a=c(rep(100,ncomp)))
Mcmc=list(R=R,keep=1)
out=rmixGibbs(Data=Data,Prior=Prior,Mcmc=Mcmc)
begin=500
end=R
## find clusters
outclusterMix=clusterMix(out$zdraw[begin:end,])
##
## check on clustering versus "truth"
## note: there could be switched labels
##
table(outclusterMix$clustera,dm$z)
table(outclusterMix$clusterb,dm$z)
}
##

```

condMom

Computes Conditional Mean/Var of One Element of MVN given All Others

Description

condMom compute moments of conditional distribution of ith element of normal given all others.

Usage

```
condMom(x, mu, sigi, i)
```

Arguments

x	vector of values to condition on - ith element not used
mu	length(x) mean vector
sigi	length(x)-dim covariance matrix
i	conditional distribution of ith element

Details

$x \sim MVN(\mu, \Sigma)$.

condMom computes moments of x_i given x_{-i} .

Value

a list containing:

cmean	cond mean
cvar	cond variance

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```
##
sig=matrix(c(1,.5,.5,.5,1,.5,.5,.5,1),ncol=3)
sigi=chol2inv(chol(sig))
mu=c(1,2,3)
x=c(1,1,1)
condMom(x,mu,sigi,2)
```

createX

Create X Matrix for Use in Multinomial Logit and Probit Routines

Description

createX makes up an X matrix in the form expected by Multinomial Logit ([rmnlIndepMetrop](#) and [rhierMnlRwMixture](#)) and Probit ([rmnpGibbs](#) and [rmvpGibbs](#)) routines. Requires an array of alternative specific variables and/or an array of "demographics" or variables constant across alternatives which may vary across choice occasions.

Usage

```
createX(p, na, nd, Xa, Xd, INT = TRUE, DIFF = FALSE, base = p)
```

Arguments

p	integer - number of choice alternatives
na	integer - number of alternative-specific vars in Xa
nd	integer - number of non-alternative specific vars
Xa	n x p*na matrix of alternative-specific vars
Xd	n x nd matrix of non-alternative specific vars

INT logical flag for inclusion of intercepts
 DIFF logical flag for differencing wrt to base alternative
 base integer - index of base choice alternative
 note: na,nd,Xa,Xd can be NULL to indicate lack of Xa or Xd variables.

Value

X matrix – $n \times (p - \text{DIFF}) \times [(\text{INT} + \text{nd}) \times (p - 1) + \text{na}]$ matrix.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rmnlIndepMetrop](#), [rmnpGibbs](#)

Examples

```
na=2; nd=1; p=3
vec=c(1,1.5,.5,2,3,1,3,4.5,1.5)
Xa=matrix(vec,byrow=TRUE,ncol=3)
Xa=cbind(Xa,-Xa)
Xd=matrix(c(-1,-2,-3),ncol=1)
createX(p=p,na=na,nd=nd,Xa=Xa,Xd=Xd)
createX(p=p,na=na,nd=nd,Xa=Xa,Xd=Xd,base=1)
createX(p=p,na=na,nd=nd,Xa=Xa,Xd=Xd,DIFF=TRUE)
createX(p=p,na=na,nd=nd,Xa=Xa,Xd=Xd,DIFF=TRUE,base=2)
createX(p=p,na=na,nd=NULL,Xa=Xa,Xd=NULL)
createX(p=p,na=NULL,nd=nd,Xa=NULL,Xd=Xd)
```

customerSat

Customer Satisfaction Data

Description

Responses to a satisfaction survey for a Yellow Pages advertising product. All responses are on a 10 point scale from 1 to 10 (10 is "Excellent" and 1 is "Poor")

Usage

```
data(customerSat)
```

Format

A data frame with 1811 observations on the following 10 variables.

- q1 Overall Satisfaction
- q2 Setting Competitive Prices
- q3 Holding Price Increase to a Minimum
- q4 Appropriate Pricing given Volume
- q5 Demonstrating Effectiveness of Purchase
- q6 Reach a Large # of Customers
- q7 Reach of Advertising
- q8 Long-term Exposure
- q9 Distribution
- q10 Distribution to Right Geographic Areas

Source

Rossi et al (2001), "Overcoming Scale Usage Heterogeneity," *JASA* 96, 20-31.

References

Case Study 3, *Bayesian Statistics and Marketing* by Rossi et al.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```
data(customerSat)
apply(as.matrix(customerSat), 2, table)
```

detailing

Physician Detailing Data from Manchanda et al (2004)

Description

Monthly data on detailing (sales calls) on 1000 physicians. 23 mos of data for each Physician. Includes physician covariates. Dependent Variable (*scripts*) is the number of new prescriptions ordered by the physician for the drug detailed.

Usage

```
data(detailing)
```

Format

This R object is a list of two data frames, `list(counts,demo)`.

List of 2:

```
$ counts:'data.frame': 23000 obs. of 4 variables:
...$ id : int [1:23000] 1 1 1 1 1 1 1 1 1 1
...$ scripts : int [1:23000] 3 12 3 6 5 2 5 1 5 3
...$ detailing : int [1:23000] 1 1 1 2 1 0 2 2 1 1
...$ lagged_scripts: int [1:23000] 4 3 12 3 6 5 2 5 1 5

$ demo :'data.frame': 1000 obs. of 4 variables:
...$ id : int [1:1000] 1 2 3 4 5 6 7 8 9 10
...$ generalphys : int [1:1000] 1 0 1 1 0 1 1 1 1 1
...$ specialist: int [1:1000] 0 1 0 0 1 0 0 0 0 0
...$ mean_samples: num [1:1000] 0.722 0.491 0.339 3.196 0.348
```

Details

`generalphys` is dummy for if doctor is a "general practitioner," `specialist` is dummy for if the physician is a specialist in the therapeutic class for which the drug is intended, `mean_samples` is the mean number of free drug samples given the doctor over the sample.

Source

Manchanda, P., P. K. Chintagunta and P. E. Rossi (2004), "Response Modeling with Non-Random Marketing Mix Variables," *Journal of Marketing Research* 41, 467-478.

Examples

```
data(detailing)
cat(" table of Counts Dep Var", fill=TRUE)
print(table(detailing$counts[,2]))
cat(" means of Demographic Variables",fill=TRUE)
mat=apply(as.matrix(detailing$demo[,2:4]),2,mean)
print(mat)

##
## example of processing for use with rhierNegbinRw
##
if(nchar(Sys.getenv("LONG_TEST")) != 0)
{

data(detailing)
counts = detailing$counts
Z = detailing$demo

# Construct the Z matrix
Z[,1] = 1
Z[,2]=Z[,2]-mean(Z[,2])
Z[,3]=Z[,3]-mean(Z[,3])
Z[,4]=Z[,4]-mean(Z[,4])
Z=as.matrix(Z)
id=levels(factor(counts$id))
nreg=length(id)
nobs = nrow(counts$id)
```



```

regdata=NULL
for (i in 1:nreg) {
  X = counts[counts[,1] == id[i],c(3:4)]
  X = cbind(rep(1,nrow(X)),X)
  y = counts[counts[,1] == id[i],2]
  X = as.matrix(X)
  regdata[[i]]=list(X=X, y=y)
}
nvar=ncol(X)          # Number of X variables
nz=ncol(Z)            # Number of Z variables
rm(detailing,counts)
cat("Finished Reading data",fill=TRUE)
fsh()

Data = list(regdata=regdata, Z=Z)
deltabar = matrix(rep(0,nvar*nz),nrow=nz)
Vdelta = 0.01 * diag(nz)
nu = nvar+3
V = 0.01*diag(nvar)
a = 0.5
b = 0.1
Prior = list(deltabar=deltabar, Vdelta=Vdelta, nu=nu, V=V, a=a, b=b)

R = 10000
keep =1
s_beta=2.93/sqrt(nvar)
s_alpha=2.93
c=2
Mcmc = list(R=R, keep = keep, s_beta=s_beta, s_alpha=s_alpha, c=c)
out = rhierNegbinRw(Data, Prior, Mcmc)

# Unit level mean beta parameters
Mbeta = matrix(rep(0,nreg*nvar),nrow=nreg)
ndraws = length(out$alphadraw)
for (i in 1:nreg) { Mbeta[i,] = rowSums(out$Betadraw[i, , ])/ndraws }

cat(" Deltadraws ",fill=TRUE)
mat=apply(out$Deltadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
print(mat)
cat(" Vbetadraws ",fill=TRUE)
mat=apply(out$Vbetadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
print(mat)
cat(" alphadraws ",fill=TRUE)
mat=apply(matrix(out$alphadraw),2,quantile,probs=c(.01,.05,.5,.95,.99))
print(mat)
}

```

eMixMargDen

Compute Marginal Densities of A Normal Mixture Averaged over MCMC Draws

Description

eMixMargDen assumes that a multivariate mixture of normals has been fitted via MCMC (using `rmixGibbs`). For each MCMC draw, the marginal densities for each component in the multivariate mixture are computed on a user-supplied grid and then averaged over draws.

Usage

```
eMixMargDen(grid, probdraw, compdraw)
```

Arguments

grid	array of grid points, grid[,i] are ordinates for ith component
probdraw	array - each row of which contains a draw of probabilities of mixture comp
compdraw	list of lists of draws of mixture comp moments

Details

length(compdraw) is number of MCMC draws.
 compdraw[[i]] is a list draws of mu and inv Chol root for each of mixture components.
 compdraw[[i]][[j]] is jth component. compdraw[[i]][[j]]\$mu is mean vector; compdraw[[i]][[j]]\$rooti is the UL decomp of Σ^{-1} .

Value

an array of the same dimension as grid with density values.

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type. To avoid errors, call with output from [rnmixGibbs](#).

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rnmixGibbs](#)

fsh

Flush Console Buffer

Description

Flush contents of console buffer. This function only has an effect on the Windows GUI.

Usage

```
fsh( )
```

Value

No value is returned.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

ghkvec

Compute GHK approximation to Multivariate Normal Integrals

Description

ghkvec computes the GHK approximation to the integral of a multivariate normal density over a half plane defined by a set of truncation points.

Usage

```
ghkvec(L, trunpt, above, r)
```

Arguments

L	lower triangular Cholesky root of Covariance matrix
trunpt	vector of truncation points
above	vector of indicators for truncation above(1) or below(0)
r	number of draws to use in GHK

Value

approximation to integral

Note

ghkvec can accept a vector of truncations and compute more than one integral. That is, length(trunpt)/length(above) number of different integrals, each with the same Sigma and mean 0 but different truncation points. See example below for an example with two integrals at different truncation points.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 2.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```
##

Sigma=matrix(c(1,.5,.5,1),ncol=2)
L=t(chol(Sigma))
trunpt=c(0,0,1,1)
above=c(1,1)
ghkvec(L,trunpt,above,100)
```

`init.rmultiregfp` *Initialize Variables for Multivariate Regression Draw*

Description

`init.rmultiregfp` initializes variables which can be pre-computed for draws from the posterior of a multivariate regression model. `init.rmultiregfp` should be called prior to use of `rmultiregfp`

Usage

```
init.rmultiregfp(X, A, Bbar, nu, V)
```

Arguments

<code>X</code>	Design matrix
<code>A</code>	Prior Precision matrix (m x k)
<code>Bbar</code>	Prior mean matrix (m x k)
<code>nu</code>	degrees of freedom parameter for Sigma prior
<code>V</code>	location parameter for Sigma prior

Details

model: $Y = XB + U$. $u_i \sim N(0, \Sigma)$. u_i is the i th row of U . Y is $n \times m$. X is $n \times k$. B is $k \times m$.
 priors: $\text{vec}(B) \sim N(\text{vec}(Bbar), \Sigma(x)A^{-1})$
 $\Sigma \sim IW(nu, V)$.

Value

A list containing ...

<code>IR</code>	Inverse of Cholesky Root of $(X'X + A)$
<code>RA</code>	Cholesky root of A
<code>RABbar</code>	$RA \%*\% Bbar$
<code>nu</code>	d.f. parm for IWishart prior
<code>V</code>	location matrix for IWishart prior

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 2.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rmultiregfp](#)

llmnl

*Evaluate Log Likelihood for Multinomial Logit Model***Description**

llmnl evaluates log-likelihood for the multinomial logit model.

Usage

```
llmnl(beta, y, X)
```

Arguments

beta	k x 1 coefficient vector
y	n x 1 vector of obs on y (1, ..., p)
X	n*p x k Design matrix (use createX to make)

Details

Let $\mu_{i,j} = X_i\beta$, then $Pr(y_i = j) = \exp(\mu_{i,j}) / \sum_k \exp(\mu_{i,k})$.
 X_i is the submatrix of X corresponding to the ith observation. X has n*p rows.
 Use [createX](#) to create X.

Value

value of log-likelihood (sum of log prob of observed multinomial outcomes).

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[createX](#), [rmnlIndepMetrop](#)

Examples

```
##
## Not run: ll=llmnl(beta,y,X)
```

llmnp

*Evaluate Log Likelihood for Multinomial Probit Model***Description**

llmnp evaluates the log-likelihood for the multinomial probit model.

Usage

```
llmnp(beta, Sigma, X, y, r)
```

Arguments

beta	k x 1 vector of coefficients
Sigma	(p-1) x (p-1) Covariance matrix of errors
X	X is n*(p-1) x k array. X is from differenced system.
y	y is vector of n indicators of multinomial response (1, ..., p).
r	number of draws used in GHK

Details

X is (p-1)*n x k matrix. Use [createX](#) with DIFF=TRUE to create X.

Model for each obs: $w = X\beta + e$. $e \sim N(0, \Sigma)$.

censoring mechanism:

if $y = j (j < p)$, $w_j > \max(w_{-j})$ and $w_j > 0$
 if $y = p$, $w < 0$

To use GHK, we must transform so that these are rectangular regions e.g. if $y = 1$, $w_1 > 0$ and $w_1 - w_{-1} > 0$.

Define A_j such that if $j=1, \dots, p-1$, $A_j w = A_j \mu + A_j e > 0$ is equivalent to $y = j$. Thus, if $y=j$, we have $A_j e > -A_j \mu$. Lower truncation is $-A_j \mu$ and $cov = A_j \Sigma A_j'$. For $j = p$, $e < -\mu$.

Value

value of log-likelihood (sum of log prob of observed multinomial outcomes).

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapters 2 and 4.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[createX](#), [rmnpGibbs](#)

Examples

```
##
## Not run: ll=llmnp(beta,Sigma,X,y,r)
```

llnhlogit

Evaluate Log Likelihood for non-homothetic Logit Model

Description

llmnp evaluates log-likelihood for the Non-homothetic Logit model.

Usage

```
llnhlogit(theta, choice, lnprices, Xexpend)
```

Arguments

theta	parameter vector (see details section)
choice	n x 1 vector of choice (1, ..., p)
lnprices	n x p array of log-prices
Xexpend	n x d array of vars predicting expenditure

Details

Non-homothetic logit model with: $\ln(\psi_i(U)) = \alpha_i - e^{k_i}U$

Structure of theta vector

alpha: (p x 1) vector of utility intercepts.

k: (p x 1) vector of utility rotation parms.

gamma: (k x 1) – expenditure variable coefs.

tau: (1 x 1) – logit scale parameter.

Value

value of log-likelihood (sum of log prob of observed multinomial outcomes).

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago (Peter.Rossi@ChicagoGsb.edu).

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 4.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[simnhlogit](#)

Examples

```
##
## Not run: ll=llnhlogit(theta,choice,lnprices,Xexpend)
```

lndIChisq

Compute Log of Inverted Chi-Squared Density

Description

lndIChisq computes the log of an Inverted Chi-Squared Density.

Usage

```
lndIChisq(nu, ssq, x)
```

Arguments

nu	d.f. parameter
ssq	scale parameter
x	ordinate for density evaluation

Details

$Z = \nu * ssq / \chi_{\nu}^2$, $Z \sim$ Inverted Chi-Squared.

lndIChisq computes the complete log-density, including normalizing constants.

Value

log density value

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, (Peter.Rossi@ChicagoGsb.edu).

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 2.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[dchisq](#)

Examples

```
##
lndIChisq(3,1,2)
```

lndIWishart	<i>Compute Log of Inverted Wishart Density</i>
-------------	--

Description

`lndIWishart` computes the log of an Inverted Wishart density.

Usage

```
lndIWishart(nu, S, IW)
```

Arguments

<code>nu</code>	d.f. parameter
<code>S</code>	"location" parameter
<code>IW</code>	ordinate for density evaluation

Details

$Z = Wishart(nu, V^{-1})^{-1}$, $Z \sim \text{Inverted Wishart}(nu, V)$.

`lndIWishart` computes the complete log-density, including normalizing constants.

Value

log density value

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 2.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

`rwishart`

Examples

```
##
lndIWishart(5,diag(3),(diag(3)+.5))
```

lndMvn

Compute Log of Multivariate Normal Density

Description

lndMvn computes the log of a Multivariate Normal Density.

Usage

```
lndMvn(x, mu, rooti)
```

Arguments

x	density ordinate
mu	mu vector
rooti	inv of Cholesky root of Sigma

Details

$z \sim N(\mu, \Sigma)$ note: does not include full normalizing constant

Value

log density value

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 2.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also[lndMvst](#)**Examples**

```
##
Sigma=matrix(c(1,.5,.5,1),ncol=2)
lndMvn(x=c(rep(0,2)),mu=c(rep(0,2)),rooti=backsolve(chol(Sigma),diag(2)))
```

lndMvst

*Compute Log of Multivariate Student-t Density***Description**

lndMvst computes the log of a Multivariate Student-t Density.

Usage

```
lndMvst(x, nu, mu, rooti)
```

Arguments

x	density ordinate
nu	d.f. parameter
mu	mu vector
rooti	inv of Cholesky root of Sigma

Details

$z \sim MVst(mu, nu, \Sigma)$ note: does not include full normalizing constant

Value

log density value

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 2.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also[lndMvn](#)**Examples**

```
##
Sigma=matrix(c(1,.5,.5,1),ncol=2)
lndMvst(x=c(rep(0,2)),nu=4,mu=c(rep(0,2)),rooti=backsolve(chol(Sigma),diag(2)))
```

logMargDenNR

*Compute Log Marginal Density Using Newton-Raftery Approx***Description**

logMargDenNR computes log marginal density using the Newton-Raftery approximation.
 Note: this approximation can be influenced by outliers in the vector of log-likelihoods. Use with **care**.

Usage

```
logMargDenNR( ll )
```

Arguments

ll vector of log-likelihoods evaluated at length(ll) MCMC draws

Value

approximation to log marginal density value.

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 6.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

margarine

*Household Panel Data on Margarine Purchases***Description**

Panel data on purchases of margarine by 516 households. Demographic variables are included.

Usage

```
data(margarine)
```

Format

This is an R object that is a list of two data frames, `list(choicePrice,demos)`

List of 2

\$ choicePrice: 'data.frame': 4470 obs. of 12 variables:

```
...$ hhid : int [1:4470] 2100016 2100016 2100016 2100016
...$ choice : num [1:4470] 1 1 1 1 1 4 1 1 4 1
...$ PPK_Stk : num [1:4470] 0.66 0.63 0.29 0.62 0.5 0.58 0.29
...$ PBB_Stk : num [1:4470] 0.67 0.67 0.5 0.61 0.58 0.45 0.51
...$ PFI_Stk : num [1:4470] 1.09 0.99 0.99 0.99 0.99 0.99 0.99
...$ PHse_Stk: num [1:4470] 0.57 0.57 0.57 0.57 0.45 0.45 0.29
...$ PGen_Stk: num [1:4470] 0.36 0.36 0.36 0.36 0.33 0.33 0.33
...$ PImp_Stk: num [1:4470] 0.93 1.03 0.69 0.75 0.72 0.72 0.72
...$ PSS_Tub : num [1:4470] 0.85 0.85 0.79 0.85 0.85 0.85 0.85
...$ PPK_Tub : num [1:4470] 1.09 1.09 1.09 1.09 1.07 1.07 1.07
...$ PFI_Tub : num [1:4470] 1.19 1.19 1.19 1.19 1.19 1.19 1.19
...$ PHse_Tub: num [1:4470] 0.33 0.37 0.59 0.59 0.59 0.59 0.59
```

Pk is Parkay; BB is BlueBonnett, FI is Fleischmanns, Hse is house, Gen is generic, Imp is Imperial, SS is Shed Spread. _Stk indicates stick, _Tub indicates Tub form.

\$ demos : 'data.frame': 516 obs. of 8 variables:

```
...$ hhid : num [1:516] 2100016 2100024 2100495 2100560
...$ Income : num [1:516] 32.5 17.5 37.5 17.5 87.5 12.5
...$ Fs3_4 : int [1:516] 0 1 0 0 0 0 0 0 0
...$ Fs5 : int [1:516] 0 0 0 0 0 0 0 0 1
...$ Fam_Size : int [1:516] 2 3 2 1 1 2 2 2 5
...$ college : int [1:516] 1 1 0 0 1 0 1 0 1
...$ whtcollar: int [1:516] 0 1 0 1 1 0 0 0 1
...$ retired : int [1:516] 1 1 1 0 0 1 0 1 0
```

Fs3_4 is dummy (family size 3-4). Fs5 is dummy for family size ≥ 5 . college, whtcollar, retired are dummies reflecting these statuses.

Details

choice is a multinomial indicator of one of the 10 brands (in order listed under format). All prices are in \$.

Source

Allenby and Rossi (1991), "Quality Perceptions and Asymmetric Switching Between Brands," *Marketing Science* 10, 185-205.

References

Chapter 5, *Bayesian Statistics and Marketing* by Rossi et al.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```
data(margarine)
cat(" Table of Choice Variable ",fill=TRUE)
print(table(margarine$choicePrice[,2]))
cat(" Means of Prices",fill=TRUE)
mat=apply(as.matrix(margarine$choicePrice[,3:12]),2,mean)
print(mat)
cat(" Quantiles of Demographic Variables",fill=TRUE)
mat=apply(as.matrix(margarine$demos[,2:8]),2,quantile)
print(mat)

##
## example of processing for use with rhierMnlRwMixture
##
if(nchar(Sys.getenv("LONG_TEST")) != 0)
{
  select= c(1:5,7) ## select brands
  chPr=as.matrix(margarine$choicePrice)
  ## make sure to log prices
  chPr=cbind(chPr[,1],chPr[,2],log(chPr[,2+select]))
  demos=as.matrix(margarine$demos[,c(1,2,5)])

  ## remove obs for other alts
  chPr=chPr[chPr[,2] <= 7,]
  chPr=chPr[chPr[,2] != 6,]

  ## recode choice
  chPr[chPr[,2] == 7,2]=6

  hhidl=levels(as.factor(chPr[,1]))
  lgtdata=NULL
  nlgt=length(hhidl)
  p=length(select) ## number of choice alts
  ind=1
  for (i in 1:nlgt) {
    nobs=sum(chPr[,1]==hhidl[i])
    if(nobs >=5) {
      data=chPr[chPr[,1]==hhidl[i],]
      y=data[,2]
      names(y)=NULL
      X=createX(p=p,na=1,Xa=data[,3:8],nd=NULL,Xd=NULL,INT=TRUE,base=1)
      lgtdata[[ind]]=list(y=y,X=X,hhid=hhidl[i]); ind=ind+1
    }
  }
  nlgt=length(lgtdata)
  ##
```

```

## now extract demos corresponding to hhs in lgtdata
##
Z=NULL
nlgt=length(lgtdata)
for(i in 1:nlgt){
  Z=rbind(Z,demos[demos[,1]==lgtdata[[i]]$hhid,2:3])
}
##
## take log of income and family size and demean
##
Z=log(Z)
Z[,1]=Z[,1]-mean(Z[,1])
Z[,2]=Z[,2]-mean(Z[,2])

keep=5
R=20000
mcmc1=list(keep=keep,R=R)
out=rhierMnlRwMixture(Data=list(p=p,lgtdata=lgtdata,Z=Z),Prior=list(ncomp=1),Mcmc=mcmc1)

begin=100/keep; end=R/keep
cat(" Posterior Mean of Delta ",fill=TRUE)
mat=apply(out$Deltadraw[begin:end,,2],2,mean)
print(matrix(mat,ncol=6))
pmom=momMix(out$probdraw[begin:end,,],out$compdraw[begin:end])
cat(" posterior expectation of mu",fill=TRUE)
print(pmom$mu)
cat(" posterior expectation of sd",fill=TRUE)
print(pmom$sd)
cat(" posterior expectation of correlations",fill=TRUE)
print(pmom$corr)

}

```

mixDen

Compute Marginal Density for Multivariate Normal Mixture

Description

mixDen computes the marginal density for each component of a normal mixture at each of the points on a user-specified grid.

Usage

```
mixDen(x, pvec, comps)
```

Arguments

x	array - ith column gives grid points for ith variable
pvec	vector of mixture component probabilities
comps	list of lists of components for normal mixture

Details

length(comps) is the number of mixture components. comps[[j]] is a list of parameters of the jth component. comps[[j]]\$mu is mean vector; comps[[j]]\$rooti is the UL decomp of Σ^{-1} .

Value

an array of the same dimension as grid with density values.

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago (Peter.Rossi@ChicagoGsb.edu).

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 3.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rnmixGibbs](#)

Examples

```
## Not run:
##
## see examples in rnmixGibbs documentation
##
## End(Not run)
```

mixDenBi

Compute Bivariate Marginal Density for a Normal Mixture

Description

mixDenBi computes the implied bivariate marginal density from a mixture of normals with specified mixture probabilities and component parameters.

Usage

```
mixDenBi(i, j, xi, xj, pvec, comps)
```


Arguments

i	index of first variable
j	index of second variable
xi	grid of values of first variable
xj	grid of values of second variable
pvec	normal mixture probabilities
comps	list of lists of components

Details

length(comps) is the number of mixture components. comps[[j]] is a list of parameters of the jth component. comps[[j]]\$mu is mean vector; comps[[j]]\$rooti is the UL decomp of Σ^{-1} .

Value

an array (length(xi)=length(xj) x 2) with density value

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago (Peter.Rossi@ChicagoGsb.edu).

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 3.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rnmixGibbs](#), [mixDen](#)

Examples

```
## Not run:
##
## see examples in rnmixGibbs documentation
##
## End(Not run)
```

mnlHess

Computes -Expected Hessian for Multinomial Logit

Description

mnlHess computes -Expected[Hessian] for Multinomial Logit Model

Usage

```
mnlHess(beta, y, X)
```

Arguments

beta	k x 1 vector of coefficients
y	n x 1 vector of choices, (1, ..., p)
X	n*p x k Design matrix

Details

See [llmnl](#) for information on structure of X array. Use [createX](#) to make X.

Value

k x k matrix

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 3.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[llmnl](#), [createX](#), [rmnlIndepMetrop](#)

Examples

```
##
## Not run: mnlHess(beta, y, X)
```

mnpProb*Compute MNP Probabilities*

Description

mnpProb computes MNP probabilities for a given X matrix corresponding to one observation. This function can be used with output from rmnpGibbs to simulate the posterior distribution of market shares or fitted probabilities.

Usage

```
mnpProb(beta, Sigma, X, r)
```

Arguments

beta	MNP coefficients
Sigma	Covariance matrix of latents
X	X array for one observation – use createX to make
r	number of draws used in GHK (def: 100)

Details

see [rmnpGibbs](#) for definition of the model and the interpretation of the beta, Sigma parameters. Uses the GHK method to compute choice probabilities. To simulate a distribution of probabilities, loop over the beta, Sigma draws from rmnpGibbs output.

Value

p x 1 vector of choice probabilities

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapters 2 and 4.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rmnpGibbs](#), [createX](#)

Examples

```
##
## example of computing MNP probabilities
## here I'm thinking of Xa as having the prices of each of the 3 alternatives
Xa=matrix(c(1,.5,1.5),nrow=1)
X=createX(p=3,na=1,nd=NULL,Xa=Xa,Xd=NULL,DIFF=TRUE)
beta=c(1,-1,-2) ## beta contains two intercepts and the price coefficient
Sigma=matrix(c(1,.5,.5,1),ncol=2)
mnpProb(beta,Sigma,X)
```

momMix

Compute Posterior Expectation of Normal Mixture Model Moments

Description

momMix averages the moments of a normal mixture model over MCMC draws.

Usage

```
momMix(probdraw, compdraw)
```

Arguments

probdraw	R x ncomp list of draws of mixture probs
compdraw	list of length R of draws of mixture component moments

Details

R is the number of MCMC draws in argument list above.
 ncomp is the number of mixture components fitted.
 compdraw is a list of lists of lists with mixture components.
 compdraw[[i]] is ith draw.
 compdraw[[i]][[j]][[1]] is the mean parameter vector for the jth component, ith MCMC draw.
 compdraw[[i]][[j]][[2]] is the UL decomposition of Σ^{-1} for the jth component, ith MCMC draw.

Value

a list of the following items ...

mu	Posterior Expectation of Mean
sigma	Posterior Expectation of Covariance Matrix
sd	Posterior Expectation of Vector of Standard Deviations
corr	Posterior Expectation of Correlation Matrix

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 5.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rmixGibbs](#)

nmat	<i>Convert Covariance Matrix to a Correlation Matrix</i>
------	--

Description

nmat converts a covariance matrix (stored as a vector, col by col) to a correlation matrix (also stored as a vector).

Usage

```
nmat(vec)
```

Arguments

vec k x k Cov matrix stored as a k*k x 1 vector (col by col)

Details

This routine is often used with apply to convert an R x (k*k) array of covariance MCMC draws to correlations. As in `corrdraws=apply(vardraws,1,nmat)`

Value

k*k x 1 vector with correlation matrix

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

Examples

```
##
set.seed(66)
X=matrix(rnorm(200,4),ncol=2)
Varmat=var(X)
nmat(as.vector(Varmat))
```

numEff	<i>Compute Numerical Standard Error and Relative Numerical Efficiency</i>
--------	---

Description

numEff computes the numerical standard error for the mean of a vector of draws as well as the relative numerical efficiency (ratio of variance of mean of this time series process relative to iid sequence).

Usage

```
numEff(x, m = as.integer(min(length(x), (100/sqrt(5000)) * sqrt(length(x)))))
```

Arguments

x	R x 1 vector of draws
m	number of lags for autocorrelations

Details

default for number of lags is chosen so that if $R = 5000$, $m = 100$ and increases as the \sqrt{R} .

Value

stderr	standard error of the mean of x
f	variance ratio (relative numerical efficiency)

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 3.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```
numEff(rnorm(1000), m=20)
numEff(rnorm(1000))
```

`rbiNormGibbs`*Illustrate Bivariate Normal Gibbs Sampler*

Description

`rbiNormGibbs` implements a Gibbs Sampler for the bivariate normal distribution. Intermediate moves are shown and the output is contrasted with the iid sampler. ⁱ This function is designed for illustrative/teaching purposes.

Usage

```
rbiNormGibbs(initx = 2, inity = -2, rho, burnin = 100, R = 500)
```

Arguments

<code>initx</code>	initial value of parameter on x axis (def: 2)
<code>inity</code>	initial value of parameter on y axis (def: -2)
<code>rho</code>	correlation for bivariate normals
<code>burnin</code>	burn-in number of draws (def:100)
<code>R</code>	number of MCMC draws (def:500)

Details

(theta1,theta2) $N((0,0), \text{Sigma}=\text{matrix}(c(1,\text{rho},\text{rho},1),\text{ncol}=2))$

Value

R x 2 array of draws

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapters 2 and 3.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```
##  
## Not run: out=rbiNormGibbs(rho=.95)
```

rbprobitGibbs

Gibbs Sampler (Albert and Chib) for Binary Probit

Description

rbprobitGibbs implements the Albert and Chib Gibbs Sampler for the binary probit model.

Usage

```
rbprobitGibbs(Data, Prior, Mcmc)
```

Arguments

Data	list(X,y)
Prior	list(betabar,A)
Mcmc	list(R,keep)

Details

Model: $z = X\beta + e$. $e \sim N(0, I)$. $y=1$, if $z > 0$.

Prior: $\beta \sim N(\text{betabar}, A^{-1})$.

List arguments contain

X Design Matrix

y n x 1 vector of observations, (0 or 1)

betabar k x 1 prior mean (def: 0)

A k x k prior precision matrix (def: .01I)

R number of MCMC draws

keep thinning parameter - keep every keepth draw (def: 1)

Value

betadraw R/keep x k array of betadraws

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 3.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rmnpGibbs](#)

Examples

```
##
## rbprobitGibbs example
##
if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=2000} else {R=10}

set.seed(66)
simbprobit=
function(X,beta) {
  ## function to simulate from binary probit including x variable
  y=ifelse((X*%beta+rnorm(nrow(X)))<0,0,1)
  list(X=X,y=y,beta=beta)
}

nobs=200
X=cbind(rep(1,nobs),runif(nobs),runif(nobs))
beta=c(0,1,-1)
nvar=ncol(X)
simout=simbprobit(X,beta)

Data=list(X=simout$X,y=simout$y)
Mcmc=list(R=R,keep=1)

out=rbprobitGibbs(Data=Data,Mcmc=Mcmc)

cat(" Betadraws ",fill=TRUE)
mat=apply(out$betadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(beta,mat); rownames(mat)[1]="beta"; print(mat)
```

rdirichlet

*Draw From Dirichlet Distribution***Description**

rdirichlet draws from Dirichlet

Usage

```
rdirichlet(alpha)
```

Arguments

alpha vector of Dirichlet parms (must be > 0)

Value

Vector of draws from Dirichlet

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 2.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```
##
set.seed(66)
rdirichlet(c(rep(3,5)))
```

rhierBinLogit

MCMC Algorithm for Hierarchical Binary Logit

Description

rhierBinLogit implements an MCMC algorithm for hierarchical binary logits with a normal heterogeneity distribution. This is a hybrid sampler with a RW Metropolis step for unit-level logit parameters.

rhierBinLogit is designed for use on choice-based conjoint data with partial profiles. The Design matrix is based on differences of characteristics between two alternatives. See Appendix A of *Bayesian Statistics and Marketing* for details.

Usage

```
rhierBinLogit(Data, Prior, Mcmc)
```

Arguments

Data	list(lgtdata,Z) (note: Z is optional)
Prior	list(Deltabar,ADelta,nu,V) (note: all are optional)
Mcmc	list(sbeta,R,keep) (note: all but R are optional)

Details

Model:

$y_{hi} = 1$ with $pr = \exp(x'_{hi} \beta_{hi}) / (1 + \exp(x'_{hi} \beta_{hi}))$. β_{hi} is $nvar \times 1$.
 $h=1, \dots, \text{length}(\text{lgtdata})$ units or "respondents" for survey data.

$\beta_{hi} = Z\Delta[h,] + u_h$.

Note: here ZDelta refers to $Z \% \Delta$, ZDelta[h,] is hth row of this product.

Delta is an $nz \times nvar$ array.

$u_h \sim N(0, V_{\beta_{hi}})$.

Priors:

$\text{delta} = \text{vec}(\Delta) \sim N(\text{vec}(\text{Deltabar}), V_{\beta_{hi}}(x) A\Delta^{-1})$

$V_{\beta_{hi}} \sim IW(nu, V)$

Lists contain:

lgtdata list of lists with each cross-section unit MNL data
 lgtdata[[h]]\$y n_h vector of binary outcomes (0,1)
 lgtdata[[h]]\$X n_h by nvar design matrix for hth unit
 Deltabar nz x nvar matrix of prior means (def: 0)
 ADelta prior prec matrix (def: .01I)
 nu d.f. parm for IW prior on norm comp Sigma (def: nvar+3)
 V pds location parm for IW prior on norm comp Sigma (def: nuI)
 sbeta scaling parm for RW Metropolis (def: .2)
 R number of MCMC draws
 keep MCMC thinning parm: keep every keepth draw (def: 1)

Value

a list containing:

Deltadraw	R/keep x nz*nvar matrix of draws of Delta
betadraw	nlgt x nvar x R/keep array of draws of betas
Vbetadraw	R/keep x nvar*nvar matrix of draws of Vbeta
llike	R/keep vector of log-like values
reject	R/keep vector of reject rates over nlgt units

Note

Some experimentation with the Metropolis scaling paramter (sbeta) may be required.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 5.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rhierMnlRwMixture](#)

Examples

```
##
if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=10000} else {R=10}

set.seed(66)
nvar=5                                ## number of coefficients
nlgt=1000                             ## number of cross-sectional units
nobs=10                               ## number of observations per unit
nz=2                                  ## number of regressors in mixing distribution

## set hyper-parameters
```

```
##      B=ZDelta + U

Z=matrix(c(rep(1,nlgt),runif(nlgt,min=-1,max=1)),nrow=nlgt,ncol=nz)
Delta=matrix(c(-2,-1,0,1,2,-1,1,-.5,.5,0),nrow=nz,ncol=nvar)
iota=matrix(1,nrow=nvar,ncol=1)
Vbeta=diag(nvar)+.5*iota%*%t(iota)

## simulate data
lgtdata=NULL

for (i in 1:nlgt)
{ beta=t(Delta)%*%Z[i,]+as.vector(t(chol(Vbeta))%*%rnorm(nvar))
  X=matrix(runif(nobs*nvar),nrow=nobs,ncol=nvar)
  prob=exp(X%*%beta)/(1+exp(X%*%beta))
  unif=runif(nobs,0,1)
  y=ifelse(unif<prob,1,0)
  lgtdata[[i]]=list(y=y,X=X,beta=beta)
}

Data=list(Dat=lgtdata,Demo=Z)
out=rhierBinLogit(Data=list(lgtdata=lgtdata,Z=Z),Mcmc=list(R=R))

cat(" Deltadraws ",fill=TRUE)
mat=apply(out$Deltadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(Delta),mat); rownames(mat)[1]="delta"; print(mat)
cat(" Vbetadraws ",fill=TRUE)
mat=apply(out$Vbetadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(Vbeta),mat); rownames(mat)[1]="Vbeta"; print(mat)

if(0){
td=as.vector(Delta)
par(mfrow=c(2,2))
matplot(out$Deltadraw[, (1:nvar)],type="l")
abline(h=td[1:nvar],col=(1:nvar))
matplot(out$Deltadraw[, ((nvar+1):(2*nvar))],type="l")
abline(h=td[(nvar+1):(2*nvar)],col=(1:nvar))
matplot(out$Vbetadraw[,c(1,7,13,19,25)],type="l")
abline(h=1.5)
matplot(out$Vbetadraw[, -c(1,7,13,19,25)],type="l")
abline(h=.5)
}
```

rhierLinearModel *Gibbs Sampler for Hierarchical Linear Model*

Description

rhierLinearModel implements a Gibbs Sampler for hierarchical linear models.

Usage

```
rhierLinearModel(Data, Prior, Mcmc)
```

Arguments

Data	list(regdata,Z) (Z optional).
Prior	list(Deltabar,A,nu.e,ssq,nu,V) (optional).
Mcmc	list(R,keep) (R required).

Details

Model: length(regdata) regression equations.
 $y_i = X_i \beta_i + e_i$. $e_i \sim N(0, \tau_{\nu_i})$. nvar X vars in each equation.

Priors:

$\tau_{\nu_i} \sim \text{nu.e} * \text{ssq}_i / \chi^2_{\text{nu.e}}$. τ_{ν_i} is the variance of e_i .

$\beta_i \sim N(\text{ZDelta}[i,], V_{\beta_i})$.

Note: ZDelta is the matrix $Z * \Delta$; [i,] refers to ith row of this product.

$\text{vec}(\Delta)$ given $V_{\beta_i} \sim N(\text{vec}(\text{Deltabar}), V_{\beta_i}(x)A^{-1})$.

$V_{\beta_i} \sim \text{IW}(\text{nu}, V)$.

Δ , Deltabar are $\text{nz} \times \text{nvar}$. A is $\text{nz} \times \text{nz}$. V_{β_i} is $\text{nvar} \times \text{nvar}$.

Note: if you don't have any z vars, set $Z=\text{iota}$ (nreg x 1).

List arguments contain:

regdata list of lists with X,y matrices for each of length(regdata) regressions
 regdata[[i]]\$X X matrix for equation i
 regdata[[i]]\$y y vector for equation i
 Deltabar $\text{nz} \times \text{nvar}$ matrix of prior means (def: 0)
 A $\text{nz} \times \text{nz}$ matrix for prior precision (def: .01I)
 nu.e d.f. parm for regression error variance prior (def: 3)
 ssq scale parm for regression error var prior (def: $\text{var}(y_i)$)
 nu d.f. parm for Vbeta prior (def: $\text{nvar}+3$)
 V Scale location matrix for Vbeta prior (def: $\text{nu} * \text{I}$)
 R number of MCMC draws
 keep MCMC thinning parm: keep every keepth draw (def: 1)

Value

a list containing

betadraw	nreg x nvar x R/keep array of individual regression coef draws
taudraw	R/keep x nreg array of error variance draws
Deltadraw	R/keep x $\text{nz} \times \text{nvar}$ array of Deltadraws
Vbetadraw	R/keep x $\text{nvar} * \text{nvar}$ array of Vbeta draws

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 3.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```
##
if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=2000} else {R=10}

nreg=100; nobs=100; nvar=3
Vbeta=matrix(c(1,.5,0,.5,2,.7,0,.7,1),ncol=3)
Z=cbind(c(rep(1,nreg)),3*runif(nreg)); Z[,2]=Z[,2]-mean(Z[,2])
nz=ncol(Z)
Delta=matrix(c(1,-1,2,0,1,0),ncol=2)
Delta=t(Delta) # first row of Delta is means of betas
Beta=matrix(rnorm(nreg*nvar),nrow=nreg)%*%chol(Vbeta)+Z%*%Delta
tau=.1
iota=c(rep(1,nobs))
regdata=NULL
for (reg in 1:nreg) { X=cbind(iota,matrix(runif(nobs*(nvar-1)),ncol=(nvar-1)))
  y=X%*%Beta[reg,]+sqrt(tau)*rnorm(nobs); regdata[[reg]]=list(y=y,X=X) }

nu.e=3
ssq=NULL
for(reg in 1:nreg) {ssq[reg]=var(regdata[[reg]]$y)}
nu=nvar+3
V=nu*diag(c(rep(1,nvar)))
A=diag(c(rep(.01,nz)),ncol=nz)
Deltabar=matrix(c(rep(0,nz*nvar)),nrow=nz)

Data=list(regdata=regdata,Z=Z)
Prior=list(Deltabar=Deltabar,A=A,nu.e=nu.e,ssq=ssq,nu=nu,V=V)
Mcmc=list(R=R,keep=1)
out=rhierLinearModel(Data=Data,Mcmc=Mcmc)

cat(" Deltadraws ",fill=TRUE)
mat=apply(out$Deltadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(Delta),mat); rownames(mat)[1]="delta"; print(mat)
cat(" Vbetadraws ",fill=TRUE)
mat=apply(out$Vbetadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(Vbeta),mat); rownames(mat)[1]="Vbeta"; print(mat)
```

rhierMnlRwMixture	<i>MCMC Algorithm for Hierarchical Multinomial Logit with Mixture of Normals Heterogeneity</i>
-------------------	--

Description

rhierMnlRwMixture is a MCMC algorithm for a hierarchical multinomial logit with a mixture of normals heterogeneity distribution. This is a hybrid Gibbs Sampler with a RW Metropolis step for the MNL coefficients for each panel unit.

Usage

```
rhierMnlRwMixture(Data, Prior, Mcmc)
```

Arguments

Data	list(p,lgtdata,Z) (Z is optional)
Prior	list(deltabar,Ad,mubar,Amu,nu,V,ncomp) (all but ncomp are optional)
Mcmc	list(s,w,R,keep) (R required)

Details**Model:**

$y_i \sim MNL(X_i, \theta_i)$. $i=1, \dots, \text{length}(\text{lgtdata})$. θ_i is $\text{nvar} \times 1$.

$\theta_i = \text{ZDelta}[i,] + u_i$.

Note: here ZDelta refers to $Z \% \% D$, ZDelta[i,] is ith row of this product.

Delta is an $\text{nz} \times \text{nvar}$ array.

$u_i \sim N(\mu_{ind}, \Sigma_{ind})$. $ind \sim \text{multinomial}(\text{pvec})$.

Priors:

$\text{pvec} \sim \text{dirichlet}(\mathbf{a})$

$\text{delta} = \text{vec}(\text{Delta}) \sim N(\text{deltabar}, A_d^{-1})$

$\mu_j \sim N(\text{mubar}, \Sigma_j(x) A_{\mu}^{-1})$

$\Sigma_j \sim \text{IW}(\text{nu}, V)$

Lists contain:

p p is number of choice alternatives

lgtdata list of lists with each cross-section unit MNL data

lgtdata[[i]]\$y n_i vector of multinomial outcomes (1,...,m)

lgtdata[[i]]\$X n_i by nvar design matrix for ith unit

deltabar $\text{nz} \times \text{nvar}$ vector of prior means (def: 0)

Ad prior prec matrix for vec(D) (def: .01I)

mubar $\text{nvar} \times 1$ prior mean vector for normal comp mean (def: 0)

Amu prior precision for normal comp mean (def: .01I)

nu d.f. parm for IW prior on norm comp Sigma (def: $\text{nvar}+3$)

V pds location parm for IW prior on norm comp Sigma (def: nuI)

ncomp number of components used in normal mixture

s scaling parm for RW Metropolis (def: $2.93/\sqrt{\text{nvar}}$)

w fractional likelihood weighting parm (def: .1)

R number of MCMC draws

keep MCMC thinning parm: keep every keepth draw (def: 1)

Value

a list containing:

Deltadraw R/keep x $\text{nz} \times \text{nvar}$ matrix of draws of Delta, first row is initial value

betadraw $\text{nlgt} \times \text{nvar} \times \text{R/keep}$ array of draws of betas

probdraw R/keep x ncomp matrix of draws of probs of mixture components (pvec)

compdraw list of list of lists (length R/keep)

loglike log-likelihood for each kept draw (length R/keep)

Note

More on `compdraw` component of return value list:

`compdraw[[i]]` the *i*th draw of components for mixtures
`compdraw[[i][[j]]]` *i*th draw of the *j*th normal mixture comp
`compdraw[[i][[j]][[1]]]` *i*th draw of *j*th normal mixture comp mean vector
`compdraw[[i][[j]][[2]]]` *i*th draw of *j*th normal mixture cov parm (rooti)

Note: *Z* does **not** include an intercept and is centered for ease of interpretation.

Be careful in assessing prior parameter, *Amu*. .01 is too small for many applications. See Rossi et al, chapter 5 for full discussion.

Note: as of version 2.0-2 of *bayesm*, the fractional weight parameter has been changed to a weight between 0 and 1. *w* is the fractional weight on the normalized pooled likelihood. This differs from what is in Rossi et al chapter 5, i.e.

$$like_i(1 - w)xlike_{pooled}(n_i/N) * w$$

Large *R* values may be required (>20,000).

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 5.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rmnlIndepMetrop](#)

Examples

```
##
if(nchar(Sys.getenv("LONG_TEST")) != 0)
{

  set.seed(66)
  p=3                                # num of choice alterns
  ncoef=3
  nlgt=300                           # num of cross sectional units
  nz=2
  Z=matrix(runif(nz*nlgt),ncol=nz)
  Z=t(t(Z)-apply(Z,2,mean))          # demean Z
  ncomp=3                             # no of mixture components
  Delta=matrix(c(1,0,1,0,1,2),ncol=2)
  comps=NULL
  comps[[1]]=list(mu=c(0,-1,-2),rooti=diag(rep(1,3)))
  comps[[2]]=list(mu=c(0,-1,-2)*2,rooti=diag(rep(1,3)))
  comps[[3]]=list(mu=c(0,-1,-2)*4,rooti=diag(rep(1,3)))
```



```

pvec=c(.4,.2,.4)

## simulate data
simlgtdata=NULL
ni=rep(50,300)
for (i in 1:nlgt)
{
  betai=Delta%*%Z[i,]+as.vector(rmixture(1,pvec,comps)$x)
  X=NULL
  for(j in 1:ni[i])
  {
    Xone=cbind(rbind(c(rep(0,p-1)),diag(p-1)),runif(p,min=-1.5,max=0))
    X=rbind(X,Xone)
  }
  outa=simmlwX(ni[i],X,betai)
  simlgtdata[[i]]=list(y=outa$y,X=X,beta=betai)
}

## plot betas
if(0){
  ## set if(1) above to produce plots
  bmat=matrix(0,nlgt,ncoef)
  for(i in 1:nlgt) {bmat[i,]=simlgtdata[[i]]$beta}
  par(mfrow=c(ncoef,1))
  for(i in 1:ncoef) hist(bmat[,i],breaks=30,col="magenta")
}

## set parms for priors and Z
nu=ncoef+3
V=nu*diag(rep(1,ncoef))
Ad=.01*(diag(rep(1,nz*ncoef)))
mubar=matrix(rep(0,ncoef),nrow=1)
deltabar=rep(0,ncoef*nz)
Amu=matrix(.01,ncol=1)
a=rep(5,ncoef)

R=10000
keep=5
w=.1
s=2.93/sqrt(ncoef)
Data1=list(p=p,lgtdata=simlgtdata,Z=Z)
Prior1=list(ncomp=ncomp,nu=nu,V=V,Amu=Amu,mubar=mubar,a=a,Ad=Ad,deltabar=deltabar)
Mcmc1=list(s=s,w=w,R=R,keep=keep)
out=rhierMnlRwMixture(Data=Data1,Prior=Prior1,Mcmc=Mcmc1)

if(R < 1000) {begin=1} else {begin=1000/keep}
end=R/keep
cat(" Deltadraws ",fill=TRUE)
mat=apply(out$Deltadraw[begin:end,],2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(Delta),mat); rownames(mat)[1]="delta"; print(mat)

tmom=momMix(matrix(pvec,nrow=1),list(comps))
pmom=momMix(out$probdraw[begin:end,],out$compdraw[begin:end])
mat=rbind(tmom$mu,pmom$mu)
rownames(mat)=c("true","post expect")
cat(" mu and posterior expectation of mu",fill=TRUE)
print(mat)
mat=rbind(tmom$sd,pmom$sd)
rownames(mat)=c("true","post expect")
cat(" std dev and posterior expectation of sd",fill=TRUE)

```

```

print(mat)
mat=rbind(as.vector(tmom$corr),as.vector(pmom$corr))
rownames(mat)=c("true","post expect")
cat(" corr and posterior expectation of corr",fill=TRUE)
print(t(mat))

if(0) {
## set if(1) to produce plots
par(mfrow=c(4,1))
plot(out$betadraw[1,1,])
abline(h=simlgtdata[[1]]$beta[1])
plot(out$betadraw[2,1,])
abline(h=simlgtdata[[2]]$beta[1])
plot(out$betadraw[100,3,])
abline(h=simlgtdata[[100]]$beta[3])
plot(out$betadraw[101,3,])
abline(h=simlgtdata[[101]]$beta[3])
par(mfrow=c(4,1))
plot(out$Deltadraw[,1])
abline(h=Delta[1,1])
plot(out$Deltadraw[,2])
abline(h=Delta[2,1])
plot(out$Deltadraw[,3])
abline(h=Delta[3,1])
plot(out$Deltadraw[,6])
abline(h=Delta[3,2])
begin=1000/keep
end=R/keep
ngrid=50
grid=matrix(0,ngrid,ncoef)
rgm=matrix(c(-3,-7,-10,3,1,0),ncol=2)
for(i in 1:ncoef) {rg=rgm[i,]; grid[,i]=rg[1] + ((1:ngrid)/ngrid)*(rg[2]-rg[1])}
mden=eMixMargDen(grid,out$probdraw[begin:end,],out$compdraw[begin:end])
par(mfrow=c(2,ncoef))
for(i in 1:ncoef)
{plot(grid[,i],mden[,i],type="l")}
for(i in 1:ncoef)
tden=mixDen(grid,pvec,comps)
for(i in 1:ncoef)
{plot(grid[,i],tden[,i],type="l")}
}
}

```

Description

rhierNegbinRw implements an MCMC strategy for the hierarchical Negative Binomial (NBD) regression model. Metropolis steps for each unit level set of regression parameters are automatically tuned by optimization. Over-dispersion parameter (alpha) is common across units.

Usage

```
rhierNegbinRw(Data, Prior, Mcmc)
```

Arguments

Data	list(regdata,Z)
Prior	list(Deltabar,Adelta,nu,V,a,b)
Mcmc	list(R,keep,s_beta,s_alpha,c,Vbeta0,Delta0)

Details

Model: $y_i \sim \text{NBD}(\text{mean}=\lambda, \text{over-dispersion}=\alpha)$.

$\lambda = \exp(X_i \beta_i)$

Prior: $\beta_i \sim N(\text{Delta}'z_i, \text{Vbeta})$.

$\text{vec}(\text{Delta}|\text{Vbeta}) \sim N(\text{vec}(\text{Deltabar}), \text{Vbeta}(x)\text{Adelta})$.

$\text{Vbeta} \sim \text{IW}(\text{nu}, \text{V})$.

$\alpha \sim \text{Gamma}(a, b)$.

note: prior mean of $\alpha = a/b$, variance = $a/(b^2)$

list arguments contain:

regdata list of lists with data on each of nreg units

regdata[[i]]\$X nobs_i x nvar matrix of X variables

regdata[[i]]\$y nobs_i x 1 vector of count responses

Z nreg x nz mat of unit chars (def: vector of ones)

Deltabar nz x nvar prior mean matrix (def: 0)

Adelta nz x nz pds prior prec matrix (def: .01I)

nu d.f. parm for IWishart (def: nvar+3)

V location matrix of IWishart prior (def: nuI)

a Gamma prior parm (def: .5)

b Gamma prior parm (def: .1)

R number of MCMC draws

keep MCMC thinning parm: keep every keepth draw (def: 1)

s_beta scaling for beta| alpha RW inc cov (def: 2.93/sqrt(nvar))

s_alpha scaling for alpha | beta RW inc cov (def: 2.93)

c fractional likelihood weighting parm (def:2)

Vbeta0 starting value for Vbeta (def: I)

Delta0 starting value for Delta (def: 0)

Value

a list containing:

llike R/keep vector of values of log-likelihood

betadraw nreg x nvar x R/keep array of beta draws

alphadraw R/keep vector of alpha draws

acceptrbeta acceptance rate of the beta draws

acceptralpha acceptance rate of the alpha draws

Note

The NBD regression encompasses Poisson regression in the sense that as α goes to infinity the NBD distribution tends to the Poisson.

For "small" values of α , the dependent variable can be extremely variable so that a large number of observations may be required to obtain precise inferences.

For ease of interpretation, we recommend demeaning Z variables.

Author(s)

Sridhar Narayanam & Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 5.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rnegbinRw](#)

Examples

```
##
if(nchar(Sys.getenv("LONG_TEST")) != 0)
{
  ##
  set.seed(66)
  simnegbin =
  function(X, beta, alpha) {
    # Simulate from the Negative Binomial Regression
    lambda = exp(X %*% beta)
    y=NULL
    for (j in 1:length(lambda))
      y = c(y,rnbinom(1,mu = lambda[j],size = alpha))
    return(y)
  }

  nreg = 100          # Number of cross sectional units
  T = 50              # Number of observations per unit
  nobs = nreg*T
  nvar=2              # Number of X variables
  nz=2                # Number of Z variables

  # Construct the Z matrix
  Z = cbind(rep(1,nreg),rnorm(nreg,mean=1,sd=0.125))

  Delta = cbind(c(0.4,0.2), c(0.1,0.05))
  alpha = 5
  Vbeta = rbind(c(0.1,0),c(0,0.1))

  # Construct the regdata (containing X)
  simnegbindata = NULL
  for (i in 1:nreg) {
    betai = as.vector(Z[i,]%*%Delta) + chol(Vbeta)%*%rnorm(nvar)
```

```

X = cbind(rep(1,T),rnorm(T,mean=2,sd=0.25))
simnegbindata[[i]] = list(y=simnegbin(X,betai,alpha), X=X,beta=betai)
}

Beta = NULL
for (i in 1:nreg) {Beta=rbind(Beta,matrix(simnegbindata[[i]]$beta,nrow=1))}

Data = list(regdata=simnegbindata, Z=Z)
Deltabar = matrix(rep(0,nvar*nz),nrow=nz)
Vdelta = 0.01 * diag(nvar)
nu = nvar+3
V = 0.01*diag(nvar)
a = 0.5
b = 0.1
Prior = list(Deltabar=Deltabar, Vdelta=Vdelta, nu=nu, V=V, a=a, b=b)

R=10000
keep =1
s_beta=2.93/sqrt(nvar)
s_alpha=2.93
c=2
Mcmc = list(R=R, keep = keep, s_beta=s_beta, s_alpha=s_alpha, c=c)
out = rhierNegbinRw(Data, Prior, Mcmc)

# Unit level mean beta parameters
Mbeta = matrix(rep(0,nreg*nvar),nrow=nreg)
ndraws = length(out$alphadraw)
for (i in 1:nreg) { Mbeta[i,] = rowSums(out$Betadraw[i, , ])/ndraws }

cat(" Deltadraws ",fill=TRUE)
mat=apply(out$Deltadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(Delta),mat); rownames(mat)[1]="Delta"; print(mat)
cat(" Vbetadraws ",fill=TRUE)
mat=apply(out$Vbetadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(Vbeta),mat); rownames(mat)[1]="Vbeta"; print(mat)
cat(" alphadraws ",fill=TRUE)
mat=apply(matrix(out$alphadraw),2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(alpha),mat); rownames(mat)[1]="alpha"; print(mat)
}

```

Description

rivGibbs is a Gibbs Sampler for a linear structural equation with an arbitrary number of instruments.

Usage

```
rivGibbs(Data, Prior, Mcmc)
```

Arguments

Data	list(z,w,x,y)
Prior	list(md,Ad,mbg,Abg,nu,V) (optional)
Mcmc	list(R,keep) (R required)

Details

Model:

$$x = z'\delta + e1.$$

$$y = \beta x + w'\gamma + e2.$$

$$e1, e2 \sim N(0, \Sigma).$$

Priors:

$$\delta \sim N(md, Ad^{-1}). \text{vec}(\beta, \gamma) \sim N(mbg, Abg^{-1})$$

$$\Sigma \sim IW(nu, V)$$

List arguments contain:

- z matrix of obs on instruments
- y vector of obs on lhs var in structural equation
- x "endogenous" var in structural eqn
- w matrix of obs on "exogenous" vars in the structural eqn
- md prior mean of delta (def: 0)
- Ad pds prior prec for prior on delta (def: .01I)
- mbg prior mean vector for prior on beta,gamma (def: 0)
- Abg pds prior prec for prior on beta,gamma (def: .01I)
- nu d.f. parm for IW prior on Sigma (def: 5)
- V pds location matrix for IW prior on Sigma (def: nuI)
- R number of MCMC draws
- keep MCMC thinning parm: keep every keepth draw (def: 1)

Value

a list containing:

deltadraw	R/keep x dim(delta) array of delta draws
betadraw	R/keep x 1 vector of beta draws
gammadraw	R/keep x dim(gamma) array of gamma draws
Sigmadraw	R/keep x 4 array of Sigma draws

Author(s)

Rob McCulloch and Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 5.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```
##
if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=2000} else {R=10}

set.seed(66)
simIV = function(delta,beta,Sigma,n,z,w,gamma) {
  eps = matrix(rnorm(2*n),ncol=2) %%% chol(Sigma)
  x = z %%% delta + eps[,1]; y = beta*x + eps[,2] + w%%gamma
  list(x=as.vector(x),y=as.vector(y)) }
n = 200 ; p=1 # number of instruments
z = cbind(rep(1,n),matrix(runif(n*p),ncol=p))
w = matrix(1,n,1)
rho=.8
Sigma = matrix(c(1,rho,rho,1),ncol=2)
delta = c(1,4); beta = .5; gamma = c(1)
simiv = simIV(delta,beta,Sigma,n,z,w,gamma)

Mcmc=list(); Prior=list(); Data = list()
Data$z = z; Data$w=w; Data$x=simiv$x; Data$y=simiv$y
Mcmc$R = R
Mcmc$keep=1
out=rivGibbs(Data=Data,Prior=Prior,Mcmc=Mcmc)

cat(" deltadraws ",fill=TRUE)
mat=apply(out$deltadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(delta,mat); rownames(mat)[1]="delta"; print(mat)
cat(" betadraws ",fill=TRUE)
qout=quantile(out$betadraw,probs=c(.01,.05,.5,.95,.99))
mat=matrix(qout,ncol=1)
mat=rbind(beta,mat); rownames(mat)=c("beta",names(qout)); print(mat)
cat(" Sigma draws",fill=TRUE)
mat=apply(out$Sigmadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(Sigma),mat); rownames(mat)[1]="Sigma"; print(mat)
```

rmixGibbs

*Gibbs Sampler for Normal Mixtures w/o Error Checking***Description**

rmixGibbs makes one draw using the Gibbs Sampler for a mixture of multivariate normals.

Usage

```
rmixGibbs(y, Bbar, A, nu, V, a, p, z, comps)
```

Arguments

y	data array - rows are obs
Bbar	prior mean for mean vector of each norm comp
A	prior precision parameter
nu	prior d.f. parm
V	prior location matrix for covariance priro

a	Dirichlet prior parms
p	prior prob of each mixture component
z	component identities for each observation – "indicators"
comps	list of components for the normal mixture

Details

rmixGibbs is not designed to be called directly. Instead, use rnmixGibbs wrapper function.

Value

a list containing:

p	draw mixture probabilities
z	draw of indicators of each component
comps	new draw of normal component parameters

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Rob McCulloch and Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Allenby, McCulloch, and Rossi, Chapter 5.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rnmixGibbs](#)

rmixture

Draw from Mixture of Normals

Description

rmixture simulates iid draws from a Multivariate Mixture of Normals

Usage

```
rmixture(n, pvec, comps)
```

Arguments

n	number of observations
pvec	ncomp x 1 vector of prior probabilities for each mixture component
comps	list of mixture component parameters

Details

comps is a list of length, ncomp = length(pvec). comps[[j]][[1]] is mean vector for the jth component. comps[[j]][[2]] is the inverse of the cholesky root of Sigma for that component

Value

A list containing ...

x	An n x length(comps[[1]][[1]]) array of iid draws
z	A n x 1 vector of indicators of which component each draw is taken from

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

See Also

[rnmixGibbs](#)

rmnlIndepMetrop	<i>MCMC Algorithm for Multinomial Logit Model</i>
-----------------	---

Description

rmnlIndepMetrop implements Independence Metropolis for the MNL.

Usage

```
rmnlIndepMetrop(Data, Prior, Mcmc)
```

Arguments

Data	list(p,y,X)
Prior	list(A,betabar) optional
Mcmc	list(R,keep,nu)

Details

Model: $y \sim \text{MNL}(X, \beta)$. $Pr(y = j) = \exp(x'_j \beta) / \sum_k \exp(x'_k \beta)$.

Prior: $\beta \sim N(\text{betabar}, A^{-1})$

list arguments contain:

p	number of alternatives
y	nobs vector of multinomial outcomes (1, ..., p)

`X` nobs*p x nvar matrix
`A` nvar x nvar pds prior prec matrix (def: .01I)
`betabar` nvar x 1 prior mean (def: 0)
`R` number of MCMC draws
`keep` MCMC thinning parm: keep every keepth draw (def: 1)
`nu` degrees of freedom parameter for independence t density (def: 6)

Value

a list containing:

`betadraw` R/keep x nvar array of beta draws
`acceptr` acceptance rate of Metropolis draws

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 5.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rhierMnlRwMixture](#)

Examples

```
##

if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=2000} else {R=10}

set.seed(66)
n=200; p=3; beta=c(1,-1,1.5,.5)
simout=simmnl(p,n,beta)
A=diag(c(rep(.01,length(beta)))); betabar=rep(0,length(beta))

Data=list(y=simout$y,X=simout$X,p=p); Mcmc=list(R=R,keep=1) ; Prior=list(A=A,betabar=beta)
out=rmnlIndepMetrop(Data=Data,Prior=Prior,Mcmc=Mcmc)
cat(" Betadraws ",fill=TRUE)
mat=apply(out$betadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(beta,mat); rownames(mat)[1]="beta"; print(mat)
```

Description

rmnpGibbs implements the McCulloch/Rossi Gibbs Sampler for the multinomial probit model.

Usage

```
rmnpGibbs(Data, Prior, Mcmc)
```

Arguments

Data	list(p, y, X)
Prior	list(betabar,A,nu,V) (optional)
Mcmc	list(beta0,sigma0,R,keep) (R required)

Details

model:

$w_i = X_i\beta + e$. $e \sim N(0, \text{Sigma})$. note: w_i, e are $(p-1) \times 1$.

$y_i = j$, if $w_{ij} > \max(0, w_{i,-j})$ $j=1, \dots, p-1$. $w_{i,-j}$ means elements of w_i other than the j th.

$y_i = p$, if all $w_i < 0$.

priors:

$\beta \sim N(\text{betabar}, A^{-1})$

$\text{Sigma} \sim \text{IW}(\text{nu}, V)$

to make up X matrix use `createX` with `DIFF=TRUE`.

List arguments contain

p	number of choices or possible multinomial outcomes
y	$n \times 1$ vector of multinomial outcomes
X	$n \times (p-1) \times k$ Design Matrix
betabar	$k \times 1$ prior mean (def: 0)
A	$k \times k$ prior precision matrix (def: $.01I$)
nu	d.f. parm for IWishart prior (def: $(p-1) + 3$)
V	pds location parm for IWishart prior (def: $\text{nu} \times I$)
beta0	initial value for beta
sigma0	initial value for sigma
R	number of MCMC draws
keep	thinning parameter - keep every keepth draw (def: 1)

Value

a list containing:

betadraw	$R/\text{keep} \times k$ array of betadraws
sigmadraw	$R/\text{keep} \times (p-1) \times (p-1)$ array of sigma draws – each row is in vector form

Note

beta is not identified. $\text{beta}/\sqrt{\text{sigma}_{11}}$ and $\text{Sigma}/\text{sigma}_{11}$ are. See Allenby et al or example below for details.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 4.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rmvpGibbs](#)

Examples

```
##
if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=2000} else {R=10}

set.seed(66)
p=3
n=500
beta=c(-1,1,1,2)
Sigma=matrix(c(1,.5,.5,1),ncol=2)
k=length(beta)
x1=runif(n*(p-1),min=-1,max=1); x2=runif(n*(p-1),min=-1,max=1)
I2=diag(rep(1,p-1)); xadd=rbind(I2)
for(i in 2:n) { xadd=rbind(xadd,I2)}
X=cbind(xadd,x1,x2)
simout=simmnp(X,p,500,beta,Sigma)

Data=list(p=p,y=simout$y,X=simout$X)
Mcmc=list(R=R,keep=1)

out=rmnpGibbs(Mcmc=Mcmc,Data=Data)

cat(" Betadraws ",fill=TRUE)
mat=apply(out$betadraw/sqrt(out$sigmadraw[,1]),2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(beta,mat); rownames(mat)[1]="beta"; print(mat)
cat(" Sigmadraws ",fill=TRUE)
mat=apply(out$sigmadraw/out$sigmadraw[,1],2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(Sigma),mat); rownames(mat)[1]="sigma"; print(mat)
```

Description

rmultireg draws from the posterior of a Multivariate Regression model with a natural conjugate prior.

Usage

```
rmultireg(Y, X, Bbar, A, nu, V)
```

Arguments

Y	n x m matrix of observations on m dep vars
X	n x k matrix of observations on indep vars (supply intercept)
Bbar	k x m matrix of prior mean of regression coefficients
A	k x k Prior precision matrix
nu	d.f. parameter for Sigma
V	m x m pdf location parameter for prior on Sigma

Details

Model: $Y = XB + U$. $cov(u_i) = Sigma$. B is k x m matrix of coefficients. $Sigma$ is m x m covariance.

Priors: $beta$ given $Sigma \sim N(betabar, Sigma(x)A^{-1})$. $betabar = vec(Bbar)$; $beta = vec(B)$
 $Sigma \sim IW(nu, V)$.

Value

A list of the components of a draw from the posterior

B	draw of regression coefficient matrix
Sigma	draw of Sigma

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, (Peter.Rossi@ChicagoGsb.edu).

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 2.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rmultiregfp](#), [init.rmultiregfp](#)

Examples

```
##
if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=2000} else {R=10}

set.seed(66)
n=200
m=2
X=cbind(rep(1,n),runif(n))
k=ncol(X)
B=matrix(c(1,2,-1,3),ncol=m)
Sigma=matrix(c(1,.5,.5,1),ncol=m); RSigma=chol(Sigma)
Y=X%*%B+matrix(rnorm(m*n),ncol=m)%*%RSigma

betabar=rep(0,k*m);Bbar=matrix(betabar,ncol=m)
A=diag(rep(.01,k))
nu=3; V=nu*diag(m)

betadraw=matrix(double(R*k*m),ncol=k*m)
Sigmadraw=matrix(double(R*m*m),ncol=m*m)
for (rep in 1:R)
  {out=rmultiregfp(Y,X,Bbar,A,nu,V);betadraw[rep,]=out$B
    Sigmadraw[rep,]=out$Sigma}

cat(" Betadraws ",fill=TRUE)
mat=apply(betadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(B),mat); rownames(mat)[1]="beta"
print(mat)
cat(" Sigma draws",fill=TRUE)
mat=apply(Sigmadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(Sigma),mat); rownames(mat)[1]="Sigma"
print(mat)
```

rmultiregfp

Draw from the Posterior of a Multivariate Regression

Description

rmultiregfp draws from the posterior of a Multivariate Regression model with a natural conjugate prior.

Usage

```
rmultiregfp(Y, X, Fparm)
```

Arguments

Y	n x m matrix of observations on m dep vars
X	n x k matrix of observations on indep vars (supply intercept)
Fparm	a list of prior parameters prepared by <code>init.rmultiregfp</code>

Details

Model: $Y = XB + U$. $cov(u_i) = Sigma$. B is $k \times m$ matrix of coefficients. $Sigma$ is an $m \times m$ covariance matrix.

Priors: $beta$ given $Sigma \sim N(betabar, Sigma(x)A^{-1})$. $betabar = vec(Bbar)$; $beta = vec(B)$. $Sigma \sim IW(nu, V)$.

prepare Fparm by call `init.rmultiregfp`

Value

A list of the components of a draw from the posterior

B draw of regression coefficient matrix

$Sigma$ draw of $Sigma$

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

`rmultireg`, `init.rmultiregfp`

rmvpGibbs

Gibbs Sampler for Multivariate Probit

Description

rmvpGibbs implements the Edwards/Allenby Gibbs Sampler for the multivariate probit model.

Usage

```
rmvpGibbs(Data, Prior, Mcmc)
```

Arguments

$Data$ list(p,y,X)

$Prior$ list(betabar,A,nu,V) (optional)

$Mcmc$ list(beta0,sigma0,R,keep) (R required)

Details

model:

$w_i = X_i \beta + e$. $e \sim N(0, \Sigma)$. note: w_i is $p \times 1$.

$y_{ij} = 1$, if $w_{ij} > 0$, else $y_{ij} = 0$. $j=1, \dots, p$.

priors:

$\beta \sim N(\beta_{\text{bar}}, A^{-1})$

$\Sigma \sim \text{IW}(\nu, V)$

to make up X matrix use createX

List arguments contain

p dimension of multivariate probit

X $n \times p \times k$ Design Matrix

y $n \times p \times 1$ vector of 0,1 outcomes

betabar $k \times 1$ prior mean (def: 0)

A $k \times k$ prior precision matrix (def: .01I)

nu d.f. parm for IWishart prior (def: (p-1) + 3)

V pds location parm for IWishart prior (def: nu*I)

beta0 initial value for beta

sigma0 initial value for sigma

R number of MCMC draws

keep thinning parameter - keep every keepth draw (def: 1)

Value

a list containing:

betadraw $R/\text{keep} \times k$ array of betadraws

sigmadraw $R/\text{keep} \times p \times p$ array of sigma draws – each row is in vector form

Note

beta and Sigma are not identified. Correlation matrix and the betas divided by the appropriate standard deviation are. See Allenby et al for details or example below.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, (Peter.Rossi@ChicagoGsb.edu).

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 4.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rmnpGibbs](#)

Examples

```
##
if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=2000} else {R=10}

set.seed(66)
p=3
n=500
beta=c(-2,0,2)
Sigma=matrix(c(1,.5,.5,.5,1,.5,.5,.5,1),ncol=3)
k=length(beta)
I2=diag(rep(1,p)); xadd=rbind(I2)
for(i in 2:n) { xadd=rbind(xadd,I2)}; X=xadd
simout=simmvp(X,p,500,beta,Sigma)

Data=list(p=p,y=simout$y,X=simout$X)
Mcmc=list(R=R,keep=1)
out=rmvpGibbs(Data=Data,Mcmc=Mcmc)

ind=seq(from=0,by=p,length=k)
inda=1:3
ind=ind+inda
cat(" Betadraws ",fill=TRUE)
mat=apply(out$betadraw/sqrt(out$sigmadraw[,ind]),2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(beta,mat); rownames(mat)[1]="beta"; print(mat)
rdraw=matrix(double((R)*p*p),ncol=p*p)
rdraw=t(apply(out$sigmadraw,1,nmat))
cat(" Draws of Correlation Matrix ",fill=TRUE)
mat=apply(rdraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(Sigma),mat); rownames(mat)[1]="Sigma"; print(mat)
```

rmvst

Draw from Multivariate Student-t

Description

rmvst draws from a Multivariate student-t distribution.

Usage

```
rmvst(nu, mu, root)
```

Arguments

nu	d.f. parameter
mu	mean vector
root	Upper Tri Cholesky Root of Sigma

Value

length(mu) draw vector

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, (Peter.Rossi@ChicagoGsb.edu).

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[lndMvst](#)

Examples

```
##
set.seed(66)
rmvst(nu=5,mu=c(rep(0,2)),root=chol(matrix(c(2,1,1,2),ncol=2)))
```

rnegbinRw

MCMC Algorithm for Negative Binomial Regression

Description

rnegbinRw implements a Random Walk Metropolis Algorithm for the Negative Binomial (NBD) regression model. beta | alpha and alpha | beta are drawn with two different random walks.

Usage

```
rnegbinRw(Data, Prior, Mcmc)
```

Arguments

Data	list(y,X)
Prior	list(betabar,A,a,b)
Mcmc	list(R,keep,s_beta,s_alpha,beta0)

Details

Model: $y \sim NBD(\text{mean} = \lambda, \text{over} - \text{dispersion} = \alpha)$.
 $\lambda = \exp(x' \beta)$

Prior: $\beta \sim N(\text{betabar}, A^{-1})$

$\alpha \sim \text{Gamma}(a, b)$.

note: prior mean of $\alpha = a/b$, variance = $a/(b^2)$

list arguments contain:

y nobs vector of counts (0,1,2,...)

X nobs x nvar matrix
 betabar nvar x 1 prior mean (def: 0)
 A nvar x nvar pds prior prec matrix (def: .01I)
 a Gamma prior parm (def: .5)
 b Gamma prior parm (def: .1)
 R number of MCMC draws
 keep MCMC thinning parm: keep every keepth draw (def: 1)
 s_beta scaling for beta | alpha RW inc cov matrix (def: 2.93/sqrt(nvar))
 s_alpha scaling for alpha | beta RW inc cov matrix (def: 2.93)

Value

a list containing:

betadraw R/keep x nvar array of beta draws
 alphadraw R/keep vector of alpha draws
 llike R/keep vector of log-likelihood values evaluated at each draw
 acceptrbeta acceptance rate of the beta draws
 acceptralp alpha acceptance rate of the alpha draws

Note

The NBD regression encompasses Poisson regression in the sense that as alpha goes to infinity the NBD distribution tends toward the Poisson.
 For "small" values of alpha, the dependent variable can be extremely variable so that a large number of observations may be required to obtain precise inferences.

Author(s)

Sridhar Narayanam & Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby, McCulloch.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rhierNegbinRw](#)

Examples

```
##
if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=1000} else {R=10}

set.seed(66)
simnegbin =
function(X, beta, alpha) {
# Simulate from the Negative Binomial Regression
lambda = exp(X %*% beta)
y=NULL
```

```

for (j in 1:length(lambda))
  y = c(y,rnbinom(1,mu = lambda[j],size = alpha))
return(y)
}

nobs = 500
nvar=2          # Number of X variables
alpha = 5
Vbeta = diag(nvar)*0.01

# Construct the regdata (containing X)
simnegbindata = NULL
beta = c(0.6,0.2)
X = cbind(rep(1,nobs),rnorm(nobs,mean=2,sd=0.5))
simnegbindata = list(y=simnegbin(X,beta,alpha), X=X, beta=beta)
Data = simnegbindata
betabar = rep(0,nvar)
A = 0.01 * diag(nvar)
a = 0.5; b = 0.1
Prior = list(betabar=betabar, A=A, a=a, b=b)

keep =1
s_beta=2.93/sqrt(nvar); s_alpha=2.93
Mcmc = list(R=R, keep = keep, s_beta=s_beta, s_alpha=s_alpha)
out = rneginRw(Data, Prior, Mcmc)

cat(" betadraws ",fill=TRUE)
mat=apply(out$betadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(beta,mat); rownames(mat)[1]="beta"; print(mat)
cat(" alphadraws ",fill=TRUE)
mat=apply(matrix(out$alphadraw),2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(alpha),mat); rownames(mat)[1]="alpha"; print(mat)

```

rnmixGibbs

Gibbs Sampler for Normal Mixtures

Description

rnmixGibbs implements a Gibbs Sampler for normal mixtures.

Usage

```
rnmixGibbs(Data, Prior, Mcmc)
```

Arguments

Data	list(y)
Prior	list(Mubar,A,nu,V,a,ncomp) (only ncomp required)
Mcmc	list(R,keep) (R required)

Details

Model:

$$y_i \sim N(mu_{ind_i}, Sigma_{ind_i}).$$

ind \sim iid multinomial(p). p is a ncomp x 1 vector of probs.

Priors:

$$mu_j \sim N(mubar, Sigma_j(x)A^{-1}). mubar = vec(Mubar).$$

$$Sigma_j \sim IW(nu, V).$$

note: this is the natural conjugate prior – a special case of multivariate regression.

$$p \sim \text{Dirchlet}(a).$$

Output of the components is in the form of a list of lists.

compsdraw[[i]] is ith draw – list of ncomp lists.

compsdraw[[i]][[j]] is list of parms for jth normal component.

jcomp=compsdraw[[i]][j]. Then jth comp $\sim N(jcomp[[1]], Sigma)$, $Sigma = t(R) \%*\% R$, $R^{-1} = jcomp[[2]]$.

List arguments contain:

y n x k array of data (rows are obs)

Mubar 1 x k array with prior mean of normal comp means (def: 0)

A 1 x 1 precision parameter for prior on mean of normal comp (def: .01)

nu d.f. parameter for prior on Sigma (normal comp cov matrix) (def: k+3)

V k x k location matrix of IW prior on Sigma (def: nuI)

a ncomp x 1 vector of Dirichlet prior parms (def: rep(5,ncomp))

ncomp number of normal components to be included

R number of MCMC draws

keep MCMC thinning parm: keep every keepth draw (def: 1)

Value

a list containing:

probdraw R/keep x ncomp array of mixture prob draws

zdraw R/keep x nobs array of indicators of mixture comp identity for each obs

compdraw R/keep lists of lists of comp parm draws

Note

In this model, the component normal parameters are not-identified due to label-switching. However, the fitted mixture of normals density is identified as it is invariant to label-switching. See Allenby et al, chapter 5 for details. Use eMixMargDen or momMix to compute posterior expectation or distribution of various identified parameters.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 3.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rmixture](#), [rmixGibbs](#), [eMixMargDen](#), [momMix](#), [mixDen](#), [mixDenBi](#)

Examples

```
##
if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=2000} else {R=10}

set.seed(66)
dim=5; k=3 # dimension of simulated data and number of "true" components
sigma = matrix(rep(0.5,dim^2),nrow=dim);diag(sigma)=1
sigfac = c(1,1,1);mufac=c(1,2,3); compsmv=list()
for(i in 1:k) compsmv[[i]] = list(mu=mufac[i]*1:dim,sigma=sigfac[i]*sigma)
comps = list() # change to "rooti" scale
for(i in 1:k) comps[[i]] = list(mu=compsmv[[i]][[1]],rooti=solve(chol(compsmv[[i]][[2]]))
pvec=(1:k)/sum(1:k)

nobs=5000
dm = rmixture(nobs,pvec,comps)

Data=list(y=dm$x)
ncomp=9
Prior=list(ncomp=ncomp)
Mcmc=list(R=R,keep=1)
out=rnmixGibbs(Data=Data,Prior=Prior,Mcmc=Mcmc)

tmom=momMix(matrix(pvec,nrow=1),list(comps))
if(R < 1000) {begin=1} else {begin=500}
pmom=momMix(out$probdraw[begin:R,],out$compdraw[begin:R])
mat=rbind(tmom$mu,pmom$mu)
rownames(mat)=c("true","post expect")
cat(" mu and posterior expectation of mu",fill=TRUE)
print(mat)
mat=rbind(tmom$sd,pmom$sd)
rownames(mat)=c("true","post expect")
cat(" std dev and posterior expectation of sd",fill=TRUE)
print(mat)
mat=rbind(as.vector(tmom$corr),as.vector(pmom$corr))
rownames(mat)=c("true","post expect")
cat(" corr and posterior expectation of corr",fill=TRUE)
print(t(mat))

if(0){
##
## plotting examples
##
## check true and estimated marginal densities
grid=NULL
for (i in 1:dim){
  rgi=range(dm$x[,i])
  gr=seq(from=rgi[1],to=rgi[2],length.out=50)
  grid=cbind(grid,gr)
}

tmdden=mixDen(grid,pvec,comps)
pmden=eMixMargDen(grid,out$probdraw[begin:end,],out$compdraw[begin:end])
```

```

## plot the marginal on third variable
plot(range(grid[,3]),c(0,1.1*max(tmden[,3],pmden[,3])),type="n",xlab="",ylab="density")
lines(grid[,3],tmden[,3],col="blue",lwd=2)
lines(grid[,3],pmden[,3],col="red",lwd=2)

## compute implied bivariate marginal distributions
i=1
j=2
rxj=range(dm$x[,1])
rxj=range(dm$x[,2])
xi=seq(from=rxj[1],to=rxj[2],length.out=50)
xj=seq(from=rxj[1],to=rxj[2],length.out=50)
den=matrix(0,ncol=length(xi),nrow=length(xj))
for(ind in as.integer(seq(from=begin,to=end,length.out=100))) {
  den=den+mixDenBi(i,j,xi,xj,out$probdraw[ind,],out$compdraw[[ind]])
}
tden=matrix(0,ncol=length(xi),nrow=length(xj))
tden=mixDenBi(i,j,xi,xj,pvec,comps)

par(mfrow=c(2,1))
image(xi,xj,tden,col=terrain.colors(100),xlab="",ylab="")
contour(xi,xj,den,add=TRUE,drawlabels=FALSE)
title("True Bivariate Marginal")
image(xi,xj,den,col=terrain.colors(100),xlab="",ylab="")
contour(xi,xj,den,add=TRUE,drawlabels=FALSE)
title("Posterior Mean of Bivariate Marginal")
}

```

rscaleUsage	<i>MCMC Algorithm for Multivariate Ordinal Data with Scale Usage Heterogeneity.</i>
-------------	---

Description

rscaleUsage implements an MCMC algorithm for multivariate ordinal data with scale usage heterogeneity.

Usage

```
rscaleUsage(Data,Prior, Mcmc)
```

Arguments

Data	list(k,x)
Prior	list(nu,V,mubar,Am,gsigma,gl11,gl22,gl12,Lambdanu,LambdaV,ge) (optional)
Mcmc	list(R,keep,ndghk,printevery,e,y,mu,Sigma,sigma,tau,Lambda) (optional)

Details

Model: $n = \text{nrow}(x)$ individuals respond to $m = \text{ncol}(x)$ questions. all questions are on a scale $1, \dots, k$. for respondent i and question j ,

$$x_{ij} = d, \text{ if } c_{d-1} \leq y_{ij} \leq c_d.$$

$$d = 1, \dots, k. \quad c_d = a + bd + ed^2.$$

$$y_i = \mu + \tau_i * \text{iota} + \sigma_i * z_i. \quad z_i \sim N(0, \text{Sigma}).$$

Priors:

$$(\tau_i, \ln(\sigma_i)) \sim N(\phi, \text{Lamda}). \quad \phi = (0, \text{lambda}_{22}).$$

$$\mu \sim N(\text{mubar}, A m^{-1}).$$

$$\text{Sigma} \sim \text{IW}(\text{nu}, V).$$

$$\text{Lambda} \sim \text{IW}(\text{Lambdanu}, \text{LambdaV}).$$

$$e \sim \text{unif on a grid}.$$

Value

a list containing:

Sigmadraw	R/keep x m*m array of Sigma draws
mudraw	R/keep x m array of mu draws
taudraw	R/keep x n array of tau draws
sigmadraw	R/keep x n array of sigma draws
Lambdadraw	R/keep x 4 array of Lamda draws
edraw	R/keep x 1 array of e draws

Warning

τ_i, σ_i are identified from the scale usage patterns in the m questions asked per respondent (# cols of x). Do not attempt to use this on data sets with only a small number of total questions!

Note

It is **highly** recommended that the user choose the default settings. This means not specifying the argument `Prior` and setting `R` in `Mcmc` and `Data` only. If you wish to change prior settings and/or the grids used, please read the case study in Allenby et al carefully.

Author(s)

Rob McCulloch and Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby, and McCulloch, Case Study on Scale Usage Heterogeneity.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```
##
if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=1000} else {R=5}
{
  data(customerSat)
  surveydat = list(k=10,x=as.matrix(customerSat))

  mcmc = list(R=R)
  set.seed(66)
  out=rscaleUsage(Data=surveydat,Mcmc=mcmc)

  cat(" mudraws ",fill=TRUE)
  mat=apply(out$mudraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
  print(mat)

}
```

rsurGibbs

Gibbs Sampler for Seemingly Unrelated Regressions (SUR)

Description

rsurGibbs implements a Gibbs Sampler to draw from the posterior of the Seemingly Unrelated Regression (SUR) Model of Zellner

Usage

```
rsurGibbs(Data, Prior, Mcmc)
```

Arguments

Data	list(regdata)
Prior	list(betabar,A, nu, V)
Mcmc	list(R,keep)

Details

Model: $y_i = X_i \beta_i + e_i$, $i=1, \dots, m$. m regressions.
 $(e(1,k), \dots, e(m,k)) \sim N(0, \Sigma)$. $k=1, \dots, \text{nobs}$.

We can also write as the stacked model:

$y = X\beta + e$ where y is a $\text{nobs} \times m$ long vector and $k=\text{length}(\beta)=\text{sum}(\text{length}(\beta_i))$.

Note: we must have the same number of observations in each equation but we can have different numbers of X variables

Priors: $\beta \sim N(\text{betabar}, A^{-1})$. $\Sigma \sim IW(\text{nu}, V)$.

List arguments contain

regdata list of lists, regdata[[i]]=list(y=yi,X=Xi)

betabar $k \times 1$ prior mean (def: 0)

A $k \times k$ prior precision matrix (def: .01I)

nu d.f. parm for Inverted Wishart prior (def: $m+3$)

V scale parm for Inverted Wishart prior (def: nu*I)
 R number of MCMC draws
 keep thinning parameter - keep every keepth draw

Value

list of MCMC draws
 betadraw R x k array of betadraws
 Sigmadraw R x (m*m) array of Sigma draws

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 3.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rmultireg](#)

Examples

```
if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=1000} else {R=10}
##
## simulate data from SUR
set.seed(66)
betal=c(1,2)
beta2=c(1,-1,-2)
nobs=100
nreg=2
iota=c(rep(1,nobs))
X1=cbind(iota,runif(nobs))
X2=cbind(iota,runif(nobs),runif(nobs))
Sigma=matrix(c(.5,.2,.2,.5),ncol=2)
U=chol(Sigma)
E=matrix(rnorm(2*nobs),ncol=2)
y1=X1%%betal+E[,1]
y2=X2%%beta2+E[,2]
##
## run Gibbs Sampler
regdata=NULL
regdata[[1]]=list(y=y1,X=X1)
regdata[[2]]=list(y=y2,X=X2)
Mcmc=list(R=R)
out=rsurGibbs(Data=list(regdata=regdata),Mcmc=Mcmc)
##
## summarize GS output
if(R < 100) {begin=1} else {begin=100}
end=Mcmc$R
cat(" betadraws ",fill=TRUE)
```

```

mat=apply(out$betadraw[begin:end,],2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(c(beta1,beta2),mat); rownames(mat)[1]="beta"; print(mat)
cat(" Sigmadraws ",fill=TRUE)
mat=apply(out$Sigmadraw[begin:end,],2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(as.vector(Sigma),mat); rownames(mat)[1]="sigma"; print(mat)

```

rtrun

*Draw from Truncated Univariate Normal***Description**

rtrun draws from a truncated univariate normal distribution

Usage

```
rtrun(mu, sigma, a, b)
```

Arguments

mu	mean
sigma	sd
a	lower bound
b	upper bound

Details

Note that due to the vectorization of the `rnorm`, `qnorm` commands in R, all arguments can be vectors of equal length. This makes the inverse CDF method the most efficient to use in R.

Value

draw (possibly a vector)

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 2.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```

##
set.seed(66)
rtrun(mu=c(rep(0,10)),sigma=c(rep(1,10)),a=c(rep(0,10)),b=c(rep(2,10)))

```

runireg

*IID Sampler for Univariate Regression***Description**

runireg implements an iid sampler to draw from the posterior of a univariate regression with a conjugate prior.

Usage

```
runireg(Data, Prior, Mcmc)
```

Arguments

Data	list(y,X)
Prior	list(betabar,A, nu, ssq)
Mcmc	list(R,keep)

Details

Model: $y = X\beta + e$. $e \sim N(0, \sigma^2)$.

Priors: $\beta \sim N(\text{betabar}, \sigma^2 A^{-1})$. $\sigma^2 \sim (nu * ssq) / \chi^2_{nu}$. List arguments contain

X $n \times k$ Design Matrix
 y $n \times 1$ vector of observations
 betabar $k \times 1$ prior mean (def: 0)
 A $k \times k$ prior precision matrix (def: .01I)
 nu d.f. parm for Inverted Chi-square prior (def: 3)
 ssq scale parm for Inverted Chi-square prior (def: var(y))
 R number of draws
 keep thinning parameter - keep every keepth draw

Value

list of iid draws
 betadraw $R \times k$ array of betadraws
 sigmasqdraw R vector of sigma-sq draws

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 2.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also[runiregGibbs](#)**Examples**

```

if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=2000} else {R=10}
set.seed(66)
n=200
X=cbind(rep(1,n),runif(n)); beta=c(1,2); sigsq=.25
y=X%*%beta+rnorm(n,sd=sqrt(sigsq))

out=runireg(Data=list(y=y,X=X),Mcmc=list(R=R))
cat(" betadraws ",fill=TRUE)
mat=apply(out$betadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(beta,mat); rownames(mat)[1]="beta"; print(mat)
cat(" Sigma-sq draws",fill=TRUE)
cat(" sigma-sq= ",sigsq,fill=TRUE)
print(quantile(out$sigmasqdraw,probs=c(.01,.05,.5,.95,.99)))

```

runiregGibbs

*Gibbs Sampler for Univariate Regression***Description**

runiregGibbs implements a Gibbs Sampler to draw from posterior of a univariate regression with a conditionally conjugate prior.

Usage

```
runiregGibbs(Data, Prior, Mcmc)
```

Arguments

Data	list(y,X)
Prior	list(betabar,A, nu, ssq)
Mcmc	list(sigmasq,R,keep)

Details

Model: $y = X\beta + e$. $e \sim N(0, \text{sigmasq})$.

Priors: $\beta \sim N(\text{betabar}, A^{-1})$. $\text{sigmasq} \sim (nu * \text{ssq}) / \text{chisq}_{nu}$. List arguments contain

X	n x k Design Matrix
y	n x 1 vector of observations
betabar	k x 1 prior mean (def: 0)
A	k x k prior precision matrix (def: .01I)
nu	d.f. parm for Inverted Chi-square prior (def: 3)
ssq	scale parm for Inverted Chi-square prior (def: var(y))
R	number of MCMC draws
keep	thinning parameter - keep every keepth draw

Value

list of MCMC draws

betadraw R x k array of betadraws

sigmasqdraw R vector of sigma-sq draws

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 3.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[runireg](#)

Examples

```
if(nchar(Sys.getenv("LONG_TEST")) != 0) {R=1000} else {R=10}
set.seed(66)
n=100
X=cbind(rep(1,n),runif(n)); beta=c(1,2); sigsq=.25
y=X*%beta+rnorm(n,sd=sqrt(sigsq))

A=diag(c(.05,.05)); betabar=c(0,0)
nu=3; ssq=1.0

Data=list(y=y,X=X); Mcmc=list(R=R,keep=1) ; Prior=list(A=A,betabar=betabar,nu=nu,ssq=ssq)
out=runiregGibbs(Data=Data,Prior=Prior,Mcmc=Mcmc)
cat(" betadraws ",fill=TRUE)
mat=apply(out$betadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
mat=rbind(beta,mat); rownames(mat)[1]="beta"; print(mat)
cat(" Sigma-sq draws",fill=TRUE)
cat(" sigma-sq= ",sigsq,fill=TRUE)
print(quantile(out$sigmasqdraw,probs=c(.01,.05,.5,.95,.99)))
```

rwishart

Draw from Wishart and Inverted Wishart Distribution

Description

rwishart draws from the Wishart and Inverted Wishart distributions.

Usage

```
rwishart(nu, V)
```

Arguments

nu	d.f. parameter
V	pds location matrix

Details

In the parameterization used here, $W \sim W(nu, V)$, $E[W] = nuV$.

If you want to use an Inverted Wishart prior, you *must invert the location matrix* before calling `rwishart`, e.g.

$Sigma \sim IW(nu, V)$; $Sigma^{-1} \sim W(nu, V^{-1})$.

Value

W	Wishart draw
IW	Inverted Wishart draw
C	Upper tri root of W
CI	$inv(C)$, $W^{-1} = CICI'$

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 2.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

Examples

```
##
set.seed(66)
rwishart(5, diag(3))$IW
```

simnml

Simulate from Multinomial Logit Model

Description

`simnml` simulates from the MNL model.

Usage

```
simnml(p, n, beta)
```

Arguments

p	number choice alternatives
n	number of observations
beta	MNL coefficient vector

Details

simnml will simulate two uniformly distributed X vars and add intercepts.

Value

y	n x 1 vector of multinomial outcomes (1, ..., p)
X	
beta	beta vector
prob	n x j array of choice probabilities

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby, and McCulloch.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[llmnl](#), [rmnlIndepMetrop](#)

simnmlwX

Simulate from MNL given X Matrix

Description

simnmlwX simulates from MNL given X Matrix.

Usage

```
simnmlwX(n, X, beta)
```

Arguments

n	number of obs
X	n*p x k Design matrix (p is number of choice alts)
beta	k x 1 coefficient vector

Value

a list containing:

y	n x 1 vector of multinomial outcomes (1, ..., nrow(X)/n)
x	Design matrix
beta	coefficient vector
prob	n x p array of choice probs

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, (Peter.Rossi@ChicagoGsb.edu).

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch.
<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[simmnl](#)

simmnp

Simulate from Multinomial Probit Model

Description

simmvp simulates from the multinomial probit model.

Usage

```
simmnp(X, p, n, beta, sigma)
```

Arguments

x	n*(p-1) x length(beta) Design matrix
p	number of choice alternatives
n	number of observations
beta	coefficient vector
sigma	(p-1) x (p-1) covariance matrix

Value

a list of

y	n vector of multinomial (1, ..., p) outcomes
X	Design matrix
beta	coefficients
sigma	covariance matrix

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, <Peter.Rossi@ChicagoGsb.edu>.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 4.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rmnpGibbs](#)

simmvp

Simulate from Multivariate Probit Model

Description

simmvp simulates from the multivariate probit model.

Usage

```
simmvp(X, p, n, beta, sigma)
```

Arguments

X	n*p x length(beta) Design matrix
p	dimension of the MVP
n	number of observations
beta	coefficient vector
sigma	p x p covariance matrix

Value

a list of	
y	p*n vector of 0/1 binary outcomes
x	Design matrix
beta	coefficients
sigma	covariance matrix

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 4.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[rmvpGibbs](#)

simnhlogit

Simulate from Non-homothetic Logit Model

Description

simnhlogit simulates from the non-homothetic logit model

Usage

```
simnhlogit(theta, lnprices, Xexpend)
```

Arguments

theta	coefficient vector
lnprices	n x p array of prices
Xexpend	n x k array of values of expenditure variables

Details

For detail on parameterization, see `llnhlogit`.

Value

a list containing:

<code>y</code>	<code>n x 1</code> vector of multinomial outcomes $(1, \dots, p)$
<code>xexpend</code>	expenditure variables
<code>lnprices</code>	price array
<code>theta</code>	coefficients
<code>prob</code>	<code>n x p</code> array of choice probabilities

Warning

This routine is a utility routine that does **not** check the input arguments for proper dimensions and type.

Author(s)

Peter Rossi, Graduate School of Business, University of Chicago, Peter.Rossi@ChicagoGsb.edu.

References

For further discussion, see *Bayesian Statistics and Marketing* by Rossi, Allenby and McCulloch, Chapter 4.

<http://gsbwww.uchicago.edu/fac/peter.rossi/research/bsm.html>

See Also

[llnhlogit](#)

Index

- *Topic **array**
 - [createX](#), [12](#)
 - [nmat](#), [36](#)
- *Topic **datasets**
 - [bank](#), [3](#)
 - [cheese](#), [8](#)
 - [customerSat](#), [13](#)
 - [detailing](#), [14](#)
 - [margarine](#), [28](#)
 - [Scotch](#), [1](#)
- *Topic **distribution**
 - [breg](#), [5](#)
 - [condMom](#), [11](#)
 - [ghkvec](#), [18](#)
 - [lndIChisq](#), [23](#)
 - [lndIWishart](#), [24](#)
 - [lndMvn](#), [25](#)
 - [lndMvst](#), [26](#)
 - [logMargDenNR](#), [27](#)
 - [rbiNormGibbs](#), [38](#)
 - [rdirichlet](#), [40](#)
 - [rmixture](#), [55](#)
 - [rmvst](#), [64](#)
 - [rtrun](#), [74](#)
- *Topic **models**
 - [breg](#), [5](#)
 - [clusterMix](#), [9](#)
 - [eMixMargDen](#), [16](#)
 - [llmnl](#), [20](#)
 - [llmnp](#), [21](#)
 - [llnhlogit](#), [22](#)
 - [mixDen](#), [30](#)
 - [mixDenBi](#), [31](#)
 - [mnlHess](#), [33](#)
 - [mnpProb](#), [34](#)
 - [rbprobitGibbs](#), [39](#)
 - [rhierBinLogit](#), [41](#)
 - [rhierMnlRwMixture](#), [45](#)
 - [rhierNegbinRw](#), [49](#)
 - [rivGibbs](#), [52](#)
 - [rmnlIndepMetrop](#), [56](#)
 - [rmnpGibbs](#), [58](#)
 - [rmvpGibbs](#), [62](#)
 - [rnegbinRw](#), [65](#)
 - [rscaleUsage](#), [70](#)
 - [simmnl](#), [78](#)
 - [simmnlwX](#), [79](#)
 - [simmnp](#), [80](#)
 - [simmvp](#), [81](#)
 - [simnhlogit](#), [82](#)
- *Topic **multivariate**
 - [clusterMix](#), [9](#)
 - [eMixMargDen](#), [16](#)
 - [mixDen](#), [30](#)
 - [mixDenBi](#), [31](#)
 - [momMix](#), [35](#)
 - [rmixGibbs](#), [54](#)
 - [rmixture](#), [55](#)
 - [rmvpGibbs](#), [62](#)
 - [rnmixGibbs](#), [67](#)
 - [rwishart](#), [77](#)
- *Topic **regression**
 - [breg](#), [5](#)
 - [init.rmultiregfp](#), [19](#)
 - [rhierLinearModel](#), [43](#)
 - [rmultireg](#), [59](#)
 - [rmultiregfp](#), [61](#)
 - [rsurGibbs](#), [72](#)
 - [runireg](#), [75](#)
 - [runiregGibbs](#), [76](#)
- *Topic **ts**
 - [numEff](#), [37](#)
- *Topic **utilities**
 - [cgetC](#), [7](#)
 - [createX](#), [12](#)
 - [fsh](#), [17](#)
 - [init.rmultiregfp](#), [19](#)
 - [nmat](#), [36](#)
 - [numEff](#), [37](#)
- [bank](#), [3](#)
- [breg](#), [5](#)
- [cgetC](#), [7](#)
- [cheese](#), [8](#)
- [clusterMix](#), [9](#)
- [condMom](#), [11](#)

createX, 12, 20–22, 33, 34, 58
 customerSat, 13

 dchisq, 24
 detailing, 14

 eMixMargDen, 16, 69

 fsh, 17

 ghkvec, 18

 init.rmultiregfp, 19, 60, 62

 llmnl, 20, 33, 79
 llmnp, 21
 llnhlogit, 22, 83
 lndIChisq, 23
 lndIWishart, 24
 lndMvn, 25, 27
 lndMvst, 26, 26, 65
 logMargDenNR, 27

 margarine, 28
 mixDen, 30, 32, 69
 mixDenBi, 31, 69
 mnlHess, 33
 mnpProb, 34
 momMix, 35, 69

 nmat, 36
 numEff, 37

 rbiNormGibbs, 38
 rbprobitGibbs, 39
 rdirichlet, 40
 rhierBinLogit, 41
 rhierLinearModel, 43
 rhierMnlRwMixture, 12, 42, 45, 57
 rhierNegbinRw, 49, 66
 rivGibbs, 52
 rmixGibbs, 36, 54, 69
 rmixture, 55, 69
 rmnlIndepMetrop, 12, 13, 20, 33, 47, 56, 79
 rmnpGibbs, 12, 13, 22, 34, 39, 58, 63, 81
 rmultireg, 59, 62, 73
 rmultiregfp, 19, 60, 61
 rmvpGibbs, 12, 59, 62, 82
 rmvst, 64
 rneginRw, 51, 65
 rnmixGibbs, 10, 17, 31, 32, 55, 56, 67
 rscaleUsage, 7, 70
 rsurGibbs, 72

 rtrun, 74
 runireg, 75, 77
 runiregGibbs, 76, 76
 rwishart, 25, 77

 Scotch, 1
 simmnl, 78, 80
 simmnlwX, 79
 simmnp, 80
 simmvp, 81
 simnhlogit, 23, 82