Package 'Bolstad2'

May 6, 2026

Version 1.0-29

Date 2022-04-11

Title Bolstad Functions

Author James Curran <j.curran@auckland.ac.nz>

Maintainer James Curran <j.curran@auckland.ac.nz>

Description A set of R functions and data sets for the book ``Understanding Computational Bayesian Statistics." This book was written by Bill (WM) Bolstad and published in 2009 by John Wiley & Sons (ISBN 978-0470046098).

License GPL (>= 2)

Encoding UTF-8

URL https://github.com/jmcurran/Bolstad2

RoxygenNote 7.1.2

Imports graphics, stats

NeedsCompilation no

Repository CRAN

Date/Publication 2022-04-11 09:22:32 UTC

Contents

`AidsSurvival.df` *HIV Survival data*

Description

Data from a hypothetical HMO-HIV+ study shown in Table 1.1 of Hosmer, D.W. and Lemeshow, S. (1998) Applied Survival Analysis: Regression Modeling of Time to Event Data, John Wiley and Sons Inc., New York, NY

Format

A data frame with 100 observations on 7 variables.

[,1]	id	numeric	l Subject ID code
[,2]	entdate	date	Entry date (ddmmyr)
[,3]	enddate	date	Entry date (ddmmyr)
[,4]	time	numeric	Survival Time = days between Entry date and End date
[,5]	age	numeric	Age in years
[,6]	drug	factor	History of IV drug use (0 = No, 1 = Yes)
[,7]	censor	factor	Follow-Up Status1 = Death due to AIDS or AIDS related factors (0 = Alive at study end or lost to follow-up)

`BayesCPH` *Bayesian Cox Proportional Hazards Modelling*

Description

Uses a Metropolis Hastings scheme on the proportional hazards model to draw sample from posterior. Uses a matched curvature Student's t candidate generating distribution with 4 degrees of freedom to give heavy tails.

Usage

```
BayesCPH(
  y,
  t,
  x,
  steps = 1000,
  priorMean = NULL,
  priorVar = NULL,
  mleMean = NULL,
  mleVar,
  startValue = NULL,
  randomSeed = NULL,
  plots = FALSE
)
```

Arguments

y	the Poisson censored response vector. It has value 0 when the variable is censored and 1 when it is not censored.
t	time
x	matrix of covariates
steps	the number of steps to use in the Metropolis-Hastings updating
priorMean	the mean of the prior
priorVar	the variance of the prior
mleMean	the mean of the matched curvature likelihood
mleVar	the covariance matrix of the matched curvature likelihood
startValue	a vector of starting values for all of the regression coefficients including the intercept
randomSeed	a random seed to use for different chains
plots	Plot the time series and auto correlation functions for each of the model coefficients

Value

A list containing the following components:

beta	a data frame containing the sample of the model coefficients from the posterior distribution
mleMean	the mean of the matched curvature likelihood. This is useful if you've used a training set to estimate the value and wish to use it with another data set
mleVar	the covariance matrix of the matched curvature likelihood. See mleMean for why you'd want this

BayesLogistic *Bayesian Logistic Regression*

Description

Performas Metropolis Hastings on the logistic regression model to draw sample from posterior. Uses a matched curvature Student's t candidate generating distribution with 4 degrees of freedom to give heavy tails.

Usage

```
BayesLogistic(
  y,
  x,
  steps = 1000,
  priorMean = NULL,
  priorVar = NULL,
  mleMean = NULL,
  mleVar,
  startValue = NULL,
  randomSeed = NULL,
  plots = FALSE
)
```

Arguments

y	the binary response vector
x	matrix of covariates
steps	the number of steps to use in the Metropolis-Hastings updating
priorMean	the mean of the prior
priorVar	the variance of the prior
mleMean	the mean of the matched curvature likelihood
mleVar	the covariance matrix of the matched curvature likelihood
startValue	a vector of starting values for all of the regression coefficients including the intercept
randomSeed	a random seed to use for different chains
plots	Plot the time series and auto correlation functions for each of the model coefficients

Value

A list containing the following components:

beta	a data frame containing the sample of the model coefficients from the posterior distribution

mleMean	the mean of the matched curvature likelihood. This is useful if you've used a training set to estimate the value and wish to use it with another data set
mleVar	the covariance matrix of the matched curvature likelihood. See mleMean for why you'd want this

Examples

```
data(logisticTest.df)
BayesLogistic(logisticTest.df$y, logisticTest.df$x)
```

BayesPois *Bayesian Pois Regression*

Description

Performs Metropolis Hastings on the logistic regression model to draw sample from posterior. Uses a matched curvature Student's t candidate generating distribution with 4 degrees of freedom to give heavy tails.

Usage

```
BayesPois(
  y,
  x,
  steps = 1000,
  priorMean = NULL,
  priorVar = NULL,
  mleMean = NULL,
  mleVar,
  startValue = NULL,
  randomSeed = NULL,
  plots = FALSE
)
```

Arguments

y	the binary response vector
x	matrix of covariates
steps	the number of steps to use in the Metropolis-Hastings updating
priorMean	the mean of the prior
priorVar	the variance of the prior
mleMean	the mean of the matched curvature likelihood
mleVar	the covariance matrix of the matched curvature likelihood
startValue	a vector of starting values for all of the regression coefficients including the intercept

randomSeed	a random seed to use for different chains
plots	Plot the time series and auto correlation functions for each of the model coefficients

Value

A list containing the following components:

beta	a data frame containing the sample of the model coefficients from the posterior distribution
mleMean	the mean of the matched curvature likelihood. This is useful if you've used a training set to estimate the value and wish to use it with another data set
mleVar	the covariance matrix of the matched curvature likelihood. See mleMean for why you'd want this

Examples

```
data(poissonTest.df)
results = BayesPois(poissonTest.df$y, poissonTest.df$x)
```

---

bivnormMH *Metropolis Hastings sampling from a Bivariate Normal distribution*

Description

This function uses the MetropolisHastings algorithm to draw a sample from a correlated bivariate normal target density using a random walk candidate and an independent candidate density respectively where we are drawing both parameters in a single draw. It can also use the blockwise Metropolis Hastings algorithm and Gibbs sampling respectively to draw a sample from the correlated bivariate normal target.

Usage

```
bivnormMH(rho, rho1 = 0.9, sigma = c(1.2, 1.2), steps = 1000, type = "ind")
```

Arguments

rho	the correlation coefficient for the bivariate normal
rho1	the correlation of the candidate distribution. Only used when type = 'ind'
sigma	the standard deviations of the marginal distributions of the independent candidate density. Only used when type = 'ind'
steps	the number of Metropolis Hastings steps
type	the type of candidate generation to use. Can be one of 'rw' = random walk, 'ind' = independent normals, 'gibbs' = Gibbs sampling or 'block' = blockwise. It is sufficient to use 'r','i','g', or 'b'

Value

returns a list which contains a data frame called targetSample with members x and y. These are the samples from the target density.

Examples

```
## independent chain
chain1.df=bivnormMH(0.9)$targetSample

## random walk chain
chain2.df=bivnormMH(0.9, type = 'r')$targetSample


## blockwise MH chain
chain3.df=bivnormMH(0.9, type = 'b')$targetSample

## Gibbs sampling chain
chain4.df=bivnormMH(0.9, type = 'g')$targetSample

oldPar = par(mfrow=c(2,2))
plot(y ~ x, type = 'l', chain1.df, main = 'Independent')
plot(y ~ x, type = 'l', chain2.df, main = 'Random Walk')
plot(y ~ x, type = 'l', chain3.df, main = 'Blockwise')
plot(y ~ x, type = 'l', chain4.df, main = 'Gibbs')
par(oldPar)
```

---

c10ex16.df *Chapter 10 Example 16 data*

Description

A random sample of size 10 from a $N(\mu, \sigma^2)$ distribution where both mu and sigma are unknown parameters.

Format

A data frame with 10 observations in a single variable called y

chd.df — *Coronary Heart Disease Chapter 8 Example 11*

Description

The age and coronory heart disease status of 100 individuals taken from Hosmer and Lemeshow (1989).

Format

A data frame with 100 observations in two columns

[,1]	age	numeric	age in years
[,2]	chd	numeric factor	coronary heat disease status. Levels (1 = Yes), (0 = No)

credInt — *Calculate a credible interval from a numerically specified posterior CDF or from a sample from the posterior*

Description

Calculates a lower, upper, or two-sided credible interval from the numerical posterior CDF or from a sample from the posterior.

Usage

```
credInt(theta, cdf = NULL, conf = 0.95, type = "twosided")
```

Arguments

theta	either a sample from the posterior density or the values over which the the posterior CDF is specified
cdf	the values of the CDF, $F(\theta) = \int_{-\infty}^{\theta} f(t).df$ where $f(t)$ is the PDF. This only needs to be specified if a numerically specified posterior is being used
conf	the desired 'confidence' level
type	the type of interval to return, 'lower' = one sided lower bound, 'two-sided' = two - sided, or 'upper' = one sided upper bound. It is sufficient to use 'l','t' or 'u'

Details

This function uses linear interpolation to calculate bounds for points that may not be specified by CDF

Value

a list containing the elements lower.bound, uppper.bound or both depending on type

Examples

```
## commands for calculating a numerical posterior CDF.
## In this example, the likelihood is proportional to
## \eqn{\theta^{3/2}\times \exp(-\theta/4)} and a N(6, 9) prior is used.
theta = seq(from = 0.001, to = 40, by = 0.001)
prior = dnorm(theta,6,3)
ppnLike = theta^1.5*exp(-theta/4)
ppnPost = prior*ppnLike
scaleFactor = sintegral(theta, ppnPost)$int
posterior = ppnPost/scaleFactor
cdf = sintegral(theta, posterior)$y
ci = credInt(theta, cdf)
par(mfrow=c(2,2))
plot(prior ~ theta, type = 'l',  main = 'Prior N(6, 9)')
plot(ppnLike ~ theta, type = 'l', main = 'Proportional likelihood')
plot(posterior ~ theta, type = 'l', main = 'Posterior')
abline(v=c(unlist(ci)))

## Use an inverse method to take a random sample of size 1000
## from the posterior
suppressWarnings({Finv = approxfun(cdf,theta)})
thetaSample = Finv(runif(1000))
ci = credInt(thetaSample)
```

`credIntNum` *Calculate a credible interval from a numerically specified posterior CDF*

Description

Calculates a lower, upper, or two-sided credible interval from the numerical posterior CDF.

Usage

```
credIntNum(theta, cdf, conf = 0.95, type = "twosided")
```

Arguments

`theta`	the values over which the the posterior CDF is specified
`cdf`	the values of the CDF, $F(\theta) = \int_{-\infty}^{\theta} f(t).df$ where $f(t)$ is the PDF.
`conf`	the desired 'confidence' level
`type`	the type of interval to return, 'lower' = one sided lower bound, 'two-sided' = two - sided, or 'upper' = one sided upper bound. It is sufficient to use 'l','t' or 'u'

Details

This function uses linear interpolation to calculate bounds for points that may not be specified by CDF

Value

a list containing the elements lower.bound, uppper.bound or both depending on type

Examples

```
## commands for calculating a numerical posterior CDF.
## In this example, the likelihood is proportional to
## \eqn{\theta^{3/2}\times \exp(-\theta/4)} and a N(6, 9) prior is used.
theta = seq(from = 0.001, to = 40, by = 0.001)
prior = dnorm(theta,6,3)
ppnLike = theta^1.5*exp(-theta/4)
ppnPost = prior*ppnLike
scaleFactor = sintegral(theta, ppnPost)$int
posterior = ppnPost/scaleFactor
cdf = sintegral(theta, posterior)$y
ci=credIntNum(theta, cdf)
par(mfrow=c(2,2))
plot(prior ~ theta, type = 'l',  main = 'Prior N(6, 9)')
plot(ppnLike ~ theta, type = 'l', main = 'Proportional likelihood')
plot(posterior ~ theta, type = 'l', main = 'Posterior')
abline(v=c(unlist(ci)))
```

`credIntSamp` *Calculate a credible interval from a numerically specified posterior CDF*

Description

Calculates a lower, upper, or two-sided credible interval from the numerical posterior CDF.

Usage

```
credIntSamp(theta, conf = 0.95, type = "twosided")
```

Arguments

`theta`	a sample from the posterior density
`conf`	the desired 'confidence' level
`type`	the type of interval to return, 'lower' = one sided lower bound, 'two-sided' = two - sided, or 'upper' = one sided upper bound. It is sufficient to use 'l','t' or 'u'

Details

This function uses linear interpolation to calculate bounds for points that may not be specified by CDF

Value

a list containing the elements lower.bound, uppper.bound or both depending on type

Examples

```
## posterior is N(0,1)
theta = rnorm(1000)
ci=credIntSamp(theta)
plot(density(theta))
abline(v=c(unlist(ci)))
```

---

describe *Give simple descriptive statistics for a matrix or a data frame*

Description

This function is designed to emulate the Minitab function DESCRIBE. It gives simple descriptive statistics for a data frame

Usage

```
describe(x, varNames = NULL)
```

Arguments

x	A matrix or data.frame with numeric entries. Different variables are represented by columns.
varNames	A vector of variable names for each of the columns

Value

A data.frame containing the following elements:

N	The number of observations for each variable
mean	The sample mean for each variable
stdev	The sample standard deviation
sterr	The standard error of the mean
min	The minimum
q1	The lower quartile
med	The median
q3	The upper quartile
max	The maximum

Examples

```
data(poissonTest.df)
describe(poissonTest.df)
```

GelmanRubin *Calculate the Gelman Rubin statistic*

Description

Calculate the Gelman Rubin statistic

Usage

```
GelmanRubin(theta)
```

Arguments

theta	A matrix containing samples from at least two chains on a parameter theta. Each chain should 2n iterations. The last n iterations will be used to calculate the statistic

Value

A list containing n, the between chain variance B, the within chain variance W, the estimated variance of the parameter vHat, and the Gelman Rubin statistic $R = \sqrt{vHat/W}$

References

Gelman, A. and Rubin, D.B. (1992) 'Inference from iterative simulations using multiple sequences with discussion.' Statistical Science 8, pp. 457-511

Examples

```
## take four chains sampling from a normal mixture density
theta0 = c(0,1)
theta1 = c(3,2)
p = 0.6
candidate = c(0, 3)

v1 = normMixMH(theta0, theta1, p, candidate, steps = 200)
v2 = normMixMH(theta0, theta1, p, candidate, steps = 200)
v3 = normMixMH(theta0, theta1, p, candidate, steps = 200)
v4 = normMixMH(theta0, theta1, p, candidate, steps = 200)

theta=cbind(v1,v2,v3,v4)
GelmanRubin(theta)
```

`hierMeanReg` *Hierarchical Normal Means Regression Model*

Description

fits a hierarchical normal model of the form $E[y_{ij}] = \mu_j + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$

Usage

```
hierMeanReg(
  design,
  priorTau,
  priorPsi,
  priorVar,
  priorBeta = NULL,
  steps = 1000,
  startValue = NULL,
  randomSeed = NULL
)
```

Arguments

`design`	a list with elements y = response vector, group = grouping vector, x = matrix of covariates or NULL if there are no covariates
`priorTau`	a list with elements tau0 and v0
`priorPsi`	a list with elements psi0 and eta0
`priorVar`	a list with elements s0 and kappa0
`priorBeta`	a list with elements b0 and bMat or NULL if x is NULL
`steps`	the number of Gibbs sampling steps to take
`startValue`	a list with possible elements tau, psi, mu, sigmasq and beta. tau, psi and sigmasq must all be scalars. mu and beta must be vectors with as many elements as there are groups and covariates respectively
`randomSeed`	a random seed for the random number generator

Value

A data frame with variables:

`tau`	Samples from the posterior distribution of tau
`psi`	Samples from the posterior distribution of psi
`mu`	Samples from the posterior distribution of mu
`beta`	Samples from the posterior distribution of beta if there are any covariates
`sigmaSq`	Samples from the posterior distribution of σ^2
`sigma`	Samples from the posterior distribution of sigma

Examples

```
priorTau = list(tau0 = 0, v0 = 1000)
priorPsi = list(psi0 = 500, eta0 = 1)
priorVar = list(s0 = 500, kappa0 = 1)
priorBeta = list(b0 = c(0,0), bMat = matrix(c(1000,100,100,1000), ncol = 2))

data(hiermeanRegTest.df)
data.df = hiermeanRegTest.df
design = list(y = data.df$y, group = data.df$group,
              x = as.matrix(data.df[,3:4]))
r=hierMeanReg(design, priorTau, priorPsi, priorVar, priorBeta)

oldPar = par(mfrow = c(3,3))
plot(density(r$tau))
plot(density(r$psi))
plot(density(r$mu.1))
plot(density(r$mu.2))
plot(density(r$mu.3))
plot(density(r$beta.1))
plot(density(r$beta.2))
plot(density(r$sigmaSq))
par(oldPar)

## example with no covariates
priorTau = list(tau0 = 0, v0 = 1000)
priorPsi = list(psi0 = 500, eta0 = 1)
priorVar = list(s0 = 500, kappa0 = 1)

data(hiermeanRegTest.df)
data.df = hiermeanRegTest.df
design = list(y = data.df$y, group = data.df$group, x = NULL)
r=hierMeanReg(design, priorTau, priorPsi, priorVar)

oldPar = par(mfrow = c(3,2))
plot(density(r$tau))
plot(density(r$psi))
plot(density(r$mu.1))
plot(density(r$mu.2))
plot(density(r$mu.3))
plot(density(r$sigmaSq))
par(oldPar)
```

---

hiermeanRegTest.df *Test data for hiermeanReg*

Description

Data for testing hiermeanReg which uses Gibbs sampling on a hierarchical normal mean model with regression on covariates

Format

A data frame with 30 observations on 4 variables.

\ [1,]	y	numeric	the response vector
[2,]	group	factor	the grouping factor levels 1-3
[3,]	x1	numeric	the first covariate
[4,]	x2	numeric	the second covariate

See Also

hiermeanReg

---

logisticTest.df *Test data for bayesLogistic*

Description

A test data set for bayesLogisticReg

Format

A data frame with 100 observations on 6 variables.

[1,]	x	numeric	the covariate
[2,]	eps	numeric	the error in the response
[3,]	logit.p	numeric	the logit of the probability of success given x = 2 + 3*x + eps
[4,]	p	numeric	the probability of success given x
[5,]	u	numeric	a U[0,1] random variable
[6,]	y	binary	if u[i]<p[i] = 1, otherwise 0

See Also

bayesLogistic

`normGibbs` *Draw a sample from a posterior distribution of data with an unknown mean and variance using Gibbs sampling*

Description

normGibbs draws a Gibbs sample from the posterior distribution of the parameters given the data fron normal distribution with unknown mean and variance. The prior for μ given var is prior mean $m0$ and prior variance $var/n0$. That means $n0$ is the 'equivalent sample size.' The prior distribution of the variance is $s0$ times an inverse chi-squared with $kappa0$ degrees of freedom. The joint prior is the product $g(var)g(mu|var)$.

Usage

```
normGibbs(y, steps = 1000, type = "ind", ...)
```

Arguments

`y`	A vector containing the data
`steps`	The number of iterations of Gibbs sampling to carry out
`type`	Either 'ind' for sampling from an independent conjugate prior or 'joint' for sampling from a joint conjugate prior. 'i' and 'j' can be used as compact notation
`...`	If type = 'ind' then the user can specify the prior for μ with a parameter priorMu which can either be a single number m0, or m0 and n0. if m0 and n0 are not specified then m0 and n0 are 0 by default. The user can also specify priorVar, which if given, must be a vector with two elements s0 and kappa0. If s0 and kappa0 are not given then they are zero by default. If type = 'joint' then priorMu must be a vector of length two with elements m0 and sd0. The user can also specify priorVar, which if given, must be a vector with two elements s0 and kappa0. If s0 and kappa0 are not given then they are zero by default.

Value

A data frame containing three variables

[1,]	mu	a sample from the posterior distribution of the mean
[2,]	sig	a sample from the posterior distribution of the standard deviation
[3,]	mu	a sample from the posterior distribution of the variance = sig^2

Author(s)

James M. Curran

Examples

```
## firstly generate some random data
mu = rnorm(1)
sigma = rgamma(1,5,1)
y = rnorm(100, mu, sigma)

## A \eqn{N(10,3^2)} prior for \eqn{\mu} and a 25 times inverse chi-squared
## with one degree of freedom prior for \eqn{\sigma^2}
MCMCSampleInd = normGibbs(y, steps = 5000, priorMu = c(10,3),
                          priorVar = c(25,1))


## We can also use a joint conjugate prior for \eqn{\mu} and \eqn{\sigma^2}.
## This will be a \emph{normal}\eqn{(m,\sigma^2/n_0)} prior for \eqn{\mu} given
## the variance \eqn{\sigma^2}, and an \eqn{s0} times an \emph{inverse
## chi-squared} prior for \eqn{\sigma^2}.
MCMCSampleJoint = normGibbs(y, steps = 5000, type = 'joint',
                            priorMu = c(10,3), priorVar = c(25,1))

## Now plot the results
oldPar = par(mfrow=c(2,2))

plot(density(MCMCSampleInd$mu),xlab=expression(mu), main =
'Independent')
abline(v=mu)
plot(density(MCMCSampleInd$sig),xlab=expression(sig), main =
'Independent')
abline(v=sigma)

plot(density(MCMCSampleJoint$mu),xlab=expression(mu), main =
'Joint')
abline(v=mu)
plot(density(MCMCSampleJoint$sig),xlab=expression(sig), main =
'Joint')
abline(v=sigma)
```

`normMixMH` *Sample from a normal mixture model using Metropolis-Hastings*

Description

normMixMH uses the Metropolis-Hastings algorithm to draw a sample from a univariate target distribution that is a mixture of two normal distributions using an independent normal candidate density or a random walk normal candidate density.

Usage

```
normMixMH(
  theta0,
  theta1,
  p,
  candidate,
  steps = 1000,
  type = "ind",
  randomSeed = NULL,
  startValue = NULL
)
```

Arguments

theta0	A vector of length two containing the mean and standard deviation of the first component of the normal mixture
theta1	A vector of length two containing the mean and standard deviation of the second component of the normal mixture
p	A value between 0 and 1 representing the mixture proportion, so that the true density is $p \times f(\mu 1, \sigma 1) + (1 - p) \times f(\mu_2, \sigma_2)$
candidate	A vector of length two containing the mean and standard deviation of the candidate density
steps	The number of steps to be used in the Metropolis-Hastings algorithm. steps must be greater than 100
type	Either 'ind' or 'rw' depending on whether a independent candidate density or random walk candidate density is to be used. 'i' and 'r' may be used as alternative compact notation
randomSeed	A seed for the random number generator. Only used when you want the same sequence of random numbers in the chain
startValue	A starting value for the chain

Value

A vector containing a sample from the normal mixture distribution.

Examples

```
## Set up the normal mixture
theta0 = c(0,1)
theta1 = c(3,2)
p = 0.8

## Sample from an independent N(0,3^2) candidate density
candidate = c(0, 3)
MCMCsampleInd = normMixMH(theta0, theta1, p, candidate)
```

```
## If we wish to use the alternative random walk N(0, 0.5^2)
## candidate density
candidate = c(0, 0.5)
MCMCsampleRW = normMixMH(theta0, theta1, p, candidate, type = 'rw')
```

---

pNull *Test a one sided hypothesis from a numerically specified posterior CDF or from a sample from the posterior*

Description

Calculates the probability of a one sided null hypothesis from a numerically calculated posterior CDF or from a sample from the posterior.

Usage

```
pNull(theta0, theta, cdf = NULL, type = "upper")
```

Arguments

theta0	the hypothesized value, i.e. H0: theta <= theta0
theta	a sample of values from the posterior density, or, if cdf is not NULL then the values over which the the posterior CDF is specified
cdf	the values of the CDF, $F(\theta) = \int_{-\infty}^{\theta} f(t).df$ where $f(t)$ is the PDF.
type	the type of probability to return, 'lower' = Pr(theta <= theta0) or 'upper' = Pr(theta >= theta0). It is sufficient to use 'l' or 'u'

Details

This function uses linear interpolation to calculate bounds for points that may not be specified by CDF

Value

a list containing the element prob which will be the upper or lower tail probability depending on type

Examples

```
## commands for calculating a numerical posterior CDF.
## In this example, the likelihood is proportional to
## \eqn{\theta^{3/2}\times \exp(-\theta/4)} and a N(6, 9) prior is used.
theta = seq(from = 0.001, to = 40, by = 0.001)
prior = dnorm(theta,6,3)
ppnLike = theta^1.5*exp(-theta/4)
ppnPost = prior*ppnLike
scaleFactor = sintegral(theta, ppnPost)$int
```

```
posterior = ppnPost/scaleFactor
cdf = sintegral(theta, posterior)$y
pNull(15, theta, cdf)

## Use an inverse method to take a random sample of size 1000
## from the posterior
suppressWarnings({Finv = approxfun(cdf, theta)})
thetaSample = Finv(runif(1000))
pNull(15, thetaSample)
```

`pnullNum` *Test a one sided hypothesis from a numerically specified posterior CDF*

Description

Calculates the probability of a one sided null hypothesis from a numerically calculated posterior CDF.

Usage

```
pnullNum(theta0, theta, cdf, type = "upper")
```

Arguments

`theta0`	the hypothesized value, i.e. H0: theta <= theta0
`theta`	the values over which the the posterior CDF is specified
`cdf`	the values of the CDF, $F(\theta) = \int_{-\infty}^{\theta} f(t).df$ where $f(t)$ is the PDF.
`type`	the type of probability to return, 'lower' = Pr(theta <= theta0) or 'upper' = Pr(theta >= theta0). It is sufficient to use 'l' or 'u'

Details

This function uses linear interpolation to calculate bounds for points that may not be specified by CDF

Value

a list containing the element prob which will be the upper or lower tail probability depending on type

Examples

```
## commands for calculating a numerical posterior CDF.
## In this example, the likelihood is proportional to
## \eqn{\theta^{3/2}\times \exp(-\theta/4)} and a N(6, 9) prior is used.
theta = seq(from = 0.001, to = 40, by = 0.001)
prior = dnorm(theta,6,3)
ppnLike = theta^1.5*exp(-theta/4)
ppnPost = prior*ppnLike
scaleFactor = sintegral(theta, ppnPost)$int
posterior = ppnPost/scaleFactor
cdf = sintegral(theta, posterior)$y
pnullNum(1, theta, cdf)
```

---

pnullSamp *Test a one sided hypothesis using a sample from a posterior density*

Description

Calculates the probability of a one sided null hypothesis from a sample from a posterior density.

Usage

```
pnullSamp(theta, theta0 = 0, type = "upper")
```

Arguments

theta	a sample of values from a posterior density
theta0	the hypothesized value, i.e. H0: theta <= theta0
type	the type of probability to return, 'lower' = Pr(theta <= theta0) or 'upper' = Pr(theta >= theta0). It is sufficient to use 'l' or 'u'

Details

This function uses linear interpolation to calculate bounds for points that may not be specified by CDF

Value

a list containing the element prob which will be the upper or lower tail probability depending on type

Examples

```
## The posterior density is N(3,1)
theta = rnorm(1000,3)

## test whether the true mean is greater than 0 (it is obviously!)
pnullSamp(theta)
```

poissonTest.df *A test data set for bayesPois*

Description

A test data set for bayesPois. The data come from the equation $\log(\lambda_i) = 1 + 5x_i + \epsilon_i$ where ϵ_i comes from N(0,0.01).

Format

A data frame with 100 observations on 5 variables.

[1,]	x	numeric	the covariate
[2,]	eps	numeric	the error in the log response
[3,]	log.lam	numeric	$\log(\lambda_i) = 1 + 5x_i + \epsilon_i$ where ϵ_i
[4,]	lam	numeric	$\exp(\log(\lambda))$
[5,]	y	numeric	a Poisson random variate with mean λ_i

See Also

bayesPois

sintegral *Numerical integration using Simpson's Rule*

Description

Takes a vector of x values and a corresponding set of postive $f(x) = y$ values and evaluates the area under the curve:

$$\int f(x)dx$$

.

Usage

```
sintegral(x, fx, n.pts = 256)
```

Arguments

x	a sequence of x values.
fx	the value of the function to be integrated at x.
n.pts	the number of points to be used in the integration.

Value

returns a list with the following elements

x	the x-values at which the integral has been evaluated
y	the cummulative integral
int	the value of the integral over the whole range

Examples

```
## integrate the normal density from -3 to 3
x=seq(-3,3,length=100)
fx=dnorm(x)
estimate=sintegral(x,fx)$int
true.val=diff(pnorm(c(-3,3)))
cat(paste('Absolute error :',round(abs(estimate-true.val),7),'\n'))
cat(paste('Relative percentage error :', 100*round((abs(estimate-true.val)/true.val),6),'%\n'))
```

---

thin *Thin an MCMC sample*

Description

Thins the output from an MCMC process

Usage

```
thin(x, k)
```

Arguments

x	A vector, matrix or data.frame containing output from an MCMC sampling scheme
k	An integer. This function takes every kth element from x

Details

Note this function does not check to see if k is sensible.

Value

A thinned vector, matrix or data frame containing every kth element of x.

Examples

```
## A blockwise Metropolis-Hastings chain of 1000 elements, thinned to
## 5th element
##

MCMCSampleBW = bivnormMH(0.9, type = 'block')
MCMCSampleBW = thin(MCMCSampleBW, 5)
```

Index