

# Package ‘MitISEM’

February 19, 2015

**Type** Package

**Title** Mixture of Student t Distributions using Importance Sampling and Expectation Maximization

**Version** 1.0

**Date** 2012-09-15

**Description** Flexible multivariate function approximation using adapted Mixture of Student t Distributions. Mixture of t distribution is obtained using Importance Sampling weighted Expectation Maximization algorithm.

**Author** N. Basturk, L.F. Hoogerheide, A. Opschoor, H.K. van Dijk

**Maintainer** N. Basturk <basturk@ese.eur.nl>

**License** GPL (>= 3)

**LazyLoad** yes

**Depends** mvtnorm

**Suggests** AdMit

**URL** <http://people.few.eur.nl/basturk/>

**Repository** CRAN

**Date/Publication** 2012-09-18 18:43:34

**NeedsCompilation** no

## R topics documented:

MitISEM-package . . . . .	2
MargLik . . . . .	2
Mit . . . . .	4
MitISEM . . . . .	5
Mvgt . . . . .	7
PredLik . . . . .	8
SeqMitISEM . . . . .	10
<b>Index</b>	<b>12</b>

---

MitISEM-package	<i>Mixture of Student t Distributions using Importance Sampling and Expectation Maximization</i>
-----------------	--

---

### Description

Approximates a univariate or multivariate density using mixture of Student  $t$  distributions, achieved by Importance Sampling and Expectation Maximization algorithms

### Details

Package: MitISEM  
 Type: Package  
 Version: 1.0  
 Date: 2012-07-31  
 License: What license is it under?

~~ An overview of how to use the package, including the most important functions ~~

### Author(s)

N. Basturk, L.F. Hoogerheide, A. Opschoor, H.K. van Dijk

### References

'A Class of Adaptive EM-based Importance Sampling Algorithms for Efficient and Robust Posterior and Predictive Simulation', 2012., by Lennart Hoogerheide and Anne Opschoor and Herman K. van Dijk", Tinbergen Institute Discussion Paper, no. 11-004/4

### Examples

```
print("to be completed")
```

---

MargLik	<i>Marginal Likelihood calculation using Importance Sampling and mixture of Student-t densities as candidate</i>
---------	--

---

### Description

Calculation of marginal likelihoods using Importance Sampling, with a Mixture of Student- $t$  candidate density. Calculated marginal likelihoods from two data samples can be used to get predictive likelihoods using [PredLik](#).

**Usage**

```
MargLik(N=1e4,mit,KERNEL,...)
```

**Arguments**

N	integer > 100 number of draws for Importance Sampling
mit	Mixture of Student- <i>t</i> density for the full sample, list describing the mixture of Student- <i>t</i> . See <a href="#">isMit</a> . The mixture density can be obtained from <a href="#">MitISEM</a> or <a href="#">SeqMitISEM</a>
KERNEL	Posterior kernel to be approximated. See *Details*. A log argument must exist. The function must return log-density if log=TRUE.
...	other arguments to be passed to KERNEL

**Details**

If MargLik is used to get the Marginal Likelihood of a single model, KERNEL must be the exact posterior density (including the scaling constant) of parameters.

If MargLik is used as an intermediate step, for instance for calculating predictive likelihoods, KERNEL can be a posterior kernel or the exact posterior density of parameters. See [PredLik](#).

**Value**

list containing:

ML.mean	Marginal likelihood (posterior mean) $\times 10^{scale}$
ML.NSE	Numerical Standard Error for mean Marginal likelihood $\times 10^{scale}$
scale	integer > 0 providing the scaling for predictive likelihood. (scaling may be necessary for numerical accuracy)

**See Also**

[isMit](#), [PredLik](#), [MitISEM](#), [SeqMitISEM](#)

**Examples**

```
mit <- list(p=1,mu=matrix(1),Sigma=matrix(0.1),df=5)
data <- rnorm(100,1)
KERNEL <- function(theta,data,log=TRUE){
  if(is.vector(theta))
    theta = matrix(theta,nrow=1)
  r <- apply(theta,1,function(x,data)(sum(dnorm(data,x,log=log))),data=data)
}
MargLik(N=1000,mit=mit,KERNEL=KERNEL,data=data)
```

Mit

*The 'mit' object***Description**

Function to check if `mit` generalized mixture of t densities is well-defined. The `mit` object is designed to be used in the rest of the `MitISEM` package functions

**Usage**

```
isMit(mit)
```

**Arguments**

`mit` an object to be tested

**Details**

Argument `mit` is a list describing the mixture of Student-t distributions with the following components:

`p` vector (of length  $H$ ) of mixture probabilities.

`mu` matrix (of size  $H \times d$ ) containing the vectors of modes (in row) of the mixture components.

`Sigma` matrix (of size  $H \times d^2$ ) containing the scale matrices (in row) of the mixture components.

`df` vector (of length  $H$ ) degree of freedom parameters for each Student-t component (double  $> 0$ ).

**Value**

logical, TRUE if `mit` definition is correct, FALSE otherwise

**Examples**

```
# a correct Mit definition returns 'TRUE'
H <- 2
p <- runif(H)
p = p / sum(p)
mu <- matrix(seq(1:H),H,1)
Sigma <- matrix(runif(H^2),H,H)
df <- seq(1:H)
isMit(mit=list(p=p,mu=mu,Sigma=Sigma,df=df))

# an incorrect Mit definition returns 'FALSE'
mu = t(mu)
isMit(mit=list(p=p,mu=mu,Sigma=Sigma,df=df))
```

---

MitISEM	<i>Mixture of Student-t distributions using Importance Sampling weighted Expectation Maximization steps</i>
---------	---

---

### Description

Approximates a  $k$  dimensional function/kernel by a mixture of student- $t$  distributions using Importance Sampling weighted Expectation Maximization steps.

### Usage

```
MitISEM(KERNEL, mu0, Sigma0=NULL, df0=1, mit0=NULL, control=list(), ...)
```

### Arguments

KERNEL	Function to be approximated. First argument should be the parameter matrix. A log argument should exist such that the function returns log-density if log=TRUE
mu0	vector of length $k$ , starting points for approximation
Sigma0	(optional) $k \times k$ dimensional positive definite symmetric initial scale matrix. Default: matrix is obtained by the BFGS algorithm.
df0	(optional) double $> 0$ initial degree of freedom of the Student-t density. Default: $df0 = 1$ .
mit0	(optional) initial mixture density defined. See *Details*.
control	(optional) list of control parameters for IS and EM optimization and stopping rule of the H component mixture of t densities. See *Details*
...	other arguments to be passed to KERNEL

### Details

Providing mit0 argument makes arguments mu0, Sigma0 and df0 obsolete. Argument mit0 (if provided) should include the following components (see [isMit](#)):

p vector (of length  $H$ ) of mixture probabilities.

mu matrix (of size  $H \times k$ ) containing the vectors of modes (in row) of the mixture components.

Sigma matrix (of size  $H \times k^2$ ) containing the scale matrices (in row) of the mixture components.

df vector (of length  $H$ ) degree of freedom parameters of the Student-t components. Each element should be above 0

Optional argument control can provide several optimization parameters:

N integer ( $>100$ ) number of draws used in the simulation. Default:  $N=1e5$ .

robust.N logical indicating if robust draws are used if robust.N=TRUE (default), simulations are repeated to get N draws with finite KERNEL values.

Hmax integer  $> 0$  maximum number of components. Default:  $H=10$ .

- `StopMethod` string, CV (default) or AR defining type of stopping criteria for MitISEM approximation. CV method stops the algorithm if the coefficient of variation in IS weights converges. AR method stops the algorithm if the (expected) acceptance rate given the current MitISEM approximation and function KERNEL converges.
- `CVtol` double in  $(0, 1)$  convergence criteria for CV method. Higher values lead to faster convergence but worse approximation. Default:  $CVtol = 0.1$ , algorithm stops if `StopMethod = CV` and the change in coefficient of variation is below 10%.
- `ARtol` double in  $(0, 1)$  convergence criteria similar to `CVtol`, used if `StopMethod = AR`. Default:  $ARtol = 0.1$ .
- `trace` logical to print partial output. Default:  $trace = FALSE$ , no tracing information.
- `trace.init` logical to print output of the first student-t optimization. Default:  $trace = FALSE$ , no tracing information.
- `maxit.init` double, maximum number of iterations in the first student-t optimization. Default:  $maxit.init = 1e4$ .
- `reltol.init` double, relative tolerance in the first student-t optimization. Default:  $reltol.init = 1e - 8$ .
- `maxit.EM` integer  $> 0$ , maximum number of iterations for the EM algorithm. Default:  $maxit.EM = 1000$ .
- `tol.EM` double  $> 0$ , tolerance for EM steps' convergence, Default:  $tol.EM = 0.001$ .
- `trace.EM` logical to print partial output during the IS-EM algorithm. Default:  $trace.EM = FALSE$ , no tracing information.
- `optim.df` logical to optimize degrees of freedom of the Student-t components. Default:  $optim.df = TRUE$  df are optimized. Note: Keeping degrees of freedom in low values may be desired if the approximation is used in a rejection sampling. If  $optim.df = FALSE$ , degree of freedom of all student t components are fixed at  $df_0$ .
- `inter.df` increasing vector of length 2 range of search values for df optimization, active if  $optim.df = TRUE$ . Default:  $inter.df = (0.01, 30)$ .
- `tol.df` double  $> 0$ , tolerance for degree of freedom optimization, active if  $optim.df = TRUE$ . Default:  $tol.df = 0.0001$
- `maxit.df` integer  $> 0$  maximum number of iterations for degree of freedom optimization, active if  $optim.df = TRUE$ . Default:  $maxit.df = 1e3$ .
- `trace.df` logical to print partial output during degree of freedom optimization, active if  $optim.df = TRUE$ . Default:  $trace.df = FALSE$ .
- `tol.pr` double in  $[0,1)$ , minimum probability required to keep mixture components. Default:  $tol.pr = 0$  all mixture components are kept.
- `ISpc` double in  $(0,1)$ , fraction of draws to construct new component. Default:  $ISpc = 0.1$ .
- `Pnc` double in  $(0,1)$ , initial probability of the new component. Default:  $Pnc = 0.1$ .

## References

Hoogerheide, L. and Opschoor, A. and Van Dijk, H. K. (2012) A Class of Adaptive Importance Sampling Weighted EM Algorithms for Efficient and Robust Posterior and Predictive Simulation. *Journal of Econometrics*, in press. <http://www.sciencedirect.com/science/article/pii/S0304407612001583>

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

```

require(graphics)
set.seed(1234);
# define Gelman Meng Kernel
GelmanMeng <- function(x, A = 1, B = 0, C1 = 3, C2 = 3, log = TRUE){
  if (is.vector(x))
    x <- matrix(x, nrow = 1)
  r <- -.5 * (A * x[,1]^2 * x[,2]^2 + x[,1]^2 + x[,2]^2
             - 2 * B * x[,1] * x[,2] - 2 * C1 * x[,1] - 2 * C2 * x[,2])
  if (!log)
    r <- exp(r)
  as.vector(r)
}
# get MitISEM approximation
mu0 <-c(3,4)
app.MitISEM <- MitISEM(KERNEL=GelmanMeng,mu0=mu0,control=list(N=2000,trace=TRUE))
mit=app.MitISEM$mit

# plot approximation (components and full approximation)
MitISEM.plot.comps <- function(mit,x1,x2,log=FALSE){
  Mitcontour <- function(x1,x2,mit,log=FALSE){
    dmvgt(cbind(x1,x2),mit=mit,log=log)
  }
  H <- length(mit$p)
  K <- ncol(mit$mu)
  cols <- 1:H
  for (h in 1:H){
    mit.h <-list(p=1,mu=matrix(mit$mu[h,],1,K),
                Sigma=matrix(mit$Sigma[h,],1,(K^2)),df=mit$df[h])
    z <- outer(x1,x2,FUN=Mitcontour,mit=mit.h)
    contour(x1,x2,z,col=h,lty=h,labels="",add=(h!=1),
            xlab="x1",ylab="x2",main='MitISEM approximation components')
  }
  legend("topright",paste("component ",1:H),lty=cols,col=cols,bty='n')
  z <- outer(x1,x2, FUN=Mitcontour,mit=mit)
  image(x1,x2,z,las=1,col=gray((20:0)/20),main='MitISEM approximation')
}
x1 <- seq(-2,6,0.05)
x2 <- seq(-2,7,0.05)
MitISEM.plot.comps(mit,x1,x2)

```

**Description**

Density and random generation for the general student t distribution

**Usage**

```
dmvgt(theta, mit = list(), log=TRUE)
rmvgt(N,mit)
```

**Arguments**

theta	Vector of length $N$ or $N \times k$ matrix of quantiles. If theta is a matrix, each row is taken to be a quantile.
N	number of observations
mit	list defining the mixture components. See <a href="#">isMit</a> for how it should be defined.
log	logical; if TRUE (default), probabilities $p$ are given as $\ln(p)$ .

**Value**

dmvgt returns vector of size  $N$  with density values for each row of theta

rmvgt returns an  $N \times k$  matrix of draws from the  $k$ -variate mixture of student t densities

**See Also**

[isMit](#)

**Examples**

```
H      <- 2
p      <- runif(H)
p      = p / sum(p)
mu     <- matrix(seq(1:H),H,1)
Sigma  <- matrix(runif(H^2),H,H)
df     <- seq(1:H)
Ndraws <- rmvgt(N=10,mit=list(p=p,mu=mu,Sigma=Sigma,df=df))
pdraws <- dmvgt(theta=Ndraws,mit=list(p=p,mu=mu,Sigma=Sigma,df=df))
```

---

PredLik

*Predictive Likelihood calculation using Importance Sampling and mixture of Student-t densities as candidate*

---

**Description**

Calculation of predictive likelihoods using Importance Sampling, given subsample and full data sample and Mixture of Student- $t$  candidate density. Predictive likelihood is calculated using the marginal likelihood from full sample and subsample. See [MargLik](#).



**Usage**

```
PredLik(N=1e4,mit.fs,mit.ss,KERNEL,data.fs,data.ss,...)
```

**Arguments**

N	integer > 100 number of draws for Importance Sampling
mit.fs	Mixture of Student- <i>t</i> density for the full sample, list describing the mixture of Student- <i>t</i> . See <a href="#">isMit</a> . The mixture density can be obtained from <a href="#">MitISEM</a> or <a href="#">SeqMitISEM</a>
mit.ss	Mixture of Student- <i>t</i> density for subsample. Must be defined as <code>mit.fs</code> .
KERNEL	Function/posterior to be approximated. <code>data</code> and <code>log</code> arguments must exist. The function must return log-density if <code>log=TRUE</code> . All data should be loaded in argument <code>data</code>
data.fs	Full data, vector (length $T1$ ) or matrix (size $T1 \times m$ ) with data values, $T1$ observations and $m$ data series.
data.ss	Sample of data, vector (length $T2$ ) or matrix (size $T2 \times m$ ) with data values, $T2$ observations and $m$ data series. $T2 < T1$ .
...	other arguments to be passed to KERNEL

**Details**

Argument KERNEL

**Value**

list containing:

PL	Predictive likelihood $\times 10^{scale}$
scale	integer > 0 providing the scaling for predictive likelihood. (scaling may be necessary for numerical accuracy)

**References**

Eklund, J. and Karlsson, S. (2007). Forecast combination and model averaging using predictive measures. *Econometric Reviews*, 26, 329-363.

Min, C. and Zellner, A. (1993). Bayesian and non-Bayesian methods for combining models and forecasts with applications to forecasting international growth rates. *Journal of Econometrics*, 56, 89-118.

**See Also**

[isMit](#), [MargLik](#), [MitISEM](#), [SeqMitISEM](#)

---

SeqMitISEM	<i>Sequential approximation using Mixture of Student-t distributions using Importance Sampling weighted Expectation Maximization steps</i>
------------	--

---

### Description

Approximates a  $k$  dimensional function/kernel using mixture of student- $t$  distributions for the initial data sample and updated data samples sequentially

### Usage

```
SeqMitISEM(data,KERNEL,mu0,Sigma0=NULL,df0=1,control.MitISEM=list(),control.seq=list(),...)
```

### Arguments

data	matrix (size $T \times m$ ) with data values, $T$ observations and $m$ data series
KERNEL	Function/posterior to be approximated. data and log arguments should exist. The function must return log-density if log=TRUE. All data should be loaded in argument data
mu0	vector of length $k$ starting points. They should be defined as in <a href="#">MitISEM</a>
Sigma0,df0	(optional) initial scale and degrees of freedom for the student t density. They should be defined as in <a href="#">MitISEM</a>
control.MitISEM	(optional) control parameters passed to MitISEM. See <a href="#">MitISEM</a> .
control.seq	control parameters for sequential updating of the MitISEM approximation. See <i>*Details*</i> .
...	other arguments to be passed to KERNEL.

### Details

The optional argument control.seq can provide several optimization parameters:

`T0` integer ( $< T$ ) number of observations. Default: `round(T/2)`.

`tau` vector of length  $t$  with iterative number of observations to add to the sample. Its elements should be positive integers, and  $T0 + \max(\text{tau}) \leq T$  should hold. Default: `tau=1`, one single observation is added to the sample for Sequential MitISEM.

`tol.seq` double in  $(0, 1)$  convergence criteria for sequential Coefficient of Variation convergence. Default: `tol.seq=0.2`.

`method` 0, 1, 2 method to select initial data sample. if `method=0` initial sample is randomly selected. if `method=1` first `T0` observations are taken as initial sample (default). if `method=2` last `T0` observations are taken as initial sample.

`trace` logical to print partial output. Default: `trace = FALSE`, no tracing information.

**References**

Hoogerheide, L. and Opschoor, A. and Van Dijk, H. K. (2012). A Class of Adaptive Importance Sampling Weighted EM Algorithms for Efficient and Robust Posterior and Predictive Simulation. *Journal of Econometrics*, in press. <http://www.sciencedirect.com/science/article/pii/S0304407612001583>

**See Also**

[MitISEM](#)

# Index

## \*Topic **package**

MitISEM-package, [2](#)

dmvgt (Mvgt), [7](#)

isMit, [3](#), [5](#), [7–9](#)

isMit (Mit), [4](#)

MargLik, [2](#), [8](#), [9](#)

Mit, [4](#)

MitISEM, [3](#), [5](#), [9–11](#)

MitISEM-package, [2](#)

Mvgt, [7](#)

PredLik, [2](#), [3](#), [8](#)

rmvgt (Mvgt), [7](#)

SeqMitISEM, [3](#), [9](#), [10](#)