

# Package ‘moewishart’

February 19, 2026

**Title** Mixture-of-Experts Wishart Models for Covariance Data

**Version** 1.0

**Description** Methods for maximum likelihood and Bayesian estimation for the Wishart mixture model and the mixture-of-experts Wishart (MoE-Wishart) model. The package provides four inference algorithms for these models, each implemented using the expectation-maximization (EM) algorithm for maximum likelihood estimation and a fully Bayesian approach via Gibbs-within-Metropolis-Hastings sampling.

**Maintainer** Zhi Zhao <zhi.zhao@medisin.uio.no>

**URL** <https://github.com/zhizuio/moewishart>

**BugReports** <https://github.com/zhizuio/moewishart/issues>

**License** GPL-3

**VignetteBuilder** knitr

**Depends** R (>= 4.1.0)

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**Imports** utils, stats, loo

**Suggests** rmarkdown, knitr

**NeedsCompilation** no

**Author** The Tien Mai [aut],  
Zhi Zhao [aut, cre]

**Repository** CRAN

**Date/Publication** 2026-02-19 19:50:02 UTC

## Contents

computeIC . . . . .	2
dWishart . . . . .	3
lmygamma . . . . .	5
mixturewishart . . . . .	6
moewishart . . . . .	9

rdirichlet . . . . .	12
sampleIW . . . . .	13
simData . . . . .	14

<b>Index</b>	<b>16</b>
--------------	-----------

---

computeIC	<i>Information criteria for Wishart mixtures and MoE models</i>
-----------	---

---

## Description

Compute AIC, BIC, and ICL for EM fits; and PSIS-LOO expected log predictive density (elpd\_loo) for Bayesian fits. Supports mixturewishart (finite mixture) and moewishart (MoE with covariates in gating).

## Usage

```
computeIC(fit)
```

## Arguments

`fit` A fitted object returned by `mixturewishart()` or `moewishart()`.

## Details

For EM fits:

- AIC:  $AIC = 2k - 2\ell$ , where  $k$  is the number of free parameters and  $\ell$  is the maximized log-likelihood (last EM iteration).
- BIC:  $BIC = k \log n - 2\ell$ .
- ICL:  $ICL = BIC + \sum_{i=1}^n \sum_{k=1}^K \tau_{ik} \log \tau_{ik}$ , i.e., BIC plus the entropy term (classification likelihood approximation).

Parameter counting  $k$ :

- For `mixturewishart`:  $k = (K - 1) + K \cdot \frac{p(p+1)}{2} + K \cdot \mathbb{I}[\text{estimate } \nu]$ , where  $(K - 1)$  are mixture weights, each  $\Sigma_k$  has  $\frac{p(p+1)}{2}$  free parameters, and  $\nu_k$  adds 1 per component when estimated.
- For `moewishart`:  $k = q(K - 1) + K \cdot \frac{p(p+1)}{2} + K \cdot \mathbb{I}[\text{estimate } \nu]$ , where  $q(K - 1)$  are gating regression coefficients (last class is reference with zero column).

For Bayesian fits:

- `elpd_loo`: computed via `loo::loo` using per-observation log-likelihood draws. Requires `fit$loglik_individual` as a matrix of size  $S \times n$  (draws by observations), as produced by the provided samplers. Returns `elpd_loo`, `se_elpd_loo`, `p_loo`, and `looic`.

Notes:

- AIC/BIC/ICL are defined for MLE (EM) fits. For Bayesian fits, prefer elpd\_loo (or WAIC) rather than AIC/BIC.
- The entropy term in ICL uses EM responsibilities  $\tau_{ik}$  (field tau for mixturewishart, gamma for moewishart).

### Value

- For method="em": a list with fields AIC, BIC, ICL. - For method="bayes": a list with fields ICL and elpd of class "loo" as returned by [loo::loo()] that contains fields 'estimates', 'pointwise', 'diagnostics'

### Examples

```
# Bayesian example (MoE)

# simulate data
set.seed(123)
n <- 500 # subjects
p <- 2
# True gating coefficients (last column zero)
set.seed(123)
Xq <- 3
K <- 3
betas <- matrix(runif(Xq * K, -2, 2), nrow = Xq, ncol = K)
betas[, K] <- 0
dat <- simData(n, p,
  Xq = 3, K = 3, betas = betas,
  pis = c(0.35, 0.40, 0.25),
  nus = c(8, 16, 3)
)
set.seed(123)
fit <- moewishart(
  dat$$,
  X = cbind(1, dat$X), K = 3,
  mh_sigma = c(0.2, 0.1, 0.1), # RW-MH variances (length K)
  mh_beta = c(0.2, 0.2), # RW-MH variances (length K-1)
  niter = 100, burnin = 50
)
computeIC(fit)
```

---

dWishart

*density of Wishart distribution*


---

### Description

Compute the (log) density of a  $p$ -dimensional Wishart distribution  $W_p(\nu, \Sigma)$  at an SPD matrix  $S$ . Returns either the log-density or the density depending on `logarithm`.

**Usage**

```
dWishart(S, nu, Sigma, detS_val = NULL, logarithm = TRUE)
```

**Arguments**

**S** Numeric  $p \times p$  SPD matrix at which to evaluate the density.

**nu** Numeric. Degrees of freedom  $\nu$  (must exceed  $p - 1$ ).

**Sigma** Numeric  $p \times p$  SPD scale matrix  $\Sigma$ .

**detS\_val** Optional numeric. Precomputed  $\log |S|$  to reuse; if NULL, it is computed internally.

**logarithm** Logical. If TRUE, return log-density; otherwise return density.

**Details**

Let  $S \sim W_p(\nu, \Sigma)$  with degrees of freedom  $\nu$  and scale matrix  $\Sigma$  (SPD). The density is:

$$f(S \mid \nu, \Sigma) = \frac{|S|^{(\nu-p-1)/2} \exp\{-\frac{1}{2} \text{tr}(\Sigma^{-1}S)\}}{2^{\nu p/2} |\Sigma|^{\nu/2} \Gamma_p(\nu/2)},$$

where  $\Gamma_p(\cdot)$  is the multivariate gamma function and  $p$  is the dimension.

Note that (i) `detS_val` can be supplied to avoid recomputing  $\log |S|$ , which is useful inside EM/MCMC loops, and (ii) small diagonal jitter is added internally to  $S$  and  $\Sigma$  when computing determinants or solves for numerical stability.

Constraints: (i)  $S$  and  $\Sigma$  must be SPD, and (ii) the Wishart requires  $\nu > p - 1$ .

**Value**

A numeric scalar: the log-density if `logarithm = TRUE`, otherwise the density.

**Examples**

```
set.seed(123)
p <- 3
# Construct an SPD Sigma
A <- matrix(rnorm(p * p), p, p)
Sigma <- crossprod(A) + diag(p) * 0.5
# Draw a Wishart matrix using base stats::rWishart()
W <- drop(rWishart(1, df = p + 5, Sigma = Sigma))
# Evaluate log-density at W
dWishart(W, nu = p + 5, Sigma = Sigma, logarithm = TRUE)
```

---

lmgamma	<i>Log multivariate gamma function</i>
---------	--

---

### Description

Compute the log of the multivariate gamma function  $\log \Gamma_p(a)$  for dimension  $p$  and parameter  $a$ .

### Usage

```
lmgamma(a, p)
```

### Arguments

**a** Numeric. Argument of  $\Gamma_p(\cdot)$  (often  $\nu/2$  in Wishart contexts).  
**p** Integer. Dimension  $p$  of the multivariate gamma.

### Details

The multivariate gamma function  $\Gamma_p(a)$  is defined by:

$$\Gamma_p(a) = \pi^{p(p-1)/4} \prod_{j=1}^p \Gamma\left(a + \frac{1-j}{2}\right).$$

Constraints: (i)  $p \in \{1, 2, \dots\}$  (positive integer), and (ii)  $a > (p-1)/2$  to keep all gamma terms finite (as in the Wishart normalization constant).

### Value

A numeric scalar equal to  $\log \Gamma_p(a)$ .

### Examples

```
# Dimension
p <- 3
# Evaluate log multivariate gamma at a = nu/2
nu <- p + 5
lmgamma(a = nu / 2, p = p)
```

---

mixturewishart

*EM/Bayesian estimation for Wishart mixture model*


---

### Description

Fit finite mixtures of Wishart-distributed SPD matrices using either a Bayesian sampler or the EM algorithm. The input `S_list` is a list of  $p \times p$  SPD matrices. Under component  $k$ ,  $S_i \mid z_i = k \sim W_p(\nu_k, \Sigma_k)$  with degrees of freedom  $\nu_k$  and SPD scale matrix  $\Sigma_k$ . Mixture weights  $\pi_k$  sum to 1.

### Usage

```
mixturewishart(
  S_list,
  K,
  niter = 3000,
  burnin = 1000,
  method = "bayes",
  thin = 1,
  alpha = NULL,
  nu0 = NULL,
  Psi0 = NULL,
  init_pi = NULL,
  init_nu = NULL,
  init_Sigma = NULL,
  marginal.z = TRUE,
  estimate_nu = TRUE,
  nu_prior_a = 2,
  nu_prior_b = 0.1,
  mh_sigma = 1,
  n_restarts = 3,
  restart_iters = 20,
  tol = 1e-06,
  verbose = TRUE
)
```

### Arguments

<code>S_list</code>	List of length $n$ of SPD matrices, each $p \times p$ . These are the observed matrices modeled by a mixture of Wisharts.
<code>K</code>	Integer. Number of mixture components.
<code>niter</code>	Integer. Total iterations. Bayesian mode: total MCMC iterations (including burn-in). EM mode: maximum EM iterations (alias to <code>maxiter</code> ).
<code>burnin</code>	Integer. Number of burn-in iterations (Bayesian mode).
<code>method</code>	Character; one of <code>c("bayes", "em")</code> . Selects sampler or optimizer.
<code>thin</code>	Integer. Thinning interval for saving draws (Bayesian).

alpha	Numeric vector length $K$ (Dirichlet prior on $\pi$ ) or NULL to default to <code>rep(1, K)</code> (Bayesian).
nu0	Numeric. Inverse-Wishart prior df for $\Sigma_k$ (Bayesian). Default: $p + 2$ .
Psi0	Numeric $p \times p$ SPD matrix. Inverse-Wishart prior scale for $\Sigma_k$ (Bayesian). Default: <code>diag(p)</code> .
init_pi	Optional numeric vector length $K$ summing to 1. EM initialization for mixture weights. If NULL, random or data-driven initialization is used.
init_nu	Optional numeric vector length $K$ of initial degrees of freedom. Used in both modes.
init_Sigma	Optional list of $K$ SPD matrices (each $p \times p$ ). EM initialization for $\Sigma_k$ .
marginal.z	Logical. If TRUE, integrates out $\pi$ when sampling $z$ (collapsed step) in Bayesian mode. If FALSE, samples $z$ conditional on current $\pi$ .
estimate_nu	Logical. If TRUE, estimate/update $\nu_k$ (MH in Bayesian mode; Newton/EM in EM). If FALSE, $\nu_k$ are fixed.
nu_prior_a	Numeric. Prior hyperparameter $a$ for $\nu_k$ (Bayesian), used when <code>estimate_nu = TRUE</code> .
nu_prior_b	Numeric. Prior hyperparameter $b$ for $\nu_k$ (Bayesian), used when <code>estimate_nu = TRUE</code> .
mh_sigma	Numeric scalar or length- $K$ vector. Proposal sd for MH updates on $\log(\nu_k)$ (Bayesian, when estimating $\nu$ ).
n_restarts	Integer. Number of random restarts for EM. Ignored in Bayesian mode.
restart_iters	Integer. Number of short EM iterations per restart used to select a good initialization. Ignored in Bayesian mode.
tol	Numeric. Convergence tolerance on absolute change of log-likelihood (EM), also used internally elsewhere.
verbose	Logical. If TRUE, print progress information.

## Details

Mixture mixture model:  $p(S_i) = \sum_{k=1}^K \pi_k f_W(S_i | \nu_k, \Sigma_k)$ .

Algorithms:

1. `method = "bayes"`: Samples latent labels  $z$ , weights  $\pi$ , component scales  $\Sigma_k$ , and optionally  $\nu_k$ . Uses a Dirichlet prior for  $\pi$ , inverse-Wishart prior for  $\Sigma_k$ , and a prior on  $\nu_k$  when `estimate_nu = TRUE`. Degrees-of-freedom are updated via MH on  $\log(\nu_k)$  with proposal sd `mh_sigma`. Can integrate out  $\pi$  when sampling  $z$  if `marginal.z = TRUE`.
2. `method = "em"`: Maximizes the observed-data log-likelihood via EM. The E-step computes responsibilities via Wishart log-densities. The M-step updates  $\pi_k$  and  $\Sigma_k$ ; optionally updates  $\nu_k$  when `estimate_nu = TRUE`. Supports multiple random restarts.

Note that (i) All matrices in `S_list` must be SPD. Small ridge terms may be added internally for stability, and (ii) Multiple EM restarts are recommended for robustness on difficult datasets.

**Value**

A list whose structure depends on method:

- For method = "bayes":
  - pi\_ik: array (nsave x n x K), saved per-observation weights.
  - pi: matrix (nsave x K), saved mixture proportions.
  - nu: matrix (nsave x K), saved degrees- of-freedom.
  - Sigma: list of length nsave; each is an array ( $p \times p \times K$ ) of  $\Sigma_k$  draws.
  - z: matrix (nsave x n), saved allocations.
  - sigma\_posterior\_mean: array ( $p \times p \times K$ ), posterior mean of  $\Sigma_k$ .
  - loglik: numeric vector (length niter), log- likelihood trace.
  - loglik\_individual: matrix (niter x n), per-observation log-likelihood.
- For method = "em":
  - pi: numeric vector length  $K$ , mixture proportions.
  - Sigma: list length  $K$ , each a  $p \times p$  SPD matrix.
  - nu: numeric vector length  $K$ , degrees-of- freedom.
  - tau: matrix ( $n \times K$ ), responsibilities.
  - loglik: numeric vector, log-likelihood per EM iteration.
  - iterations: integer, number of EM iterations performed.

**Examples**

```
# simulate data
set.seed(123)
n <- 500 # subjects
p <- 2
dat <- simData(n, p,
  K = 3,
  pis = c(0.35, 0.40, 0.25),
  nus = c(8, 16, 3)
)

set.seed(123)
fit <- mixturewishart(
  dat$S,
  K = 3,
  mh_sigma = c(0.2, 0.1, 0.1), # tune this for MH acceptance 20-40%
  niter = 100, burnin = 50
)

# Posterior means for degrees of freedom of Wishart distributions:
nu_mcmc <- fit$nu[-c(1:fit$burnin), ]
colMeans(nu_mcmc)
```



moewishart

*EM/Bayesian estimation for Wishart MoE model***Description**

Fit a mixture-of-experts model for symmetric positive-definite (SPD) matrices with covariate-dependent mixing proportions (gating network). Components are Wishart-distributed. Supports Bayesian sampling and EM-based maximum-likelihood estimation.

**Usage**

```
moewishart(
  S_list,
  X,
  K,
  niter = 3000,
  burnin = 1000,
  method = "bayes",
  thin = 1,
  nu0 = NULL,
  Psi0 = NULL,
  init_nu = NULL,
  estimate_nu = TRUE,
  nu_prior_a = 2,
  nu_prior_b = 0.1,
  mh_sigma = 0.1,
  mh_beta = 0.05,
  sigma_beta = 10,
  init = NULL,
  tol = 1e-06,
  ridge = 1e-08,
  verbose = TRUE
)
```

**Arguments**

<code>S_list</code>	List of length $n$ of SPD matrices, each $p \times p$ . These are the observed responses modeled by the MoE.
<code>X</code>	Numeric matrix $n \times q$ of covariates for the gating network. Include an intercept column if desired.
<code>K</code>	Integer. Number of mixture components (experts).
<code>niter</code>	Integer. Total iterations. Bayesian mode: total MCMC iterations (including burn-in). EM mode: maximum EM iterations.
<code>burnin</code>	Integer. Number of burn-in iterations (Bayesian mode).
<code>method</code>	Character; one of <code>c("bayes", "em")</code> . Selects sampler or optimizer.

<code>thin</code>	Integer. Thinning interval for saving draws (Bayesian).
<code>nu0</code>	Numeric. Inverse-Wishart prior df for $\Sigma_k$ (Bayesian). Default: $p + 2$ if NULL.
<code>Psi0</code>	Numeric $p \times p$ SPD matrix. Inverse-Wishart prior scale for $\Sigma_k$ (Bayesian). Default: <code>diag(p)</code> if NULL.
<code>init_nu</code>	Optional numeric vector length $K$ of initial dfs $\nu_k$ . Used for initialization.
<code>estimate_nu</code>	Logical. If TRUE, estimate $\nu_k$ (MH in Bayesian; Newton/EM in EM). If FALSE, keep $\nu_k$ fixed at <code>init_nu</code> .
<code>nu_prior_a</code>	Numeric. Prior hyperparameter $a$ for $\nu_k$ (Bayesian), used when <code>estimate_nu = TRUE</code> .
<code>nu_prior_b</code>	Numeric. Prior hyperparameter $b$ for $\nu_k$ (Bayesian), used when <code>estimate_nu = TRUE</code> .
<code>mh_sigma</code>	Numeric scalar or length- $K$ vector. Proposal sd for MH updates on $\log(\nu_k)$ (Bayesian, when estimating $\nu$ ).
<code>mh_beta</code>	Numeric scalar or length- $K - 1$ vector. Proposal sd for MH updates of the free $B$ columns (Bayesian).
<code>sigma_beta</code>	Numeric. Prior sd of the Gaussian prior on $B$ (Bayesian).
<code>init</code>	Optional list with fields for EM initialization, e.g., <code>beta</code> , <code>Sigma</code> , <code>nu</code> . See return structure.
<code>tol</code>	Numeric. Convergence tolerance on absolute change of log-likelihood (EM), also used internally.
<code>ridge</code>	Numeric. Small diagonal ridge added to $\Sigma_k$ updates in EM for numerical stability.
<code>verbose</code>	Logical. If TRUE, print progress information.

## Details

### MoE-Wishart Model:

- Observation:  $S_i$  is a  $p \times p$  SPD matrix. Given allocation  $z_i = k$ ,  $S_i \mid z_i \sim W_p(\nu_k, \Sigma_k)$  with df  $\nu_k$  and scale  $\Sigma_k$ .
- Gating (MoE): Let  $X_i$  be  $q$ -dimensional covariates. Mixing weights  $\pi_{ik} = \Pr(z_i = k \mid X_i)$  follow a softmax regression:  $\pi_{ik} = \exp(\eta_{ik}) / \sum_{j=1}^K \exp(\eta_{ij})$ , where  $\eta_i = X_i^\top B$ ,  $B$  is  $q \times K$ . Identifiability: last column of  $B$  is fixed to zero.

### Algorithms:

1. Bayesian (method = "bayes"): Metropolis-within-Gibbs sampler for  $z$ ,  $\Sigma_k$ , optional  $\nu_k$ , and  $B$ . Gaussian priors on  $B$  with sd `sigma_beta`. Proposals use `mh_sigma` for  $\log(\nu_k)$  and `mh_beta` for  $B$ .
2. EM (method = "em"): E-step responsibilities using Wishart log-densities and softmax gating. M-step updates  $\Sigma_k$ , optional  $\nu_k$ , and  $B$  via weighted multinomial logistic regression (BFGS).

Note that: (i) include an intercept column in  $X$ ; none is added by default, and (ii) all `S_list` elements must be SPD. A small ridge may be added for stability.

**Value**

A list whose fields depend on method:

- For method = "bayes":
  - Beta\_samples: array (nsave x q x K), saved draws of  $B$  (last column zero).
  - nu\_samples: matrix (nsave x K), draws of  $\nu_k$ .
  - Sigma\_samples: list of length nsave; each element is an array ( $p \times p \times K$ ) of  $\Sigma_k$  draws.
  - z\_samples: matrix (nsave x n), draws of allocations.
  - pi\_ik: array (nsave x n x K), per-observation gating probabilities.
  - pi\_mean: matrix (n x K), posterior mean of gating probabilities.
  - loglik: numeric vector (length niter), log-likelihood trace.
  - loglik\_individual: matrix (niter x n), per-observation log-likelihood.
- For method = "em":
  - K, p, q, n: problem dimensions.
  - Beta: matrix ( $q \times K$ ), gating coefficients with last column zero (reference class).
  - Sigma: list length K, each a  $p \times p$  SPD matrix (scale).
  - nu: numeric vector length K, degrees of freedom.
  - gamma: matrix ( $n \times K$ ), final responsibilities.
  - loglik: numeric vector, log-likelihood by EM iteration.
  - iter: integer, number of EM iterations performed.

**Examples**

```
# simulate data
set.seed(123)
n <- 500 # subjects
p <- 2
# True gating coefficients (last column zero)
set.seed(123)
Xq <- 3
K <- 3
betas <- matrix(runif(Xq * K, -2, 2), nrow = Xq, ncol = K)
betas[, K] <- 0
dat <- simData(n, p,
  Xq = 3, K = 3, betas = betas,
  pis = c(0.35, 0.40, 0.25),
  nus = c(8, 16, 3)
)

set.seed(123)
fit <- moewishart(
  dat$$,
  X = cbind(1, dat$X), K = 3,
  mh_sigma = c(0.2, 0.1, 0.1), # RW-MH variances (length K)
  mh_beta = c(0.2, 0.2), # RW-MH variances (length K-1)
  niter = 500, burnin = 200
)
```

```
# Posterior means for degrees of freedom of Wishart distributions:
nu_mcmc <- fit$nu[-c(1:fit$burnin), ]
colMeans(nu_mcmc)
```

---

rdirichlet

*Dirichlet random sampling*


---

### Description

Generate random draws from a Dirichlet distribution with parameter vector  $\alpha \in \mathbb{R}_+^K$ . Each draw is a length- $K$  probability vector on the simplex.

### Usage

```
rdirichlet(n, alpha)
```

### Arguments

**n** Integer. Number of independent Dirichlet draws to generate.

**alpha** Numeric vector of positive concentration parameters  $\alpha = (\alpha_1, \dots, \alpha_K)$ . Its length  $K$  defines the dimension of the simplex.

### Details

Definition: If  $Y_k \sim \text{Gamma}(\alpha_k, 1)$  independently for  $k = 1, \dots, K$  (shape  $\alpha_k$ , rate 1), then the normalized vector  $X_k = Y_k / \sum_{j=1}^K Y_j$  follows  $\text{Dirichlet}(\alpha)$ .

Note that alpha must be a numeric vector with strictly positive entries.

### Value

A numeric matrix of size  $n \times K$ , where each row is an independent Dirichlet draw that sums to 1.

### Examples

```
set.seed(123)
# 3-dimensional Dirichlet with asymmetric concentration
alpha <- c(2, 5, 3)
rdirichlet(5, alpha)
```

---

sampleIW	<i>Fast sampler for the inverse-Wishart distribution</i>
----------	--

---

**Description**

Draw a random sample from an inverse-Wishart distribution  $\mathcal{IW}_p(\nu, \Psi)$  using the identity  $S \sim \mathcal{IW}_p(\nu, \Psi)$  iff  $S^{-1} \sim W_p(\nu, \Psi^{-1})$ . This implementation accepts  $\Psi^{-1}$  directly for speed.

**Usage**

```
sampleIW(nu, Psi_inv)
```

**Arguments**

nu	Numeric. Degrees of freedom $\nu$ of the inverse-Wishart (must exceed $p - 1$ ).
Psi_inv	Numeric $p \times p$ SPD matrix equal to $\Psi^{-1}$ , the inverse of the inverse-Wishart scale matrix.

**Details**

Sampling scheme:

- Sample  $W \sim W_p(\nu, \Psi^{-1})$  using `rWishart`.
- Return  $S = W^{-1}$ , which has  $\mathcal{IW}_p(\nu, \Psi)$ .

Parameterization:

- $\nu$  is the degrees of freedom, must satisfy  $\nu > p - 1$ .
- $\Psi$  is the SPD scale matrix of the inverse-Wishart. This function expects its inverse  $\Psi^{-1}$  as input for efficiency (avoids repeated matrix inversions).

Note that: (i) internally calls `rWishart(1, df = nu, Sigma = Psi_inv)`, and (ii) returns `solve(W)`; if numerical issues arise, consider adding a small ridge to  $\Psi^{-1}$  prior to sampling.

**Value**

A  $p \times p$  SPD matrix  $S$  distributed as  $\mathcal{IW}_p(\nu, \Psi)$ .

**Examples**

```
set.seed(123)
p <- 3
# Construct an SPD scale matrix Psi
A <- matrix(rnorm(p * p), p, p)
Psi <- crossprod(A) + diag(p) * 0.5
Psi_inv <- solve(Psi)

# Draw one inverse-Wishart sample
S <- sampleIW(nu = p + 5, Psi_inv = Psi_inv)
```

S

simData

*Simulate data from a Wishart mixture or mixture-of-experts model***Description**

Generate synthetic SPD matrices from either: (i) a finite mixture of Wishart components with fixed mixing proportions, or (ii) a mixture-of-experts (MoE) where mixing proportions depend on covariates via a softmax gating model.

**Usage**

```
simData(
  n = 200,
  p = 2,
  Xq = 0,
  K = NA,
  betas = NULL,
  pis = c(0.4, 0.6),
  nus = c(8, 12),
  Sigma = NULL
)
```

**Arguments**

n	Integer. Number of observations to simulate.
p	Integer. Dimension of the Wishart distribution (matrix size $p \times p$ ).
Xq	Integer. Number of covariates for the gating network (MoE case). If $Xq = 0$ , a standard mixture (no covariates) is simulated.
K	Integer. Number of latent components. Required when $Xq > 0$ . If $Xq = 0$ , defaults to <code>length(pis)</code> .
betas	Numeric matrix $Xq \times K$ of gating coefficients used when $Xq > 0$ . If NULL, random coefficients are generated and the last column is set to zero (reference class).
pis	Numeric vector of length $K$ giving fixed mixture proportions when $Xq = 0$ . Ignored when $Xq > 0$ .
nus	Numeric vector length $K$ , degrees of freedom $\nu_k$ for each component (must exceed $p - 1$ ).
Sigma	Optional list length $K$ of SPD scale matrices $\Sigma_k$ (each $p \times p$ ). If NULL, defaults are generated based on $K$ and $p$ .

## Details

Models:

- Fixed mixture (no covariates,  $\chi_q = \emptyset$ ):  $z_i \sim \text{Categorical}(\pi)$ , and  $S_i \mid z_i = k \sim W_p(\nu_k, \Sigma_k)$ .
- Mixture-of-experts (covariates,  $\chi_q > \emptyset$ ): Let  $X_i \in \mathbb{R}^{\chi_q}$ . The mixing weights are  $\pi_{ik} = \Pr(z_i = k \mid X_i)$  given by softmax regression  $\pi_{ik} = \exp(X_i^\top B_k) / \sum_{j=1}^K \exp(X_i^\top B_j)$ . Labels  $z_i$  are drawn from  $\text{Categorical}(\pi_i)$  and  $S_i \mid z_i = k \sim W_p(\nu_k, \Sigma_k)$ .

Simulation steps:

1. Construct `pis`:
  - If  $\chi_q = \emptyset$ , replicate the provided `pis` over `n` rows.
  - If  $\chi_q > \emptyset$ , generate  $X \sim N(0, I)$  and compute softmax probabilities using `betas` (last column set to zero by default identifiability).
2. If `Sigma` is not provided, create default  $\Sigma_k$  matrices (SPD) depending on `K` and `p`.
3. Sample labels  $z_i \sim \text{Categorical}(\pi_i)$ .
4. Draw  $S_i$  from  $W_p(\nu_{z_i}, \Sigma_{z_i})$  via `rWishart`.

Note that: (i) in the MoE case, no intercept is automatically added to `X`. Use `χq` to include desired covariates; the default `betas` is randomly generated with `betas[, K] = 0`, and (ii) provided `Sigma` must be a list of SPD  $p \times p$  matrices. Provided `nus` must exceed  $p - 1$ .

## Value

A list with the following elements:

- `S`: list of length `n` of simulated SPD matrices  $S_i$ .
- `z`: integer vector length `n` of component labels.
- `nu`: numeric vector length `K` of degrees of freedom.
- `pi`: matrix  $n \times K$  of mixing probabilities (rows sum to 1).
- `Sigma_list`: list length `K` of the scale matrices used for simulation.
- `X`: matrix  $n \times \chi_q$  of covariates if  $\chi_q > \emptyset$ , otherwise `NULL`.
- `betas`: the gating coefficient matrix used when  $\chi_q > \emptyset$ , otherwise `NULL`.

## Examples

```
# simulate data from mixture model (no covariates)
set.seed(123)
n <- 200 # subjects
p <- 10
dat <- simData(n, p,
  K = 3,
  pis = c(0.35, 0.40, 0.25),
  nus = c(8, 12, 3)
)
str(dat)
```

# Index

`computeIC`, 2

`dWishart`, 3

`lmvgamma`, 5

`mixturewishart`, 6

`moewishart`, 9

`rdirichlet`, 12

`sampleIW`, 13

`simData`, 14