

# Package ‘LMN’

May 7, 2026

**Type** Package

**Title** Inference for Linear Models with Nuisance Parameters

**Version** 1.1.3

**Date** 2022-08-11

**Description** Efficient Frequentist profiling and Bayesian marginalization of parameters for which the conditional likelihood is that of a multivariate linear regression model. Arbitrary inter-observation error correlations are supported, with optimized calculations provided for independent-heteroskedastic and stationary dependence structures.

**URL** <https://github.com/mlsy/LMN>

**BugReports** <https://github.com/mlsy/LMN/issues>

**License** GPL-3

**Imports** Rcpp (>= 0.12.4.4), SuperGauss, stats

**LinkingTo** Rcpp, RcppEigen

**Encoding** UTF-8

**RoxygenNote** 7.2.1

**Suggests** testthat, numDeriv, mniw, knitr, rmarkdown, bookdown, kableExtra

**VignetteBuilder** knitr

**NeedsCompilation** yes

**Author** Martin Lysy [aut, cre],  
Bryan Yates [aut]

**Maintainer** Martin Lysy <[mlsy@uwaterloo.ca](mailto:mlsy@uwaterloo.ca)>

**Repository** CRAN

**Date/Publication** 2022-08-22 16:20:02 UTC

## Contents

LMN-package . . . . .	2
list2mniw . . . . .	3

lmn_loglik . . . . .	3
lmn_marg . . . . .	4
lmn_post . . . . .	5
lmn_prior . . . . .	6
lmn_prof . . . . .	7
lmn_suff . . . . .	8
<b>Index</b>	<b>11</b>

LMN-package

*Inference for Linear Models with Nuisance Parameters.*

## Description

Efficient profile likelihood and marginal posteriors when nuisance parameters are those of linear regression models.

## Details

Consider a model  $p(\mathbf{Y} \mid \mathbf{B}, \boldsymbol{\Sigma}, \boldsymbol{\theta})$  of the form

$$\mathbf{Y} \sim \text{Matrix-Normal}(\mathbf{X}(\boldsymbol{\theta})\mathbf{B}, \mathbf{V}(\boldsymbol{\theta}), \boldsymbol{\Sigma}),$$

where  $\mathbf{Y}_{n \times q}$  is the response matrix,  $\mathbf{X}(\boldsymbol{\theta})_{n \times p}$  is a covariate matrix which depends on  $\boldsymbol{\theta}$ ,  $\mathbf{B}_{p \times q}$  is the coefficient matrix,  $\mathbf{V}(\boldsymbol{\theta})_{n \times n}$  and  $\boldsymbol{\Sigma}_{q \times q}$  are the between-row and between-column variance matrices, and (suppressing the dependence on  $\boldsymbol{\theta}$ ) the Matrix-Normal distribution is defined by the multivariate normal distribution  $\text{vec}(\mathbf{Y}) \sim \mathcal{N}(\text{vec}(\mathbf{X}\mathbf{B}), \boldsymbol{\Sigma} \otimes \mathbf{V})$ , where  $\text{vec}(\mathbf{Y})$  is a vector of length  $nq$  stacking the columns of  $\mathbf{Y}$ , and  $\boldsymbol{\Sigma} \otimes \mathbf{V}$  is the Kronecker product.

The model above is referred to as a Linear Model with Nuisance parameters (LMN)  $(\mathbf{B}, \boldsymbol{\Sigma})$ , with parameters of interest  $\boldsymbol{\theta}$ . That is, the **LMN** package provides tools to efficiently conduct inference on  $\boldsymbol{\theta}$  first, and subsequently on  $(\mathbf{B}, \boldsymbol{\Sigma})$ , by Frequentist profile likelihood or Bayesian marginal inference with a Matrix-Normal Inverse-Wishart (MNIW) conjugate prior on  $(\mathbf{B}, \boldsymbol{\Sigma})$ .

## Author(s)

**Maintainer:** Martin Lysy <mlysy@uwaterloo.ca>

Authors:

- Bryan Yates

## See Also

Useful links:

- <https://github.com/mlysy/LMN>
- Report bugs at <https://github.com/mlysy/LMN/issues>

---

list2mniw	<i>Convert list of MNIW parameter lists to vectorized format.</i>
-----------	---

---

**Description**

Converts a list of return values of multiple calls to `lmn_prior()` or `lmn_post()` to a single list of MNIW parameters, which can then serve as vectorized arguments to the functions in **mniw**.

**Usage**

```
list2mniw(x)
```

**Arguments**

`x` List of  $n$  MNIW parameter lists.

**Value**

A list with the following elements:

`Lambda` The mean matrices as an array of size  $p \times p \times n$ .

`Omega` The between-row precision matrices, as an array of size  $p \times p \times n$ .

`Psi` The between-column scale matrices, as an array of size  $q \times q \times n$ .

`nu` The degrees-of-freedom parameters, as a vector of length  $n$ .

---

lmn_loglik	<i>Loglikelihood function for LMN models.</i>
------------	---

---

**Description**

Loglikelihood function for LMN models.

**Usage**

```
lmn_loglik(Beta, Sigma, suff)
```

**Arguments**

`Beta` A  $p \times q$  matrix of regression coefficients (see `lmn_suff()`).

`Sigma` A  $q \times q$  matrix of error variances (see `lmn_suff()`).

`suff` An object of class `lmn_suff` (see `lmn_suff()`).

**Value**

Scalar; the value of the loglikelihood.

**Examples**

```
# generate data
n <- 50
q <- 3
Y <- matrix(rnorm(n*q),n,q) # response matrix
X <- 1 # intercept covariate
V <- 0.5 # scalar variance specification
suff <- lmn_suff(Y, X = X, V = V) # sufficient statistics

# calculate loglikelihood
Beta <- matrix(rnorm(q),1,q)
Sigma <- diag(rexp(q))
lmn_loglik(Beta = Beta, Sigma = Sigma, suff = suff)
```

---

lmn\_marg

---

*Marginal log-posterior for the LMN model.*


---

**Description**

Marginal log-posterior for the LMN model.

**Usage**

```
lmn_marg(suff, prior, post)
```

**Arguments**

suff	An object of class <code>lmn_suff</code> (see <code>lmn_suff()</code> ).
prior	A list with elements <code>Lambda</code> , <code>Omega</code> , <code>Psi</code> , <code>nu</code> corresponding to the parameters of the prior MNIW distribution. See <code>lmn_prior()</code> .
post	A list with elements <code>Lambda</code> , <code>Omega</code> , <code>Psi</code> , <code>nu</code> corresponding to the parameters of the posterior MNIW distribution. See <code>lmn_post()</code> .

**Value**

The scalar value of the marginal log-posterior.

**Examples**

```
# generate data
n <- 50
q <- 2
p <- 3
Y <- matrix(rnorm(n*q),n,q) # response matrix
X <- matrix(rnorm(n*p),n,p) # covariate matrix
V <- .5 * exp(-(1:n)/n) # Toeplitz variance specification

suff <- lmn_suff(Y = Y, X = X, V = V, Vtype = "acf") # sufficient statistics
```

```
# default noninformative prior pi(Beta, Sigma) ~ |Sigma|^(-(q+1)/2)
prior <- lmn_prior(p = suff$p, q = suff$q)
post <- lmn_post(suff, prior = prior) # posterior MNIW parameters
lmn_marg(suff, prior = prior, post = post)
```

---

lmn\_post

*Parameters of the posterior conditional distribution of an LMN model.*


---

## Description

Calculates the parameters of the LMN model's Matrix-Normal Inverse-Wishart (MNIW) conjugate posterior distribution (see **Details**).

## Usage

```
lmn_post(suff, prior)
```

## Arguments

`suff` An object of class `lmn_suff` (see [lmn\\_suff\(\)](#)).

`prior` A list with elements `Lambda`, `Omega`, `Psi`, `nu` as returned by [lmn\\_prior\(\)](#).

## Details

The Matrix-Normal Inverse-Wishart (MNIW) distribution  $(B, \Sigma) \sim \text{MNIW}(\Lambda, \Omega, \Psi, \nu)$  on random matrices  $X_{p \times q}$  and symmetric positive-definite  $\Sigma_{q \times q}$  is defined as

$$\begin{aligned} \Sigma &\sim \text{Inverse-Wishart}(\Psi, \nu) \\ B \mid \Sigma &\sim \text{Matrix-Normal}(\Lambda, \Omega^{-1}, \Sigma), \end{aligned}$$

where the Matrix-Normal distribution is defined in [lmn\\_suff\(\)](#).

The posterior MNIW distribution is required to be a proper distribution, but the prior is not. For example, `prior = NULL` corresponds to the noninformative prior

$$\pi(B, \Sigma) \sim |\mathbf{Sigma}|^{-(q+1)/2}.$$

## Value

A list with elements named as in `prior` specifying the parameters of the posterior MNIW distribution. Elements `Omega = NA` and `nu = NA` specify that parameters `Beta = 0` and `Sigma = diag(q)`, respectively, are known and not to be estimated.

## Examples

```
# generate data
n <- 50
q <- 2
p <- 3
Y <- matrix(rnorm(n*q),n,q) # response matrix
X <- matrix(rnorm(n*p),n,p) # covariate matrix
V <- .5 * exp(-(1:n)/n) # Toeplitz variance specification

suff <- lmn_suff(Y = Y, X = X, V = V, Vtype = "acf") # sufficient statistics
```

---

lmn\_prior

*Conjugate prior specification for LMN models.*


---

## Description

The conjugate prior for LMN models is the Matrix-Normal Inverse-Wishart (MNIW) distribution. This convenience function converts a partial MNIW prior specification into a full one.

## Usage

```
lmn_prior(p, q, Lambda, Omega, Psi, nu)
```

## Arguments

p	Integer specifying row dimension of Beta. $p = 0$ corresponds to no Beta in the model, i.e., $X = \emptyset$ in <code>lmn_suff()</code> .
q	Integer specifying the dimension of Sigma.
Lambda	Mean parameter for Beta. Either: <ul style="list-style-type: none"> <li>• A <math>p \times q</math> matrix.</li> <li>• A scalar, in which case <code>Lambda = matrix(Lambda, p, q)</code>.</li> <li>• Missing, in which case <code>Lambda = matrix(0, p, q)</code>.</li> </ul>
Omega	Row-wise precision parameter for Beta. Either: <ul style="list-style-type: none"> <li>• A <math>p \times p</math> matrix.</li> <li>• A scalar, in which case <code>Omega = diag(rep(Omega, p))</code>.</li> <li>• Missing, in which case <code>Omega = matrix(0, p, p)</code>.</li> <li>• NA, which signifies that Beta is known, in which case the prior is purely Inverse-Wishart on Sigma (see <b>Details</b>).</li> </ul>
Psi	Scale parameter for Sigma. Either: <ul style="list-style-type: none"> <li>• A <math>q \times q</math> matrix.</li> <li>• A scalar, in which case <code>Psi = diag(rep(Psi, q))</code>.</li> <li>• Missing, in which case <code>Psi = matrix(0, q, q)</code>.</li> </ul>
nu	Degrees-of-freedom parameter for Sigma. Either a scalar, missing (defaults to $\nu = 0$ ), or NA, which signifies that <code>Sigma = diag(q)</code> is known, in which case the prior is purely Matrix-Normal on Beta (see <b>Details</b> ).

**Details**

The Matrix-Normal Inverse-Wishart (MNIW) distribution  $(B, \Sigma) \sim \text{MNIW}(\Lambda, \Omega, \Psi, \nu)$  on random matrices  $X_{p \times q}$  and symmetric positive-definite  $\Sigma_{q \times q}$  is defined as

$$\begin{aligned} \Sigma &\sim \text{Inverse-Wishart}(\Psi, \nu) \\ B \mid \Sigma &\sim \text{Matrix-Normal}(\Lambda, \Omega^{-1}, \Sigma), \end{aligned}$$

where the Matrix-Normal distribution is defined in `lmn_suff()`.

**Value**

A list with elements Lambda, Omega, Psi, nu with the proper dimensions specified above, except possibly Omega = NA or nu = NA (see **Details**).

**Examples**

```
# problem dimensions
p <- 2
q <- 4

# default noninformative prior pi(Beta, Sigma) ~ |Sigma|^(-(q+1)/2)
lmn_prior(p, q)

# pi(Sigma) ~ |Sigma|^(-(q+1)/2)
# Beta | Sigma ~ Matrix-Normal(0, I, Sigma)
lmn_prior(p, q, Lambda = 0, Omega = 1)

# Sigma = diag(q)
# Beta ~ Matrix-Normal(0, I, Sigma = diag(q))
lmn_prior(p, q, Lambda = 0, Omega = 1, nu = NA)
```

---

lmn\_prof

*Profile loglikelihood for the LMN model.*


---

**Description**

Calculate the loglikelihood of the LMN model defined in `lmn_suff()` at the MLE Beta = Bhat and Sigma = Sigma.hat.

**Usage**

```
lmn_prof(suff, noSigma = FALSE)
```

**Arguments**

suff            An object of class `lmn_suff` (see `lmn_suff()`).

noSigma        Logical. If TRUE assumes that Sigma = diag(ncol(Y)) is known and therefore not estimated.

**Value**

Scalar; the calculated value of the profile loglikelihood.

**Examples**

```
# generate data
n <- 50
q <- 2
Y <- matrix(rnorm(n*q),n,q) # response matrix
X <- matrix(1,n,1) # covariate matrix
V <- exp(-(1:n)/n) # diagonal variance specification
suff <- lmn_suff(Y, X = X, V = V, Vtype = "diag") # sufficient statistics

# profile loglikelihood
lmn_prof(suff)

# check that it's the same as loglikelihood at MLE
lmn_loglik(Beta = suff$Bhat, Sigma = suff$S/suff$n, suff = suff)
```

---

lmn\_suff

---

*Calculate the sufficient statistics of an LMN model.*


---

**Description**

Calculate the sufficient statistics of an LMN model.

**Usage**

```
lmn_suff(Y, X, V, Vtype, npred = 0)
```

**Arguments**

- |          |   |
|----------|---|
| Y        | An $n \times q$ matrix of responses.  |
| X        | An $N \times p$ matrix of covariates, where $N = n + \text{npred}$ (see <b>Details</b> ). May also be passed as: <ul style="list-style-type: none"> <li>• A scalar, in which case the one-column covariate matrix is <math>X = X * \text{matrix}(1, N, 1)</math>. <math>-X = 0</math>, in which case the mean of <math>Y</math> is known to be zero, i.e., no regression coefficients are estimated.</li> </ul>   |
| V, Vtype | The between-observation variance specification. Currently the following options are supported: <ul style="list-style-type: none"> <li>• <code>Vtype = "full"</code>: <math>V</math> is an <math>N \times N</math> symmetric positive-definite matrix.</li> <li>• <code>Vtype = "diag"</code>: <math>V</math> is a vector of length <math>N</math> such that <math>V = \text{diag}(V)</math>.</li> <li>• <code>Vtype = "scalar"</code>: <math>V</math> is a scalar such that <math>V = V * \text{diag}(N)</math>.</li> <li>• <code>Vtype = "acf"</code>: <math>V</math> is either a vector of length <math>N</math> or an object of class <code>SuperGauss::Toeplitz</code>, such that <math>V = \text{toeplitz}(V)</math>.</li> </ul> |

	For $V$ specified as a matrix or scalar, $V$ type is deduced automatically and need not be specified.
npred	A nonnegative integer. If positive, calculates sufficient statistics to make predictions for new responses. See <b>Details</b> .

### Details

The multi-response normal linear regression model is defined as

$$\mathbf{Y} \sim \text{Matrix-Normal}(\mathbf{X}\mathbf{B}, \mathbf{V}, \mathbf{\Sigma}),$$

where  $\mathbf{Y}_{n \times q}$  is the response matrix,  $\mathbf{X}_{n \times p}$  is the covariate matrix,  $\mathbf{B}_{p \times q}$  is the coefficient matrix,  $\mathbf{V}_{n \times n}$  and  $\mathbf{\Sigma}_{q \times q}$  are the between-row and between-column variance matrices, and the Matrix-Normal distribution is defined by the multivariate normal distribution  $\text{vec}(\mathbf{Y}) \sim \mathcal{N}(\text{vec}(\mathbf{X}\mathbf{B}), \mathbf{\Sigma} \otimes \mathbf{V})$ , where  $\text{vec}(\mathbf{Y})$  is a vector of length  $nq$  stacking the columns of  $\mathbf{Y}$ , and  $\mathbf{\Sigma} \otimes \mathbf{V}$  is the Kronecker product.

The function `lmm_suff()` returns everything needed to efficiently calculate the likelihood function

$$\mathcal{L}(\mathbf{B}, \mathbf{\Sigma} \mid \mathbf{Y}, \mathbf{X}, \mathbf{V}) = p(\mathbf{Y} \mid \mathbf{X}, \mathbf{V}, \mathbf{B}, \mathbf{\Sigma}).$$

When `npred > 0`, define the variables  $\mathbf{Y}_\text{star} = \text{rbind}(\mathbf{Y}, \mathbf{y})$ ,  $\mathbf{X}_\text{star} = \text{rbind}(\mathbf{X}, \mathbf{x})$ , and  $\mathbf{V}_\text{star} = \text{rbind}(\text{cbind}(\mathbf{V}, \mathbf{w}), \text{cbind}(\mathbf{t}(\mathbf{w}), \mathbf{v}))$ . Then `lmm_suff()` calculates summary statistics required to estimate the conditional distribution

$$p(\mathbf{y} \mid \mathbf{Y}, \mathbf{X}_\text{star}, \mathbf{V}_\text{star}, \mathbf{B}, \mathbf{\Sigma}).$$

The inputs to `lmm_suff()` in this case are  $\mathbf{Y} = \mathbf{Y}$ ,  $\mathbf{X} = \mathbf{X}_\text{star}$ , and  $\mathbf{V} = \mathbf{V}_\text{star}$ .

### Value

An S3 object of type `lmm_suff`, consisting of a list with elements:

**Bhat** The  $p \times q$  matrix  $\hat{\mathbf{B}} = (\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-1}\mathbf{X}'\mathbf{V}^{-1}\mathbf{Y}$ .

**T** The  $p \times p$  matrix  $\mathbf{T} = \mathbf{X}'\mathbf{V}^{-1}\mathbf{X}$ .

**S** The  $q \times q$  matrix  $\mathbf{S} = (\mathbf{Y} - \mathbf{X}\hat{\mathbf{B}})'\mathbf{V}^{-1}(\mathbf{Y} - \mathbf{X}\hat{\mathbf{B}})$ .

**ldV** The scalar log-determinant of  $\mathbf{V}$ .

**n, p, q** The problem dimensions, namely  $n = \text{nrow}(\mathbf{Y})$ ,  $p = \text{nrow}(\mathbf{Beta})$  (or  $p = 0$  if  $\mathbf{X} = \emptyset$ ), and  $q = \text{ncol}(\mathbf{Y})$ .

In addition, when `npred > 0` and with  $\mathbf{x}$ ,  $\mathbf{w}$ , and  $\mathbf{v}$  defined in **Details**:

**Ap** The `npred`  $\times$   $q$  matrix  $\mathbf{A}_p = \mathbf{w}'\mathbf{V}^{-1}\mathbf{Y}$ .

**Xp** The `npred`  $\times$   $p$  matrix  $\mathbf{X}_p = \mathbf{x} - \mathbf{w}\mathbf{V}^{-1}\mathbf{X}$ .

**Vp** The scalar  $V_p = \mathbf{v} - \mathbf{w}\mathbf{V}^{-1}\mathbf{w}$ .

**Examples**

```
# Data
n <- 50
q <- 3
Y <- matrix(rnorm(n*q),n,q)

# No intercept, diagonal V input
X <- 0
V <- exp(-(1:n)/n)
lmn_suff(Y, X = X, V = V, Vtype = "diag")

# X = (scaled) Intercept, scalar V input (no need to specify Vtype)
X <- 2
V <- .5
lmn_suff(Y, X = X, V = V)

# X = dense matrix, Toeplitz variance matrix
p <- 2
X <- matrix(rnorm(n*p), n, p)
Tz <- SuperGauss::Toeplitz$new(acf = 0.5*exp(-seq(1:n)/n))
lmn_suff(Y, X = X, V = Tz, Vtype = "acf")
```

# Index

`list2mniw`, 3  
LMN (LMN-package), 2  
LMN-package, 2  
`lmn_loglik`, 3  
`lmn_marg`, 4  
`lmn_post`, 5  
`lmn_post()`, 3, 4  
`lmn_prior`, 6  
`lmn_prior()`, 3–5  
`lmn_prof`, 7  
`lmn_suff`, 8  
`lmn_suff()`, 3–7  
`SuperGauss::Toeplitz`, 8