Package ‘LMN’

August 22, 2022

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/mlysy/LMN

BugReports https://github.com/mlysy/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 <mlysy@uwaterloo.ca>

Repository CRAN

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

R topics documented:

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

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

where $\boldsymbol{Y}_{n\times q}$ is the response matrix, $\boldsymbol{X}(\theta)_{n\times p}$ is a covariate matrix which depends on $\boldsymbol{\theta}$, $\boldsymbol{B}_{p\times q}$ is the coefficient matrix, $\boldsymbol{V}(\boldsymbol{\theta})_{n\times n}$ 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 $\text{vec}(\boldsymbol{Y}) \sim \mathcal{N}(\text{vec}(\boldsymbol{X}\boldsymbol{B}), \boldsymbol{\Sigma} \otimes \boldsymbol{V})$, where $\text{vec}(\boldsymbol{Y})$ is a vector of length nq stacking the columns of of $\boldsymbol{Y}$, and $\boldsymbol{\Sigma} \otimes \boldsymbol{V}$ is the Kronecker product.

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

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 — *Convert list of MNIW parameter lists to vectorized format.*

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 x p x n`.

`Omega` The between-row precision matrices, as an array of size `p x p x .`

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

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

lmn_loglik — *Loglikelihood function for LMN models.*

Description

Loglikelihood function for LMN models.

Usage

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

Arguments

`Beta`	A `p x q` matrix of regression coefficients (see `lmn_suff()`).
`Sigma`	A `q x 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 `lmn_suff` (see `lmn_suff()`).
`prior`	A list with elements `Lambda`, `Omega`, `Psi`, `nu` corresponding to the parameters of the prior MNIW distribution. See `lmn_prior()`.
`post`	A list with elements `Lambda`, `Omega`, `Psi`, `nu` corresponding to the parameters of the posterior MNIW distribution. See `lmn_post()`.

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 $(\boldsymbol{B}, \boldsymbol{\Sigma}) \sim \text{MNIW}(\boldsymbol{\Lambda}, \boldsymbol{\Omega}, \boldsymbol{\Psi}, \nu)$ on random matrices $\boldsymbol{X}_{p \times q}$ and symmetric positive-definite $\boldsymbol{\Sigma}_{q \times q}$ is defined as

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

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

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

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 = 0` in `lmn_suff()`.
q	Integer specifying the dimension of `Sigma`.
Lambda	Mean parameter for `Beta`. Either: • A `p x q` matrix. • A scalar, in which case `Lambda = matrix(Lambda, p, q)`. • Missing, in which case `Lambda = matrix(0, p, q)`.
Omega	Row-wise precision parameter for `Beta`. Either: • A `p x p` matrix. • A scalar, in which case `Omega = diag(rep(Omega,p))`. • Missing, in which case `Omega = matrix(0, p, p)`. • `NA`, which signifies that `Beta` is known, in which case the prior is purely Inverse-Wishart on `Sigma` (see **Details**).
Psi	Scale parameter for `Sigma`. Either: • A `q x q` matrix. • A scalar, in which case `Psi = diag(rep(Psi,q))`. • Missing, in which case `Psi = matrix(0, q, q)`.
nu	Degrees-of-freedom parameter for `Sigma`. Either a scalar, missing (defaults to `nu = 0`), or `NA`, which signifies that `Sigma = diag(q)` is known, in which case the prior is purely Matrix-Normal on `Beta` (see **Details**).

Details

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

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

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 x q` matrix of responses.
`X`	An `N x p` matrix of covariates, where `N` = `n` + `npred` (see **Details**). May also be passed as: • A scalar, in which case the one-column covariate matrix is `X` = `X * matrix(1, N, 1)`. -`X` = `0`, in which case the mean of `Y` is known to be zero, i.e., no regression coefficients are estimated.
`V, Vtype`	The between-observation variance specification. Currently the following options are supported: • `Vtype = "full"`: `V` is an `N x N` symmetric positive-definite matrix. • `Vtype = "diag"`: `V` is a vector of length `N` such that `V` = `diag(V)`. • `Vtype = "scalar"`: `V` is a scalar such that `V` = `V * diag(N)`. • `Vtype = "acf"`: `V` is either a vector of length `N` or an object of class `SuperGauss::Toeplitz`, such that `V` = `toeplitz(V)`.

For `V` specified as a matrix or scalar, `Vtype` is deduced automatically and need not be specified.

`npred` A nonnegative integer. If positive, calculates sufficient statistics to make predictions for new responses. See **Details**.

Details

The multi-response normal linear regression model is defined as

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

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

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

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

When `npred > 0`, define the variables `Y_star = rbind(Y, y)`, `X_star = rbind(X, x)`, and `V_star = rbind(cbind(V, w), cbind(t(w), v))`. Then `lmn_suff()` calculates summary statistics required to estimate the conditional distribution

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

The inputs to `lmn_suff()` in this case are `Y = Y`, `X = X_star`, and `V = V_star`.

Value

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

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

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

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

`ldV` The scalar log-determinant of `V`.

`n`, `p`, `q` The problem dimensions, namely `n = nrow(Y)`, `p = nrow(Beta)` (or `p = 0` if `X = 0`), and `q = ncol(Y)`.

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

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

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

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

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