Package 'colvR'

November 12, 2026

Type Package

Title Poisson Lognormal with Dimension Reduction for Count Matrices with Missing Data

Version 0.2.1

Description Impute missing data in count tables using a Poisson lognormal model with dimension reduction, with and without zero inflation, as described in Bricout et al. (2026) <doi:10.48550/arXiv.2602.10673>. Parameters are estimated with a variational expectation-maximization algorithm. The package also provides the asymptotic variance of the estimators, prediction intervals for the imputed counts, and model selection criteria (Bayesian information criterion, BIC, and integrated completed likelihood, ICL) to choose the dimension of the latent space.

License GPL (>= 3)

Encoding UTF-8

Language en-US

Imports Rcpp, PLNmodels, Matrix, mvtnorm, pbapply, lori, MASS

LinkingTo Rcpp, RcppArmadillo, nloptr

Suggests testthat (>= 3.0.0), knitr, rmarkdown, spelling

Config/testthat/edition 3

VignetteBuilder knitr

SystemRequirements NLopt, C++17

URL https://github.com/BBricout/colvR, https://bbricout.github.io/colvR/

BugReports https://github.com/BBricout/colvR/issues

Depends R (>= 3.5)

Config/roxygen2/version 8.1.1

NeedsCompilation yes

Author Barbara Bricout [aut, cre]

Maintainer Barbara Bricout <barbara.bricout@gmail.com>

Repository CRAN

Date/Publication 2026-11-12 14:50:08 UTC

Contents

BIC . . . 2
covmat . . . 3
Elbo_grad . . . 4
Elbo_grad_logS . . . 5
IC . . . 6
ICL . . . 6
Init . . . 7
Init_ZIP . . . 8
Init_ZIP_q0 . . . 9
Miss.PLNPCA . . . 10
Miss.ZIPLNPCA . . . 11
Miss.ZIPLNPCA.logS . . . 13
Miss.ZIPLNPCA_Steps . . . 14
Miss.ZIPLNPCA_VE . . . 16
Predictions . . . 18
select_model_bic . . . 20
Simul . . . 21
V_theta . . . 22
waterbird . . . 23

Index **24**

`BIC` *BIC from the ELBO for PLN-PCA and ZIP-PLN-PCA models*

Description

Computes a BIC-like score in which the log-likelihood is replaced by the ELBO of the fitted model (higher is better):

$$\text{BIC} = \text{ELBO} - \frac{\log(n)}{2}\Big[q\{p - (q-1)/2\} + k\Big],$$

where $q\{p-(q-1)/2\}$ is the number of free parameters in C (identifiable up to a rotation) and k the number of regression parameters: $2d$ for the ZIP-PLN-PCA model (β and γ), d for the PLN-PCA model with a vectorized design ($(np) \times d$) and pd with a rowwise design ($n \times d$). n is the number of rows of `Y` (sites), which are the independent units of the model.

Usage

```
BIC(Y, X, fit, q, model = c("ZIPLNPCA", "PLNPCA"))
```

Arguments

Y	Numeric n x p count matrix (NAs allowed).
X	Design matrix, (n*p) x d (or n x d for a rowwise PLN-PCA design).
fit	Fitted object containing $elbo (output of Miss.ZIPLNPCA or Miss.PLNPCA).
q	Integer, size of the latent space.
model	"ZIPLNPCA" (default) or "PLNPCA".

Value

A single numeric value, the BIC score.

See Also

ICL

---

covmat *Wrapper for 'lori::covmat'*

Description

Thin helper that calls `lori::covmat()` without requiring users to attach the **lori** package. It constructs the block-structured covariate matrix X (with $n \times p$ rows) from site-level covariates `R`, year-level covariates `C`, and optional site–year covariates `E`. Any of `R`, `C`, or `E` can be `NULL` to omit that block.

Usage

```
covmat(n, p, R = NULL, C = NULL, E = NULL, center = FALSE)
```

Arguments

n	Integer; number of sites.
p	Integer; number of years.
R	Optional numeric matrix with n rows and d_R columns (site-level covariates). Default: `NULL`.
C	Optional numeric matrix with p rows and d_C columns (year-level covariates). Default: `NULL`.
E	Optional numeric matrix with np rows and d_E columns (site–year covariates; one row per site–year observation in the same order as X). Default: `NULL`.
center	Logical; whether to center columns within each block (as in `lori::covmat`). Default: `FALSE`.

Value

A numeric matrix with $n \times p$ rows and $d = d_R + d_C + d_E$ columns (when present), identical to the value returned by `lori::covmat()`.

See Also

covmat

Examples

```
if (requireNamespace("lori", quietly = TRUE)) {
  set.seed(1)
  n <- 3; p <- 2
  R <- matrix(rnorm(n * 2), nrow = n, ncol = 2)  # site-level (d_R = 2)
  C <- matrix(rnorm(p * 2), nrow = p, ncol = 2)  # year-level (d_C = 2)
  X <- covmat(n, p, R = R, C = C)
  dim(X)  # 6 x 4
}
```

Elbo_grad *ELBO and parameter gradients (ZIP-PLN variational objective)*

Description

Computes the Evidence Lower Bound (ELBO) and its gradients with respect to the model and variational parameters for a zero-inflated Poisson log-normal (ZIP-PLN) latent factor model. This is a thin R wrapper around a fast C++ implementation, with a small preprocessing step that replaces NA values in Y by zeros before delegation.

Usage

```
Elbo_grad(data, params, tolxi)
```

Arguments

data	A list with elements:
	Y Numeric n x p count matrix (may contain NA).
	X Numeric n x d design matrix for the abundance/log-mean part.
	R Numeric/binary design matrix for the zero-inflation (logit) part (same number of rows as Y); can be NULL if not used.
params	A list with elements:
	B Matrix of regression coefficients for X (abundance).
	D Matrix of regression coefficients for X (zero-inflation).
	C Loadings matrix for the latent factors (rank q).
	M Variational means of the latent factors.
	S Variational standard deviations (or log-SDs) of the latent factors.
tolxi	Numeric, convergence tolerance used for the variational bound on the logistic term.

Details

This function sets NA entries of data$Y to 0 internally (they are not dropped) before calling the C++ backend Elbo_grad_Rcpp. Ensure the dimensions of X, R and the parameter matrices are compatible with Y.

Value

A list containing at least:

elbo Scalar numeric, the ELBO value.

grad_B, grad_D, grad_C, grad_M, grad_S Gradients w.r.t. the corresponding parameters.

See Also

Elbo_grad_Rcpp (backend C++).

Elbo_grad_logS *Calculate the Elbo with log(S) instead of S and the gradients*

Description

Calculate the Elbo with log(S) instead of S and the gradients

Usage

```
Elbo_grad_logS(data, params)
```

Arguments

data	list(Y, R, X)
params	list(B, D, C, M, S)

Value

Elbo and gradients

IC *95% interval for a theta 95% confidence interval for theta*

Description

Computes the 95% confidence interval for `theta`.

Usage

```
IC(theta, var)
```

Arguments

`theta`	Numeric vector of estimates.
`var`	Numeric vector of associated variances (same length as `theta`).

Value

A numeric vector of length 2: lower and upper bounds.

Examples

```
IC(1, 0.04)
```

ICL *ICL from the ELBO for PLN-PCA and ZIP-PLN-PCA models*

Description

Integrated Completed Likelihood: the BIC minus the entropy of the variational distribution of the latent variables (higher is better):

$$\mathrm{ICL} = \mathrm{BIC} - \mathcal{H}(W) - \mathcal{H}(U).$$

- $\mathcal{H}(W)$: entropy of the Gaussian variational distribution of W, $\frac{1}{2}\sum_{i,k}\log\sigma^2_{ik} + \frac{nq}{2}\{1 + \log(2\pi)\}$, where $\sigma^2 = S$ for the ZIP-PLN-PCA model (`S` is a variance) and $\sigma^2 = S^2$ for the PLN-PCA model (`S` is a standard deviation). Zero when `q = 0`.
- $\mathcal{H}(U)$: entropy of the Bernoulli variational distribution of U, $-\sum_{ij}\{\xi_{ij}\log\xi_{ij} + (1-\xi_{ij})\log(1-\xi_{ij})\}$ (ZIP-PLN-PCA model only).

Usage

```
ICL(Y, X, fit, q, model = c("ZIPLNPCA", "PLNPCA"))
```

Arguments

Y	Numeric n x p count matrix (NAs allowed).
X	Design matrix, (n*p) x d (or n x d for a rowwise PLN-PCA design).
fit	Fitted object containing $elbo (output of Miss.ZIPLNPCA or Miss.PLNPCA).
q	Integer, size of the latent space.
model	"ZIPLNPCA" (default) or "PLNPCA".

Value

A single numeric value, the ICL score.

See Also

BIC

Init *Initialize parameters for a PLN-PCA with missing data*

Description

Provides a quick, SVD-based initialization of model and variational parameters from a (possibly sparse) count matrix Y and a design matrix X. The routine fits a (log-transformed) linear model to obtain regression coefficients and uses the SVD of residuals to initialize the latent structure (loadings and variational means).

Usage

```
Init(Y, X, q)
```

Arguments

Y	Numeric n x p count matrix (may contain NA).
X	Numeric design matrix of covariates. Either n x d (rowwise design) or (n*p) x d (vectorized design matching MatrixToVector(Y)).
q	Integer, target latent dimension (rank) for the factor structure.

Details

Missing values in Y are handled via na.exclude in the linear model; residuals at missing entries are then set to zero before the SVD.

Value

A list with elements:

- `B` Matrix of regression coefficients (dimensions follow the chosen design; typically `(1 x d)` in the rowwise case).
- `C` Loadings matrix (`p x q`) for the latent factors.
- `M` Variational means of the latent factors (`n x q`).
- `S` Variational standard deviations (initialized to a small constant; `n x q`).

See Also

`MatrixToVector`, `VectorToMatrix`

Examples

```
set.seed(1)
n <- 30; p <- 8; d <- 2; q <- 2
Y <- matrix(rpois(n * p, 2), n, p)
X <- cbind(1, rnorm(n))          # rowwise design
init <- Init(Y, X, q)
str(init)
```

`Init_ZIP` *Initialize parameters for the ZIP case (zero-inflated)*

Description

SVD-based initialization of parameters for a zero-inflated Poisson log-normal latent factor model (ZIP-PLN). Regression coefficients for the abundance part (`B`) are obtained by regressing `log(1+vec(Y))` on `X`; zero-inflation coefficients (`D`) come from a logistic regression of `I(Y>0)` on `X`. The latent structure is initialized from the SVD of the residual matrix.

Usage

```
Init_ZIP(Y, X, q)
```

Arguments

`Y`	Numeric `n x p` count matrix (may contain `NA`).
`X`	Numeric design matrix with `n*p` rows and `d` columns, aligned with `vec(Y)` (column-wise vectorization).
`q`	Integer, target latent dimension (rank).

Value

A list with elements:

`B` Abundance (Poisson) regression coefficients (1 x d).

`D` Zero-inflation (logit) regression coefficients (1 x d).

`C` Loadings matrix (p x q).

`M` Variational means of latent factors (n x q).

`S` Variational scale parameters.

See Also

`Init` for the non-ZI initializer; `MatrixToVector`, `VectorToMatrix`

Examples

```
set.seed(1)
n <- 30; p <- 10; d <- 3; q <- 2
Y <- matrix(rpois(n*p, 2), n, p)
X <- cbind(1, rnorm(n*p), rnorm(n*p))  # (n*p) x d, vectorized design
init <- Init_ZIP(Y, X, q)
str(init)
```

`Init_ZIP_q0` *Init_ZIP_q0*

Description

Parameters initialisation in the zero inflated case without the latent factors

Usage

```
Init_ZIP_q0(Y, X, q)
```

Arguments

`Y`	count matrix
`X`	covariates
`q`	size of the latent space

Value

A list with elements:

B Matrix of Poisson regression coefficients (1 x d).

D Matrix of logistic regression coefficients (1 x d).

`Miss.PLNPCA` *PLN-PCA with missing data (variational fit + imputation)*

Description

Fits a Poisson log-normal PCA model (PLN-PCA) to a count matrix with missing values, using a variational objective optimized by NLOpt-based routines. The function estimates model/variational parameters and returns expected counts to impute missing entries.

Usage

```
Miss.PLNPCA(Y, X, q, O = NULL, w = NULL, params = NULL, config = NULL)
```

Arguments

`Y`	Numeric `n x p` count matrix. May contain `NA`.
`X`	Numeric design matrix of covariates: either `n x d` (rowwise) or `(n*p) x d` (vectorized to match `MatrixToVector(Y)`).
`q`	Integer, latent rank (dimension of the latent space).
`O`	Optional numeric `n x p` matrix of offsets (default: zeros).
`w`	Optional numeric vector of length `n` with observation weights (default: all ones).
`params`	Optional initial parameters (list) typically produced by `Init`; if `NULL`, they are initialized internally.
`config`	Optional optimizer configuration; if `NULL`, defaults to `PLNPCA_param()$config_optim`.

Details

Two designs for `X` are supported:

1. **Rowwise design** (`nrow(X) == n`): one row per sample.
2. **Vectorized design** (`nrow(X) == n*p`): `X` aligns with `vec(Y)` (vectorization by columns).

Value

A list with components:

`mStep` List with `beta` (`1 x d`) and `C` (`p x q`).

`eStep` List with `M` (`n x q`) and `S` (`n x q`).

`pred` List with `A` and `predicted`, both equal to the `n x p` matrix of expected counts used for imputation:

$$A = \exp\big(O + XB + MC^\top + 0.5\,(S \odot S)\,(C \odot C)^\top\big),$$

where $\odot$ denotes the Hadamard product and `X B` is reshaped to `n x p` when `X` is vectorized.

`iter` Integer, number of iterations.

`elboPath` Numeric vector of objective (ELBO) values over iterations.

`elbo` Final ELBO value.

`params.init` The initial parameters used.

`monitoring` Optimizer log/diagnostics as returned by the backend.

See Also

Init, PLNPCA_param MatrixToVector, VectorToMatrix, Elbo_grad

Examples

```
set.seed(1)
n <- 30; p <- 5; q <- 2
Y <- matrix(rpois(n * p, 2), n, p)
Y[sample(n * p, 20)] <- NA                    # missing entries
X <- cbind(1, rnorm(n))                       # rowwise design, n x d

fit <- Miss.PLNPCA(Y = Y, X = X, q = q)
fit$elbo
str(fit$pred$A)
```

Miss.ZIPLNPCA *ZI-PLN-PCA with missing data (variational fit + imputation)*

Description

Fits a zero-inflated Poisson log-normal (ZI-PLN-PCA) latent factor model to a count matrix with missing values. The routine optimizes a variational objective with NLOpt backends and returns parameter estimates, variational parameters and an imputed matrix.

Usage

```
Miss.ZIPLNPCA(Y, X, q, params = NULL, config = NULL, tolS = NULL, tolxi = NULL)
```

Arguments

Y	Numeric n x p count matrix. May contain NA.
X	Numeric design matrix with n*p rows and d columns, aligned with vec(Y) (column-wise vectorization).
q	Integer, latent rank (dimension of the latent space).
params	Optional list of initial parameters. If NULL, they are initialized internally via Init_ZIP_q0 (when q = 0) or Init_ZIP (when q > 0).
config	Optional list of optimizer controls. If NULL, a default configuration is used (NLOpt backend, MMA algorithm, sensible tolerances).
tolS	List with numeric bounds for the variational scales S: elements lower and upper. Defaults to list(lower = 0, upper = 1).
tolxi	Numeric tolerance for the Jaakkola-type ξ updates in the logistic bound (default 1e-4).

Details

This function supports latent rank q >= 0. When q = 0, it reduces to a ZIP regression (no latent factors). When q > 0, it fits a ZIP-PLN factor model.

Internally, a binary mask R = 1_{observed}(Y) is created; a copied matrix Y.na sets missing entries to zero for objective evaluation. The optimizer bounds for S are taken from tolS. Ensure that the vectorization and parameter stacking used in the optimizer are consistent with the shapes listed above.

Value

A list with components:

mStep List of model parameters. For q = 0: gamma (d x 1) and beta (d x 1). For q > 0: same plus the loading matrix C (p x q).

eStep List of variational parameters. For q = 0: only xi (n x p). For q > 0: M (n x q), S (n x q) and xi (n x p).

pred List with predictors and expected counts: mu (n x p, abundance mean), nu (n x p, zero-inflation mean), A (n x p, PLN expectation), predicted (n x p, ZIP expectation for all entries: $\xi \odot A$, with $A = \exp(XB + MC^\top + 0.5\, S\, (C \odot C)^\top)$, S being the variational variances).

imputed n x p matrix equal to predicted at missing entries of Y, and Y elsewhere (ZIP expectation).

iter Integer, number of iterations.

elboPath Numeric vector of objective (ELBO) values over iterations.

elbo Final ELBO value.

params.init Parameters as recorded from the backend call.

monitoring Optimizer diagnostics/logs from the backend.

gradB, gradD, gradC, gradM, gradS Gradients of the ELBO with respect to the corresponding parameters, obtained via Elbo_grad.

See Also

Init_ZIP, Init_ZIP_q0, Elbo_grad, VectorToMatrix

Examples

```
set.seed(1)
n <- 30; p <- 5; d <- 2; q <- 2
X <- cbind(1, rnorm(n * p))                      # (n*p) x d design
theta <- list(B = matrix(c(1, 0.5)), D = matrix(c(1, -0.5)),
              C = matrix(rnorm(p * q, sd = 0.5), p, q))
Y <- Simul(X, theta, list(n = n, p = p, d = d, q = q))$Y
Y[sample(n * p, 20)] <- NA                       # missing entries

fit <- Miss.ZIPLNPCA(Y, X, q)
str(fit$mStep)
fit$imputed[is.na(Y)][1:5]                       # imputed values
```

`Miss.ZIPLNPCA.logS` *ZI-PLN-PCA with missing data using log(S) parametrization*

Description

Fits a zero-inflated Poisson log-normal (ZI-PLN-PCA) latent factor model to a count matrix with missing values, using a variational objective and an **alternative parametrization** on the variational scales:

`log(S)` instead of S.

This guarantees positivity of \(S\) and allows simple box constraints on \(`log(S)` \) via `tolLogS`.

Usage

```
Miss.ZIPLNPCA.logS(
  Y,
  X,
  q,
  params = NULL,
  config = NULL,
  tolxi = NULL,
  tolLogS = NULL
)
```

Arguments

`Y`	Numeric `n` x `p` count matrix. May contain `NA`.
`X`	Numeric design matrix with `n*p` rows and `d` columns, aligned with `vec(Y)` (vectorization by columns).
`q`	Integer, latent rank (dimension of the latent space).
`params`	Optional list of initial parameters. If `NULL`, they are initialized by `Init_ZIP`. If provided, it must include a positive `S`; the function internally adds `logS <- log(S)`.
`config`	Optional list of optimizer controls. If `NULL`, defaults to `PLNPCA_param()$config_optim`.
`tolxi`	Numeric tolerance for the Jaakkola-type ξ updates in the logistic bound (default `1e-4`).
`tolLogS`	Numeric upper bound applied to `logS` (box constraint). Defaults to `Inf`. Use this to avoid excessively large variational scales when needed.

Details

Missing entries are handled by a mask `R = 1_{observed}(Y)`; a working matrix `Y.na` sets missings to 0 for the objective evaluation. Box constraints are set on `logS` via `config$upper_bounds`, using `tolLogS`. Make sure `X` matches `vec(Y)`.

Value

A list with components:

`mStep` Model parameters: `gamma` (d x 1), `beta` (d x 1), and loadings `C` (p x q).

`eStep` Variational parameters: means `M` (n x q), scales `S` (n x q), and logistic bound parameters `xi` (n x p).

`pred` List with predictors and expected counts: `mu` (n x p), `nu` (n x p), backend Poisson mean `A` (n x p), and `predicted` (n x p, ZIP expectation for all entries: $\xi \odot A$, with $A = \exp(XB + MC^\top + 0.5\, S\,(C \odot C)^\top)$, `S` being the variational variances).

`imputed` n x p matrix equal to `predicted` at missing entries of `Y`, and `Y` elsewhere.

`iter` Number of iterations; `elboPath` (trajectory of ELBO); `elbo` (final ELBO).

`params.init` Initial parameters passed to the backend (with `logS` added).

`monitoring` Optimizer diagnostics/logs.

`gradB, gradD, gradC, gradM, gradS` ELBO gradients w.r.t. corresponding parameters (via `Elbo_grad`).

See Also

`Miss.ZIPLNPCA` for the `S` parametrization, `Init_ZIP`, `Elbo_grad`

Examples

```
set.seed(1)
n <- 30; p <- 5; d <- 2; q <- 2
X <- cbind(1, rnorm(n * p))                       # (n*p) x d design
theta <- list(B = matrix(c(1, 0.5)), D = matrix(c(1, -0.5)),
              C = matrix(rnorm(p * q, sd = 0.5), p, q))
Y <- Simul(X, theta, list(n = n, p = p, d = d, q = q))$Y
Y[sample(n * p, 20)] <- NA                        # missing entries

fit <- Miss.ZIPLNPCA.logS(Y, X, q, tolLogS = 2)   # cap logS
fit$elbo
str(fit$eStep$S)
```

`Miss.ZIPLNPCA_Steps` *ZI-PLN-PCA (missing data) with parameterwise optimization steps*

Description

Fits a zero-inflated Poisson log-normal (ZI-PLN-PCA) latent factor model to a count matrix with missing values using a **blockwise** optimization strategy. At each outer iteration, parameters are updated by activating one block among {B, D, C, M, S} while keeping the others fixed, via an NLOpt backend. The loop stops when both parameter and ELBO changes are below thresholds or when `maxiter` is reached.

Usage

```
Miss.ZIPLNPCA_Steps(Y, X, q, params = NULL, config_vem = NULL, config = NULL)
```

Arguments

`Y`	Numeric `n x p` count matrix. May contain `NA`.
`X`	Numeric design matrix with `n*p` rows and `d` columns, aligned with `vec(Y)` (column-wise vectorization).
`q`	Integer, latent rank (dimension of the latent space).
`params`	Optional list of initial parameters. If `NULL`, `Init_ZIP` is used.
`config_vem`	List of controls for the outer blockwise loop (variational EM-like), with fields:
	`maxiter` Maximum number of outer iterations (default `1e6`).
	`ftol` ELBO tolerance; stop if `|ELBO_{t}-ELBO_{t-1}| <= ftol`.
	`xtol` Parameter tolerance; stop if `max|theta_{t}-theta_{t-1}| <= xtol`.
	`tolS` List with `lower`, `upper` bounds for `S`.
	`tolxi` Tolerance used in the Jaakkola-type ξ updates for the logistic bound.
`config`	List of controls for the NLOpt backend (per-block inner solve), e.g. `algorithm`, `maxeval`, `ftol_abs`, `xtol_abs`, `ftol_rel`, `xtol_rel`, `trace`, `maxtime`, as well as `upper_bounds`, `lower_bounds` (filled here from `tolS` and dimensions).

Details

Missing entries are handled through a binary mask and a working copy of `Y` with zeros at missing locations for the objective evaluation.

A binary mask `R = 1_{observed}(Y)` is built; a working matrix `Y.na` replaces missings by 0 for the objective. Box constraints for `S` are set using `config_vem$tolS` and injected into `config$upper_bounds` and `config$lower_bounds`. Each block update calls the backend `nlopt_optimize_ZIP_Steps(data, params, config, tolxi, active_blocks)` with a single active block at a time.

Predicted mean: in this implementation, the Poisson mean used for the R-side recomputation is

$$\exp\big(\mu + MC^\top + 0.5\, S\,(C \odot C)^\top\big),$$

where $\mu = XB$ reshaped into `n x p`. Note that other functions in le package emploient $0.5\,(S \odot S)\,(C \odot C)^\top$. Vérifie que la paramétrisation voulue est cohérente avec le backend.

Value

A list with components:

`mStep` Model parameters: `gamma` (`d x 1`), `beta` (`d x 1`), loadings `C` (`p x q`).

`eStep` Variational parameters: `M` (`n x q`), `S` (`n x q`), logistic bound parameters `xi` (`n x p`).

`pred` List with predictors and expected counts: `mu` (`n x p`), `nu` (`n x p`), backend mean `A` (`n x p`), and `predicted` (`n x p`, ZIP expectation for all entries: $\xi \odot A$, with $A = \exp(XB + MC^\top + 0.5\, S\,(C \odot C)^\top)$, `S` being the variational variances).

`imputed` `n x p` matrix equal to `predicted` at missing entries of `Y`, and `Y` elsewhere.

`iter` Number of outer iterations performed.

`elboPath` Numeric vector of ELBO values across outer iterations.

`elbo` Final ELBO value.

`params.init` Parameters used for initialization.

`monitoring` List with `status` (stopping reason) and `iterations`.

`gradB, gradD, gradC, gradM, gradS` ELBO gradients w.r.t. each parameter block (from `Elbo_grad`) at the solution.

See Also

`Miss.ZIPLNPCA`, `Miss.ZIPLNPCA.logS`, `Init_ZIP`, `Elbo_grad`

Examples

```
set.seed(1)
n <- 30; p <- 5; d <- 2; q <- 2
X <- cbind(1, rnorm(n * p))                      # (n*p) x d design
theta <- list(B = matrix(c(1, 0.5)), D = matrix(c(1, -0.5)),
              C = matrix(rnorm(p * q, sd = 0.5), p, q))
Y <- Simul(X, theta, list(n = n, p = p, d = d, q = q))$Y
Y[sample(n * p, 20)] <- NA                       # missing entries

fit <- Miss.ZIPLNPCA_Steps(Y, X, q)
fit$elbo
plot(fit$elboPath, type = "l", xlab = "Outer iteration", ylab = "ELBO")
```

`Miss.ZIPLNPCA_VE` *ZI-PLN-PCA (missing data) — variational E-step solver*

Description

Fits a zero-inflated Poisson log-normal (ZI-PLN-PCA) latent factor model on a count matrix with missing values using a single call to a variational optimization backend (VE). Returns model/variational parameters and expected Poisson means suitable for imputation and prediction.

Usage

```
Miss.ZIPLNPCA_VE(Y, X, q, params = NULL, config_vem = NULL, config = NULL)
```

Arguments

`Y`	Numeric `n` x `p` count matrix. May contain `NA`.
`X`	Numeric design matrix with `n*p` rows and `d` columns, aligned with `vec(Y)` (vectorization by columns).
`q`	Integer, latent rank (dimension of the latent space).

`params` Optional list of initial parameters. If `NULL`, `Init_ZIP` is called.

`config_vem` List of outer VE controls (stopping and bounds), with fields:

- `maxiter` Maximum iterations (not always used by this backend, default `10000`).
- `ftol` ELBO tolerance (default `1e-08`).
- `xtol` Parameter tolerance (default `1e-04`).
- `tolS` List with `lower`, `upper` bounds for `S` (default `list(lower = 1e-4, upper = 1)`).
- `tolxi` Tolerance for the Jaakkola-type ξ updates (default `1e-04`).

`config` List of NLOpt controls for the backend (algorithm, tolerances, `maxeval`, etc.). If `NULL`, sensible defaults are provided.

Details

Missing entries are handled via a binary mask `R = 1_{observed}(Y)`, and a working copy `Y.na` with zeros at missing locations for the objective evaluation. Box constraints on `S` are injected into `config$lower_bounds` and `config$upper_bounds` using `config_vem$tolS`.

Predicted mean: ici, la recomposition côté R utilise

$$\exp\big(XB + MC^\top + 0.5\,S\,(C \odot C)^\top\big).$$

D'autres fonctions du package emploient $0.5\,(S \odot S)\,(C \odot C)^\top$. Vérifie et harmonise selon la paramétrisation attendue par ton backend.

Value

A list with components:

`mStep` Model parameters: `gamma` (`d x 1`), `beta` (`d x 1`), loadings `C` (`p x q`).

`eStep` Variational parameters: `M` (`n x q`), `S` (`n x q`), `xi` (`n x p`).

`pred` List with predictors and expected counts: `mu` (`n x p`), `nu` (`n x p`), backend mean `A` (`n x p`), and `predicted` (`n x p`, ZIP expectation for all entries: $\xi \odot A$, with $A = \exp(XB + MC^\top + 0.5\,S\,(C \odot C)^\top)$, `S` being the variational variances).

`elbo` Final ELBO value.

`params.init` Initial parameters used.

`monitoring` Backend diagnostics (if provided).

`elbo1, elbo2, elbo3, elbo4, elbo5` Decomposition terms of the ELBO (as returned by the backend).

Examples

```
set.seed(1)
n <- 30; p <- 5; d <- 2; q <- 2
X <- cbind(1, rnorm(n * p))                       # (n*p) x d design
theta <- list(B = matrix(c(1, 0.5)), D = matrix(c(1, -0.5)),
              C = matrix(rnorm(p * q, sd = 0.5), p, q))
Y <- Simul(X, theta, list(n = n, p = p, d = d, q = q))$Y
Y[sample(n * p, 20)] <- NA                        # missing entries
```

```
fit <- Miss.ZIPLNPCA_VE(Y, X, q)
fit$elbo
str(fit$eStep)
```

`Predictions` *Predictive simulations and intervals*

Description

Draws parameter samples using the sandwich variance `V_theta`, simulates predictive counts with zero-inflation and returns per-cell prediction intervals. Two types of intervals are available: conditional (latent variables drawn from their variational posterior, after one VE step per draw) and marginal (latent variables drawn from their prior).

Usage

```
Predictions(
  Y,
  X,
  fit,
  MC,
  type = c("conditional", "marginal"),
  probs = c(0, 0.9),
  output = "intervals",
  cl = NULL,
  verbose = TRUE
)
```

Arguments

`Y`	An n x p count matrix (NAs allowed).
`X`	An (n*p) x d design matrix (stacked by blocks of size n).
`fit`	Estimator output list (from `Miss.ZIPLNPCA`).
`MC`	Integer, number of Monte Carlo draws.
`type`	`"conditional"` (default) or `"marginal"`.
	conditional for each parameter draw, a VE step gives M, S and ξ; then $U \sim \mathcal{B}(\xi)$ and $Z_i \sim \mathcal{N}(X\beta + M_i C^\top, C\,\mathrm{diag}(S_i)\,C^\top)$.
	marginal no VE step; $U \sim \mathcal{B}(\mathrm{logit}^{-1}(X\gamma))$ and $Z_i \sim \mathcal{N}(X\beta, CC^\top)$.
	In both cases $U = 1$ means presence and the prediction is $U \times \mathcal{P}(\exp(Z))$.
`probs`	Numeric vector of length 2, the lower and upper quantile levels of the interval. Default `c(0, 0.9)`.
`output`	Character vector, what to return besides the intervals. One or several of `"intervals"` (always returned), `"draws"`, `"latent"`, `"params"`, or `"all"`.

`cl`	Parallelisation of the VE steps (conditional type only), passed to `pblapply`. `NULL` (default): sequential. An integer: number of forked processes (not available on Windows). A cluster from `makeCluster`: works on all platforms. All random draws are made in the main process, so results for a given seed do not depend on `cl`.
`verbose`	Logical. If `TRUE` (default), prints the current step and shows progress bars for the VE steps and the simulation of the latent variables. Progress bars can also be controlled globally with `pboptions`.

Value

A list with elements:

- `lower`, `upper`: n x p matrices of predictive quantiles at levels `probs`.
- `draws` (if requested): list with `count` (Poisson draws) and `presence` (Bernoulli draws U), arrays n x p x MC. The prediction is `count * presence`.
- `latent` (if requested): list with `Z`, array n x p x MC of Gaussian latent draws, and for the conditional type `M`, `S` (arrays n x q x MC) and `xi` (array n x p x MC) from the VE steps.
- `params` (if requested): list with `gamma`, `beta` (matrices MC x d) and `C` (matrix MC x (p*q), column-major vectorisation of C).
- `settings`: list with `type`, `probs` and `MC`.

See Also

`V_theta`, `Miss.ZIPLNPCA_VE`

Examples

```
set.seed(1)
n <- 30; p <- 5; d <- 2; q <- 2
X <- cbind(1, rnorm(n * p))                       # (n*p) x d design
theta <- list(B = matrix(c(1, 0.5)), D = matrix(c(1, -0.5)),
              C = matrix(rnorm(p * q, sd = 0.5), p, q))
Y <- Simul(X, theta, list(n = n, p = p, d = d, q = q))$Y
Y[sample(n * p, 20)] <- NA                        # missing entries
fit <- Miss.ZIPLNPCA(Y, X, q)


# Conditional intervals (default), 90% upper bound
pi_cond <- Predictions(Y, X, fit, MC = 20)
pi_cond$upper[1:3, ]

# Marginal intervals at other levels, with the draws
pi_marg <- Predictions(Y, X, fit, MC = 20, type = "marginal",
                       probs = c(0.05, 0.95), output = "draws")
dim(pi_marg$draws$count)
```

`select_model_bic` *Model selection over candidate latent dimensions (BIC or ICL)*

Description

For each value in `latents`, fits a model with `fun` and computes the chosen criterion (`BIC` or `ICL`). Returns the criterion values, the latent dimension that maximizes the criterion and the corresponding fit.

Usage

```
select_model_bic(
  Y,
  X,
  fun,
  latents,
  model = c("ZIPLNPCA", "PLNPCA"),
  criterion = c("BIC", "ICL")
)
```

Arguments

`Y`	Numeric `n` x `p` count matrix (NAs allowed).
`X`	Design matrix, `(n*p)` x `d`.
`fun`	Fitting function, called as `fun(Y = Y, X = X, q = q)` (e.g. `Miss.ZIPLNPCA` or `Miss.PLNPCA`).
`latents`	Integer vector of candidate latent dimensions.
`model`	`"ZIPLNPCA"` (default) or `"PLNPCA"`; must match `fun`. Passed to `BIC` and `ICL`.
`criterion`	`"BIC"` (default) or `"ICL"`.

Details

A progress bar is displayed via **pbapply**. The function checks that `model` matches the fits returned by `fun`: a ZIP-PLN-PCA fit contains `mStep$gamma`, a PLN-PCA fit does not.

Value

A list with elements:

- `criterion`: the criterion used.
- `values`: numeric vector of criterion values, named by `latents`.
- `q_star`: the latent dimension maximizing the criterion.
- `fit_star`: the fit obtained with `q_star`.

See Also

BIC, ICL

Examples

```
set.seed(1)
n <- 30; p <- 5; d <- 2; q <- 2
X <- cbind(1, rnorm(n * p))                      # (n*p) x d design
theta <- list(B = matrix(c(1, 0.5)), D = matrix(c(1, -0.5)),
              C = matrix(rnorm(p * q, sd = 0.5), p, q))
Y <- Simul(X, theta, list(n = n, p = p, d = d, q = q))$Y
Y[sample(n * p, 20)] <- NA                       # missing entries


sel <- select_model_bic(Y, X, Miss.ZIPLNPCA, latents = 1:3)
sel$values
sel$q_star
fit <- sel$fit_star

sel_icl <- select_model_bic(Y, X, Miss.ZIPLNPCA, latents = 1:3,
                            criterion = "ICL")
sel_icl$q_star
```

Simul *Simulate zero-inflated Poisson log-normal PCA data*

Description

Generates simulated observations Y under a ZIP-LN-PCA structure from parameters in theta and dimensions in dim.

Usage

```
Simul(X, theta, dim)
```

Arguments

X	An $(n \times p) \cdot d$ design matrix stacked so that VectorToMatrix(X %*% B, n, p) produces the mean matrix mu.
theta	List of model parameters with components: • B: coefficient matrix (d x 1 or compatible) • D: zero-inflation coefficient matrix (d x 1 or compatible) • C: loading matrix (p x q)
dim	List of dimensions with entries n, p, d, q.

Details

Relies on `VectorToMatrix()` being available in the package.

Value

A list with elements:

- `Y`: simulated counts (n x p)
- `W`: latent scores (n x q)
- `U`: zero-inflation indicators (n x p)
- `Z`: Poisson draws before zero-inflation (n x p)
- `Lambda`: Poisson intensities (n x p)
- `mu`: mean matrix (n x p)

Examples

```
set.seed(1)
n <- 30; p <- 5; d <- 2; q <- 2
X <- cbind(1, rnorm(n * p))                      # (n*p) x d design
theta <- list(B = matrix(c(1, 0.5)), D = matrix(c(1, -0.5)),
              C = matrix(rnorm(p * q, sd = 0.5), p, q))
sim <- Simul(X, theta, list(n = n, p = p, d = d, q = q))
str(sim)
mean(sim$Y == 0)                                 # proportion of zeros
```

`V_theta` *Variance-covariance matrix of theta*

Description

Computes $V(\hat{\theta})$ using a sandwich (robust) variance formula.

Usage

```
V_theta(Y, X, fit, ginv_tol = 1e-12, verbose = TRUE)
```

Arguments

`Y`	An n x p matrix of counts.
`X`	An (n*p) x d design matrix, stacked by blocks of size n.
`fit`	Output list returned by the estimator.
`ginv_tol`	Numeric tolerance used by MASS::ginv() when a pseudo-inverse is needed.
`verbose`	Logical; if TRUE, prints diagnostic messages when solve() fails or is unstable.

Value

A symmetric variance-covariance matrix for theta.

`waterbird` *Waterbird count data*

Description

Counts of a waterbird species from a multi-year, multi-site monitoring program. The data are incomplete: many site-year combinations were not surveyed and are recorded as missing values. Counts also contain a large proportion of zeros, which makes the data a suitable example for the zero-inflated Poisson log-normal model implemented in the package. Sites and years are anonymized.

Usage

```
data(waterbird)
```

Format

A named list with two elements:

Y Numeric count matrix with 419 rows (sites, `site_001` to `site_419`) and 17 columns (years, `year_01` to `year_17`). Unobserved site-year combinations are `NA`.

X Numeric design matrix with $419 \times 17 = 7123$ rows (site-year combinations, stacked year by year, i.e. by blocks of 419 sites) and 7 columns: `intercept` (column of ones), and the standardized covariates `dist_coast` (distance to the coast), `dist_city` (distance to the nearest city), `elevation`, `year` (linear time trend), `spring_temp_anomaly` (spring temperature anomaly) and `agricultural_land` (percentage of agricultural land).

Source

Anonymized data from a waterbird monitoring program.

Examples

```
data(waterbird, package = "colvR")
str(waterbird, max.level = 1)
mean(is.na(waterbird$Y))               # proportion of missing entries
mean(waterbird$Y == 0, na.rm = TRUE)   # proportion of zeros
```

Index

∗ **datasets**
waterbird, 23

BIC, 2, *7*, *20, 21*

covmat, 3, *4*

Elbo_grad, 4, *11, 12*, *14*, *16*
Elbo_grad_logS, 5

IC, 6
ICL, *3*, 6, *20, 21*
Init, 7, *9–11*
Init_ZIP, 8, *11–17*
Init_ZIP_q0, 9, *11, 12*

makeCluster, *19*
MatrixToVector, *8, 9*, *11*
Miss.PLNPCA, 10
Miss.ZIPLNPCA, 11, *14*, *16*
Miss.ZIPLNPCA.logS, 13, *16*
Miss.ZIPLNPCA_Steps, 14
Miss.ZIPLNPCA_VE, 16, *19*

pblapply, *19*
pboptions, *19*
PLNPCA_param, *11*
Predictions, 18

select_model_bic, 20
Simul, 21

V_theta, *19*, 22
VectorToMatrix, *8, 9*, *11, 12*

waterbird, 23