Package: pECV (via r-universe)

May 26, 2026

Type Package

Title Entrywise Splitting Cross-Validation for Factor Models

Version 1.0.1

Description Implements entrywise splitting cross-validation (ECV) and its penalized variant (pECV) for selecting the number of factors in generalized factor models.

License GPL-3

Encoding UTF-8

Language en-US

Depends R (>= 3.5.0)

Imports stats, Rcpp (>= 1.0.0), irlba

Suggests mirtjml, testthat (>= 3.0.0)

LinkingTo Rcpp, RcppArmadillo

URL https://github.com/wangATsu/ECV

BugReports https://github.com/wangATsu/ECV/issues

RoxygenNote 7.3.2

Config/testthat/edition 3

ByteCompile true

NeedsCompilation yes

Author Zhijing Wang [aut, cre]

Maintainer Zhijing Wang <wangzhijing@sjtu.edu.cn>

Repository https://wzjwangzhijing.r-universe.dev

Date/Publication 2025-08-28 08:50:07 UTC

RemoteUrl https://github.com/cran/pECV

RemoteRef HEAD

RemoteSha 9c207a078afb96719ab5eb405751fd1179d3aca4

Contents

`estimate_C` *Estimate constraint constant C for continuous data*

Description

Data-driven estimation of the constraint constant C in alternating maximization algorithm for continuous data using truncated SVD approach. This function decomposes the data matrix and estimates C based on the maximum row norms.

Usage

```
estimate_C(X, qmax = 8, safety = 1.2)
```

Arguments

`X`	n x p continuous data matrix
`qmax`	Rank for truncated SVD (default 8)
`safety`	Safety parameter for conservative estimation (default 1.2)

Details

The function performs the following steps: 1. Computes truncated SVD of X with rank qmax 2. Constructs factor matrices A = U * sqrt(D) and B = V * sqrt(D) 3. Calculates row 2-norms for matrices A and B 4. Takes the maximum norm and multiplies by safety parameter

For count data, it is recommended to transform the data using log(X + 1) before applying this function.

Value

A list containing:

qmax	Truncation rank used
safety	Safety parameter applied
C_norm_hat	Original maximum row norm
C_est	Final conservative estimate of C
a_norms	Row norms of factor matrix A
b_norms	Row norms of factor matrix B

Examples

```
# Example 1: Continuous data
set.seed(123)
n <- 100; p <- 50; q <- 3
theta_true <- matrix(runif(n * q), n, q)
A_true <- matrix(runif(p * q), p, q)
X <- theta_true %*% t(A_true) + matrix(rnorm(n * p, sd = 0.5), n, p)

# Estimate C
C_result <- estimate_C(X, qmax = 5)
print(C_result$C_est)

# Example 2: Count data (apply log transformation)
lambda <- exp(theta_true %*% t(A_true))
X_count <- matrix(rpois(n * p, lambda = as.vector(lambda)), n, p)
X_transformed <- log(X_count + 1)
C_count <- estimate_C(X_transformed, qmax = 5)
print(C_count$C_est)
```

---

estimate_C_binary *Estimate constraint constant C for binary data*

Description

Data-driven estimation of the constraint constant C for binary data using cross-window smoothing and empirical logit transformation.

Usage

```
estimate_C_binary(X, qmax = 8, safety = 1.5, eps = 1e-12, radius = 1)
```

Arguments

X	n x p binary data matrix (0/1 values)
qmax	Rank for truncated SVD (default 8)
safety	Safety parameter for conservative estimation (default 1.5)
eps	Small constant to avoid logit divergence when p=0 or p=1 (default 1e-12)
radius	Radius for cross-window smoothing (default 1)

Details

The function performs the following steps: 1. Applies cross-window smoothing to estimate probabilities 2. Performs empirical logit transformation with smoothing 3. Computes truncated SVD of the transformed matrix 4. Constructs matrices A and B and calculates row norms 5. Estimates C as the maximum norm times safety parameter

The cross-window smoothing helps stabilize probability estimates, especially for sparse binary data.

Value

A list containing:

`radius`	Cross-window radius used
`qmax`	Truncation rank used
`safety`	Safety parameter applied
`C0`	Original maximum row norm
`C_est`	Final conservative estimate of C
`a_norms`	Row norms of factor matrix A
`b_norms`	Row norms of factor matrix B
`Mhat`	Logit-transformed matrix
`P_smooth`	Smoothed probability matrix
`N_counts`	Count of values in each smoothing window

`generate_binary_data` *Generate binary data example*

Description

Generate simulated data from a binary (logistic) factor model.

Usage

```
generate_binary_data(n = 100, p = 50, q = 3)
```

Arguments

`n`	Integer. Number of observations.
`p`	Integer. Number of variables.
`q`	Integer. True number of latent factors.

Value

A named list with components:

resp Binary matrix (n x p). Generated 0/1 responses.

true_q Integer. True number of factors used in simulation.

theta_true Numeric matrix (n x q). True latent factor scores.

A_true Numeric matrix (p x q). True factor loadings.

d_true Numeric vector (length p). Item intercepts.

---

generate_binary_data_miss

Generate binary data with missing values

Description

Generate simulated data from a binary (logistic) factor model with missing values.

Usage

```
generate_binary_data_miss(n = 100, p = 50, q = 3, miss_prop = 0.05)
```

Arguments

n	Integer. Number of observations.
p	Integer. Number of variables.
q	Integer. True number of latent factors.
miss_prop	Numeric in (0,1). Proportion of missing values (default 0.05).

Value

A named list with components:

resp Binary matrix (n x p). Generated 0/1 responses with missing values (NA).

resp_complete Binary matrix (n x p). Complete data before missingness.

true_q Integer. True number of factors used in simulation.

theta_true Numeric matrix. True latent factor scores.

A_true Numeric matrix. True factor loadings.

d_true Numeric vector (length p). Item intercepts.

miss_prop Numeric. Proportion of entries set to missing.

generate_continuous_data

Generate continuous data example

Description

Generate simulated data from a Gaussian factor model.

Usage

```
generate_continuous_data(n = 100, p = 50, q = 3, noise_sd = 1)
```

Arguments

n	Integer. Number of observations.
p	Integer. Number of variables.
q	Integer. True number of latent factors.
noise_sd	Numeric. Standard deviation of Gaussian noise.

Value

A named list with components:

resp Numeric matrix (n x p). Generated observed data.

true_q Integer. True number of factors used in simulation.

theta_true Numeric matrix (n x (q+1)). True latent factor scores with intercept.

A_true Numeric matrix (p x (q+1)). True factor loadings.

generate_continuous_data_miss

Generate continuous data with missing values

Description

Generate simulated data from a Gaussian factor model with missing values.

Usage

```
generate_continuous_data_miss(
  n = 100,
  p = 50,
  q = 3,
  noise_sd = 1,
  miss_prop = 0.05
)
```

Arguments

n	Integer. Number of observations.
p	Integer. Number of variables.
q	Integer. True number of latent factors.
noise_sd	Numeric. Standard deviation of Gaussian noise.
miss_prop	Numeric in (0,1). Proportion of missing values (default 0.05).

Value

A named list with components:

resp Numeric matrix (n x p). Generated data with missing values (NA).

resp_complete Numeric matrix (n x p). Complete data before missingness.

true_q Integer. True number of factors used in simulation.

theta_true Numeric matrix (n x (q+1)). True latent factor scores with intercept.

A_true Numeric matrix (p x (q+1)). True factor loadings.

miss_prop Numeric. Proportion of entries set to missing.

---

generate_count_data *Generate count data example*

Description

Generate simulated data from a Poisson factor model.

Usage

```
generate_count_data(n = 100, p = 50, q = 3)
```

Arguments

n	Integer. Number of observations.
p	Integer. Number of variables.
q	Integer. True number of latent factors.

Value

A named list with components:

resp Integer matrix (n x p). Generated Poisson observations.

true_q Integer. True number of factors used in simulation.

theta_true Numeric matrix (n x (q+1)). True latent factor scores with intercept.

A_true Numeric matrix (p x (q+1)). True factor loadings.

`generate_count_data_miss` *Generate count data with missing values*

Description

Generate simulated data from a Poisson factor model with missing values.

Usage

```
generate_count_data_miss(n = 100, p = 50, q = 3, miss_prop = 0.05)
```

Arguments

`n`	Integer. Number of observations.
`p`	Integer. Number of variables.
`q`	Integer. True number of latent factors.
`miss_prop`	Numeric in (0,1). Proportion of missing values (default 0.05).

Value

A named list with components:

resp Integer matrix (n x p). Generated data with missing values (NA).

resp_complete Integer matrix (n x p). Complete data before missingness.

true_q Integer. True number of factors used in simulation.

theta_true Numeric matrix (n x (q+1)). True latent factor scores with intercept.

A_true Numeric matrix (p x (q+1)). True factor loadings.

miss_prop Numeric. Proportion of entries set to missing.

`pECV` *Entrywise Splitting Cross-Validation for Factor Models*

Description

Uses (Penalized) Entrywise Splitting Cross-Validation (ECV / pECV) to estimate the number of latent factors in generalized factor models.

Usage

```
pECV(
  resp,
  C = 5,
  qmax = 8,
  fold = 5,
  tol_val = 0.01,
  theta0 = NULL,
  A0 = NULL,
  seed = 1,
  data_type = NULL
)
```

Arguments

resp	Observation data matrix (n x p); can be continuous, count, or binary.
C	Constraint constant, default is 5.
qmax	Maximum number of factors to consider, default is 8.
fold	Number of folds in cross-validation, default is 5.
tol_val	Convergence tolerance, default is 0.01 (interpreted as 0.01 / number of estimated elements).
theta0	Optional initial matrix for factors; sampled from Uniform if not provided.
A0	Optional initial matrix for loadings; sampled from Uniform if not provided.
seed	Random seed, default is 1.
data_type	Data type, one of "continuous", "count", "binary". If not specified, it is auto-detected.

Details

The example below may take more than 5 seconds on some machines and is therefore not run during routine checks.

Value

A named **list** with components:

ECV Integer. Number of factors selected by standard ECV.

p1ECV Integer. Number of factors selected by ECV with penalty 1.

p2ECV Integer. Number of factors selected by ECV with penalty 2.

p3ECV Integer. Number of factors selected by ECV with penalty 3.

p4ECV Integer. Number of factors selected by ECV with penalty 4.

ECV_loss Numeric vector. Cross-validation loss for each candidate factor number (typically of length qmax).

data_type Character. The detected/used data type: "continuous", "count", or "binary".

The return value has base R types (no special S3/S4 class).

Examples

```
set.seed(123)
# Generate count data
n <- 50; p <- 50; q <- 2
theta_true <- cbind(1, matrix(runif(n * q, -2, 2), n, q))
A_true <- matrix(runif(p * (q + 1), -2, 2), p, (q + 1))
lambda <- exp(theta_true %*% t(A_true))
resp <- matrix(
  rpois(length(lambda), lambda = as.vector(lambda)),
  nrow = nrow(lambda), ncol = ncol(lambda)
)
result <- pECV(resp, C = 4, qmax = 4, fold = 5)
print(result)
```

---

pECV.miss *Entrywise Splitting Cross-Validation with Missing Data*

Description

Uses (Penalized) Entrywise Splitting Cross-Validation to estimate the number of latent factors in generalized factor models when the data contain missing values.

Usage

```
pECV.miss(
  resp,
  C = 5,
  qmax = 8,
  fold = 5,
  tol_val = 0.01,
  theta0 = NULL,
  A0 = NULL,
  seed = 1,
  data_type = NULL
)
```

Arguments

resp	Observation data matrix (n x p) with missing values as NA; can be continuous, count, or binary.
C	Constraint constant, default is 5.
qmax	Maximum number of factors to consider, default is 8.
fold	Number of folds in cross-validation, default is 5.
tol_val	Convergence tolerance, default is 0.01 (interpreted as 0.01 / number of estimated elements).

theta0	Optional initial matrix for factors; sampled from Uniform if not provided.
A0	Optional initial matrix for loadings; sampled from Uniform if not provided.
seed	Random seed, default is 1.
data_type	Data type, one of "continuous", "count", "binary". If not specified, it is auto-detected.

Details

The example below may take more than 5 seconds on some machines and is therefore not run during routine checks.

Value

A named **list** with components:

ECV Integer. Number of factors selected by standard ECV.

p1ECV Integer. Number of factors selected by ECV with penalty 1.

p2ECV Integer. Number of factors selected by ECV with penalty 2.

p3ECV Integer. Number of factors selected by ECV with penalty 3.

p4ECV Integer. Number of factors selected by ECV with penalty 4.

ECV_loss Numeric vector. Cross-validation loss for each candidate factor number (typically of length qmax).

data_type Character. The detected/used data type: "continuous", "count", or "binary".

miss_percent Numeric scalar. Percentage of missing entries in resp.

The return value uses base R types (no special S3/S4 class).

Examples

```
set.seed(123)
# Generate count data with missing values
n <- 50; p <- 50; q <- 2
theta_true <- cbind(1, matrix(runif(n * q, -2, 2), n, q))
A_true <- matrix(runif(p * (q + 1), -2, 2), p, (q + 1))
lambda <- exp(theta_true %*% t(A_true))
resp <- matrix(
  rpois(length(lambda), lambda = as.vector(lambda)),
  nrow = nrow(lambda), ncol = ncol(lambda)
)
# Introduce 5% missing values
miss_idx <- sample(1:(n * p), size = 0.05 * n * p)
resp[miss_idx] <- NA
result <- pECV.miss(resp, C = 4, qmax = 4, fold = 5)
print(result)
```

Index