Package 'scmix'

August 20, 2026

Type Package

Title Bayesian Model-Based Clustering with Sparse Conditional Mixture Models

Version 0.1.1

Description Fits Bayesian sparse conditional (Gaussian) mixture models for model-based clustering. Each mixture component factorizes into a chain of univariate polynomial regressions with per-component, per-equation Bayesian variable selection under a centered Zellner g-prior; the number of clusters is selected within a single run via an overfitted sparse mixture (Dirichlet concentration 1/K). The blocked Gibbs sampler draws the selection sets exactly by enumeration (or by validated single-flip Metropolis-Hastings in higher dimension), is provably well-posed under a documented proper fallback prior, and reports a label-invariant consensus partition (Dahl's least-squares criterion). Companion package to Dong, Liao, and Lee (2026), ``Replacing three nested searches with one sweep: a Bayesian treatment of sparse conditional mixture clustering". Multiple-imputation functionality for the same engine is also exposed.

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 4.1)

Imports graphics, stats

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

Config/testthat/edition 3

VignetteBuilder knitr

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Aqi Dong [aut, cre],
Yang-Li Liao [aut],
Danhyang Lee [aut]

Maintainer Aqi Dong <donga2@erau.edu>

Repository CRAN

Date/Publication 2026-08-20 14:30:02 UTC

Contents

`rubin_pool_scmix` *Rubin's rules pooling*

Description

Pools estimates and variances across completed data sets from `scmix_impute`.

Usage

```
rubin_pool_scmix(completed, est_fun, var_fun)
```

Arguments

`completed`	list of completed data matrices.
`est_fun, var_fun`	functions mapping a completed matrix to a named numeric vector of estimates and matching variances.

Value

List with pooled estimates, total variance, standard errors, degrees of freedom, and 95 percent interval endpoints.

Examples

```
Y <- scmix_sim_mw3(n = 120, seed = 7)$Y
set.seed(7)
Y[sample(length(Y), 30)] <- NA
imp <- scmix_impute(Y, M = 2, K = 4, n_iter = 300, burn = 100, seed = 8)
pool <- rubin_pool_scmix(imp$completed, colMeans,
                         function(d) apply(d, 2, stats::var) / nrow(d))
pool$est
pool$se
```

scmix *Bayesian sparse conditional mixture clustering*

Description

Fits the Bayesian sparse conditional mixture model of Dong, Liao, and Lee to a numeric data matrix and returns a clustering. Each mixture component factorizes into a chain of univariate polynomial regressions (degree `m`) with per-component, per-equation Bayesian variable selection under a centered Zellner g-prior; the number of clusters is selected within the run by an overfitted sparse mixture (Dirichlet concentration `a = 1/K`). A single run returns the estimated number of clusters, a label-invariant consensus partition, selection probabilities for every term of every within-component regression, and coefficient summaries.

Usage

```
scmix(
  Y,
  K = 7L,
  m = 2L,
  n_iter = 1500L,
  burn = 600L,
  order = NULL,
  a = 1/K,
  g = NULL,
  enum_max = 12L,
  seed = NULL,
  verbose = 0L,
  ...
)
```

Arguments

`Y`	numeric matrix (n x p); rows are observations. Must be complete for `m >= 2`.
`K`	integer, overfitted cap on the number of components (the number of occupied components is estimated; `K` only needs to exceed the plausible truth). Default 7.
`m`	integer polynomial degree of the conditional means (2 = quadratic, the recommended default; 1 = linear).
`n_iter`, `burn`	total and burn-in Gibbs sweeps.
`order`	optional integer permutation of `1:p` giving the conditioning order. Defaults to the column order of `Y`. For `m >= 2`, results depend on the order (see the paper's order-sensitivity study); an inflated number of clusters relative to other orders is a practical misfit diagnostic.
`a`	Dirichlet concentration of the mixture weights (default `1/K`; must be small for superfluous components to empty out).

g	g-prior scale: NULL (default) uses unit information g = n_k per component; a number fixes g; a function(nk) is evaluated per component. See the paper for the effect on the sigma^2 posterior at extreme signal-to-noise ratios.
enum_max	exact enumeration of selection sets while a regression has at most enum_max candidate terms; larger sets use validated single-flip Metropolis-Hastings. Default 12; 6 is a good speed/exactness compromise in higher dimension.
seed	optional integer seed (full run reproducibility).
verbose	print progress every verbose sweeps (0 = silent).
...	further arguments passed to the engine (see scmix_engine), e.g. n_min, nflip.

Value

An object of class "scmix": a list with components

cluster integer vector, the consensus partition (Dahl's least-squares criterion over the posterior similarity matrix).

K number of clusters in the consensus partition.

Kstar_mode posterior mode of the number of occupied components across sweeps.

inclusion list (one matrix per mixture component label) of posterior selection probabilities; rows = equations, columns = candidate terms (named).

beta, sig2 coefficient summaries (posterior means conditional on inclusion) and residual-variance posterior means, per label.

alive_frac fraction of post-burn-in sweeps each label was occupied (interpret per-label summaries only for labels with alive_frac near 1).

z_draw the final-sweep posterior draw of the allocations (a sample, not a point estimate; prefer cluster).

order, call, runtime bookkeeping.

References

Dong, A., Liao, Y.-L., and Lee, D. (2026). Replacing three nested searches with one sweep: a Bayesian treatment of sparse conditional mixture clustering. Melnykov, V. and Wang, Y. (2023). Conditional mixture modeling and model-based clustering. *Pattern Recognition* 133, 108994. doi:10.1016/j.patcog.2022.108994 Dahl, D. B. (2006). Model-based clustering for expression data via a Dirichlet process mixture model. In *Bayesian Inference for Gene Expression and Proteomics*, 201–218. Cambridge University Press.

Examples

```
set.seed(1)
n <- 200
x1 <- rnorm(n)
x2 <- c(0.5 + x1[1:100]^2, -2 - 0.8 * x1[101:200]^2) + rnorm(n, 0, 0.6)
fit <- scmix(cbind(x1, x2), K = 5, m = 2, n_iter = 400, burn = 150,
             seed = 42)
fit$K            # curved clusters found from one run
table(fit$cluster, rep(1:2, each = 100))
```

```
summary(fit)
```

`scmix_ari` *Adjusted Rand index*

Description

The adjusted Rand index between two partitions (Hubert and Arabie, 1985); label-invariant, 1 for identical partitions, ~0 for unrelated ones.

Usage

```
scmix_ari(a, b)
```

Arguments

a, b	Two label vectors of equal length.

Value

The adjusted Rand index, a scalar.

Examples

```
a <- rep(1:3, each = 5)
b <- c(rep(2:1, each = 5), rep(3, 5))   # same partition, labels permuted
scmix_ari(a, b)
```

`scmix_engine` *Low-level engine (advanced use)*

Description

The validated sampling engine underlying `scmix` and `scmix_impute`, exposed for power users; see the package source for complete documentation of every argument and output.

Usage

```
scmix_engine(Y, ...)
```

Arguments

Y	numeric matrix, NA = missing (complete required for m >= 2).
...	engine arguments; see `scmix`.

Value

Engine output list; see scmix for main fields.

Examples

```
d <- scmix_sim_mw3(n = 150, seed = 2)
eng <- scmix_engine(d$Y, K = 4, m = 1, n_iter = 200, burn = 80,
                    consensus = TRUE, seed = 3)
scmix_ari(eng$z_consensus, d$z)   # linear conditionals, short run

# quadratic conditionals at a more realistic length
eng2 <- scmix_engine(d$Y, K = 4, m = 2, n_iter = 400, burn = 200,
                     consensus = TRUE, seed = 3)
eng2$Kstar_consensus
scmix_ari(eng2$z_consensus, d$z)
```

---

scmix_impute *Multiple imputation with the sparse conditional mixture engine*

Description

Runs the same engine as scmix in multiple-imputation mode (linear conditionals, m = 1) for a data matrix with missing values, returning completed data sets and Rubin-pooling utilities.

Usage

```
scmix_impute(
  Y,
  M = 10L,
  K = 10L,
  n_iter = 1500L,
  burn = 500L,
  a = 1/K,
  seed = NULL,
  verbose = 0L,
  ...
)
```

Arguments

Y	numeric matrix with NAs.
M	number of completed data sets to return.
K	integer, overfitted cap on the number of components (the number of occupied components is estimated; K only needs to exceed the plausible truth). Default 7.
n_iter, burn	total and burn-in Gibbs sweeps.

a	Dirichlet concentration of the mixture weights (default 1/K; must be small for superfluous components to empty out).
seed	optional integer seed (full run reproducibility).
verbose	print progress every verbose sweeps (0 = silent).
...	further engine arguments.

Value

List with completed (list of M completed matrices) and the fitted-model summaries of scmix.

See Also

rubin_pool_scmix

Examples

```
Y <- scmix_sim_mw3(n = 120, seed = 7)$Y
set.seed(7)
Y[sample(length(Y), 30)] <- NA
imp <- scmix_impute(Y, M = 2, K = 4, n_iter = 300, burn = 100, seed = 8)
length(imp$completed)
anyNA(imp$completed[[1]])
```

---

scmix_sim_mw3 *Simulate the p = 3 replication setting of Melnykov and Wang (2023)*

Description

Draws one data set from the three-component, quadratic-conditional setting of Melnykov and Wang (2023), supplementary Table S-1 (p = 3), as replicated in the accompanying paper.

Usage

```
scmix_sim_mw3(n = 1000, seed = NULL)
```

Arguments

n	Sample size.
seed	Optional seed.

Value

A list with elements Y (an n x 3 matrix) and z (the true component labels).

References

Melnykov, V. and Wang, Y. (2023). Conditional mixture modeling and model-based clustering. *Pattern Recognition*, 133:108994. doi:10.1016/j.patcog.2022.108994

Examples

```
d <- scmix_sim_mw3(n = 300, seed = 1)
dim(d$Y)
table(d$z)
```

Index