Package: mzimp (via r-universe)

November 6, 2026

Title Zero-Inflated Mixed Poisson Regression for Multivariate Count Data

Version 0.1.0

Description Fits and selects zero-inflated mixed Poisson models for multivariate count regression. Supports multiple mixing distributions and variance specifications for flexible marginal modeling, while allowing dependence in the structural-zero components and latent mixing intensities to be modeled separately through copulas. The methodology is described in Nguyen (2026) <doi:10.48550/arXiv.2608.12732>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

Imports Rcpp, numDeriv, MASS, statmod, VineCopula, codetools, mvtnorm, pbivnorm, pkgload, ps, randtoolbox

LinkingTo Rcpp, RcppArmadillo

Suggests knitr, rmarkdown, PLNmodels

NeedsCompilation yes

Author Quang Huy Nguyen [aut, cre, cph]

Maintainer Quang Huy Nguyen <huynqtkt@neu.edu.vn>

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

Date/Publication 2026-11-06 18:09:40 UTC

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

RemoteRef HEAD

RemoteSha 759569c5cf4cd2a3bb06c991974a9ec505cbe98b

Contents

fit_bizimp 2
fit_multizimp 9

fit_zimp 16
print.bizimp_selection_summary 21
print.zimp_selection_summary 22
select_bizimp 22
select_multi_margin 29
select_zimp 37

Index **43**

`fit_bizimp` *Fit a Bivariate Mixed Poisson or Zero-Inflated Mixed Poisson Model*

Description

Fits a bivariate Mixed Poisson (MP) or Zero-Inflated Mixed Poisson (ZIMP) regression model using inference functions for margins (IFM).

The two marginal models are first fitted separately using `fit_zimp()`. Dependence between their latent mixing variables is then modeled by a specified bivariate copula. When both margins are zero-inflated, dependence between the Bernoulli activation indicators may also be modeled.

Usage

```
fit_bizimp(
  formula1 = NULL,
  formula2 = NULL,
  data = NULL,
  Y = NULL,
  X = NULL,
  Z = NULL,
  mixing.distribution.1 = "gamma",
  mixing.distribution.2 = "gamma",
  var.power = NULL,
  mixing.copula = 1L,
  zero.dependence = c("gaussian", "comonotone", "countermonotone"),
  d0 = 10L,
  n_gh = 20,
  gh = NULL,
  ini.par = NULL,
  progress = FALSE,
  trace = 0,
  eval.max = 100,
  rel.tol = 1e-05,
  x.tol = 1e-05,
  iter.max = 100,
  gaussian.sign.tol = 1e-06,
  t.df.grid = c(3, 5, 10, 30)
)
```

Arguments

`formula1` Optional formula specifying the marginal model for the first response. Use a one-part formula for an MP margin and a two-part formula of the form `y1 ~ ... | ...` for a ZIMP margin.

`formula2` Optional formula specifying the marginal model for the second response. It follows the same syntax as `formula1`.

`data` Data frame containing the variables referenced by `formula1` and `formula2`. When the formula interface is used, `formula1`, `formula2`, and `data` must all be supplied.

`Y` Response for the direct matrix interface. Either an $n \times 2$ numeric matrix/data frame or a list of exactly two numeric vectors. Responses must contain finite, non-negative integer counts.

`X` Count-component design for the direct interface. May be a numeric matrix/data frame shared by both margins or a list of exactly two margin-specific design matrices. Include intercept columns explicitly when required.

`Z` Optional zero-component design for the direct interface. May be `NULL`, a numeric matrix/data frame shared by both margins, or a list of exactly two margin-specific designs. A `NULL` list element specifies an MP margin; a non-`NULL` element specifies a ZIMP margin.

`mixing.distribution.1`
Character string specifying the mixing distribution for margin 1. One of `"gamma"`, `"lnorm"`, `"ig"`, `"weibull"`, `"loglogistic"`, or `"betaprime"`.

`mixing.distribution.2`
Character string specifying the mixing distribution for margin 2. One of `"gamma"`, `"lnorm"`, `"ig"`, `"weibull"`, `"loglogistic"`, or `"betaprime"`.

`var.power` Optional variance-power specification for the two marginal models. If `NULL`, each margin uses the native/default parameterization of its mixing distribution. A scalar value from `1`, `2`, or `3` is applied to both margins. A vector of length two specifies the variance power separately for margins 1 and 2.

`mixing.copula` Integer `VineCopula` family code specifying exactly one copula family for dependence between the two latent mixing variables. Supported family codes are:

- `0L` Independence copula.
- `1L` Gaussian copula (default).
- `2L` Student-t copula.
- `3L` Clayton copula.
- `4L` Gumbel copula.
- `5L` Frank copula.
- `6L` Joe copula.
- `7L` BB1 copula.
- `8L` BB6 copula.
- `9L` BB7 copula.
- `10L` BB8 copula.
- `13L`, `14L`, `16L–20L` 180-degree rotations of Clayton, Gumbel, Joe, and BB copulas.

23L, 24L, 26L–30L 90-degree rotations of Clayton, Gumbel, Joe, and BB copulas.

33L, 34L, 36L–40L 270-degree rotations of Clayton, Gumbel, Joe, and BB copulas.

104L, 114L, 124L, 134L, 204L, 214L, 224L, 234L Tawn copula families and their rotations.

Family codes follow the convention used by the VineCopula package. The default is 1L, corresponding to the Gaussian copula. fit_bizimp() fits exactly one requested family; use select_bizimp() to compare multiple copula families.

zero.dependence

Character string specifying dependence between the Bernoulli activation indicators when both margins are zero-inflated. One of "gaussian", "comonotone", or "countermonotone". This argument is inactive unless both margins are zero-inflated.

d0 Positive integer specifying the checkerboard grid size used to approximate the mixing copula. The default is 10L.

n_gh Positive integer specifying the number of Gauss–Hermite quadrature nodes used for lognormal marginal mixing. The default is 20.

gh Optional precomputed Gauss–Hermite quadrature object. It is passed to the lognormal marginal calculations. If NULL, the required quadrature rule is constructed internally.

ini.par Optional list of length two containing initial parameter vectors for the two marginal fit_zimp() fits. NULL uses the default initialization procedure for both margins.

progress Logical indicating whether fitting progress should be displayed. The default is FALSE.

trace Numeric optimizer trace level. The default is 0.

eval.max Positive integer giving the maximum number of objective function evaluations. The default is 100.

rel.tol Positive numeric value giving the relative convergence tolerance used in marginal fitting. The default is 1e-5.

x.tol Positive numeric value giving the parameter convergence tolerance used in marginal fitting. The default is 1e-5.

iter.max Positive integer giving the maximum number of optimizer iterations. The default is 100.

gaussian.sign.tol

Positive numeric tolerance used internally to classify the sign of the Gaussian pilot dependence estimate. The default is 1e-6.

t.df.grid Integer vector of candidate degrees of freedom greater than two used when mixing.copula = 2, corresponding to the Student-t copula. The default is c(3, 5, 10, 30).

Details

`fit_bizimp()` fits one fully specified bivariate model. It does not perform model selection over marginal distributions or dependence structures.

Let the two observed counts be

$$Y_j = B_j N_j, \qquad j = 1, 2,$$

where N_j follows a Mixed Poisson model and B_j is an optional Bernoulli activation indicator. For an MP margin, $B_j = 1$ identically.

Conditional on the latent intensity,

$$N_j \mid \Lambda_j \sim \text{Poisson}(\Lambda_j),$$

with marginal mean

$$\mu_{ij} = \exp(x_{ij}^\top \beta_j).$$

For a zero-inflated margin,

$$\Pr(B_{ij} = 1) = \pi_{ij}, \qquad \text{logit}(\pi_{ij}) = z_{ij}^\top \gamma_j.$$

Thus, $B_{ij} = 1$ indicates that observation i belongs to the count-generating component of margin j.

Dependence between Λ_1 and Λ_2 is represented by the copula specified by `mixing.copula`. The copula is evaluated through a checkerboard approximation with grid size `d0`.

When both margins are zero-inflated, dependence between B_1 and B_2 is specified separately through `zero.dependence`.

Value

An object of class `"bizimp_fit"` containing the fitted marginal models, dependence model, likelihood, and information criteria. Important components include:

`fit_margin_1`, `fit_margin_2` Fitted univariate marginal models returned by `fit_zimp()`.

`marginal_1`, `marginal_2` Internal representations of the fitted marginal models used in the dependence likelihood.

`best_model` Fitted dependence model. It contains the requested mixing-copula family and estimated copula parameter(s), the zero-dependence parameter when applicable, the maximized bivariate log-likelihood, convergence information, and the full-model AIC and BIC.

`mixing.distribution.1`, `mixing.distribution.2` Mixing distributions used for margins 1 and 2.

`var.power` User-supplied marginal variance-power specification, or `NULL` when native/default parameterizations were used.

`mixing.copula` `VineCopula` family code used for latent mixing dependence.

`zero.dependence` Zero-dependence specification when both margins are zero-inflated; otherwise `NA`.

`zero.copula.rho` Estimated Gaussian zero-copula correlation when both margins are zero-inflated and `zero.dependence = "gaussian"`; otherwise `NA`.

`zero_inflation` Logical vector of length two indicating whether each margin is zero-inflated.

`npar` Total number of parameters used to calculate the full-model information criteria.

`AIC`, `BIC` Akaike and Bayesian information criteria for the complete bivariate model.

`interface` Character string identifying whether the formula or direct matrix interface was used.

`formula1`, `formula2` Marginal formulas when the formula interface was used.

`call` Matched call to `fit_bizimp()`.

Estimation

Estimation follows the IFM principle:

1. Fit margin 1 using `fit_zimp()`.
2. Fit margin 2 independently using `fit_zimp()`.
3. Conditional on the fitted marginal parameters, estimate the dependence parameter(s) from the bivariate likelihood.

Consequently, the marginal estimates returned in `fit_margin_1` and `fit_margin_2` are the corresponding independent `fit_zimp()` estimates and are not re-estimated during dependence fitting.

`fit_bizimp()` fits exactly one requested mixing-copula family. When the requested family is not Gaussian, a Gaussian dependence model may be fitted internally as a pilot for initialization and dependence-sign information; it is not an additional candidate model.

Marginal configurations

Each margin may independently be an MP or ZIMP model. The four supported configurations are therefore:

- MP–MP;
- ZIMP–MP;
- MP–ZIMP;
- ZIMP–ZIMP.

For direct input, `Z = NULL` specifies MP–MP. A list such as `Z = list(Z1, NULL)` specifies ZIMP–MP, while `Z = list(NULL, Z2)` specifies MP–ZIMP. Supplying a shared matrix `Z` applies the zero-component design to both margins.

Dependence between the Bernoulli activation indicators is estimated only for the ZIMP–ZIMP configuration.

Model specification

Formula interface

Two formulas are supplied, one for each margin. For example,

```
formula1 = y1 ~ 1 + x1 + x2
formula2 = y2 ~ 1 + x1
```

specifies two MP margins. Zero-inflated margins are specified using two-part formulas:

```
formula1 = y1 ~ 1 + x1 + x2 | z1 + z2
formula2 = y2 ~ 1 + x1      | z1
```

The expression before `|` specifies the count component and the expression after `|` specifies the zero-inflation component.

As in `fit_zimp()`, intercepts are included only when explicitly requested with `1`. This convention applies separately to the count and zero-inflation components.

Matrix interface

`Y` may be an $n \times 2$ matrix/data frame or a list containing two response vectors.

`X` may be a common design matrix for both margins or a list `list(X1, X2)` containing margin-specific count designs. Similarly, `Z` may be a common zero-component design or a list containing margin-specific designs. Intercept columns must be included explicitly when required.

Mixing distributions

The mixing distribution is specified separately for each margin through `mixing.distribution.1` and `mixing.distribution.2`. Supported distributions are:

`"gamma"` Gamma mixing distribution.

`"lnorm"` Lognormal mixing distribution.

`"ig"` Inverse Gaussian mixing distribution.

`"weibull"` Weibull mixing distribution.

`"loglogistic"` Log-logistic mixing distribution.

`"betaprime"` Beta-prime mixing distribution.

The two margins may use different mixing distributions.

Dependence structure

`mixing.copula` is a single integer family code supported by `VineCopula`. It specifies the copula connecting the two latent mixing variables. The default `mixing.copula = 1` corresponds to the Gaussian copula.

For `mixing.copula = 2`, corresponding to the Student-t copula, the degrees of freedom are selected over the values supplied in `t.df.grid`.

When both margins are zero-inflated, `zero.dependence` specifies the dependence structure between their Bernoulli activation indicators:

`"gaussian"` Gaussian copula dependence with an estimated correlation parameter.

`"comonotone"` Upper Fréchet-bound dependence.

`"countermonotone"` Lower Fréchet-bound dependence.

See Also

`fit_zimp()` for fitting a univariate marginal model and `select_bizimp()` for selecting among multiple bivariate model specifications.

Examples

```
## Generate a small bivariate Mixed Poisson data set
set.seed(12345)
n <- 50
x <- rnorm(n)
X <- cbind(1, x)

mu1 <- exp(X %*% c(0.4, 0.3))
mu2 <- exp(X %*% c(0.2, -0.2))

## Gamma-mixed Poisson margins
y1 <- rnbinom(n, size = 1 / 0.2, mu = mu1)
y2 <- rnbinom(n, size = 1 / 0.3, mu = mu2)
Y <- cbind(y1, y2)

## MP--MP model with a Gaussian mixing copula
fit <- fit_bizimp(
  Y = Y,
  X = X,
  mixing.distribution.1 = "gamma",
  mixing.distribution.2 = "gamma",
  var.power = 2,
  mixing.copula = 1L,
  progress = FALSE
)

fit$best_model
fit$AIC
fit$BIC

## Margin-specific designs and a ZIMP--MP configuration
z1 <- rnorm(n)
Z1 <- cbind(z1)

fit_zm <- fit_bizimp(
  Y = Y,
  X = list(X, X),
  Z = list(Z1, NULL),
  mixing.distribution.1 = "gamma",
  mixing.distribution.2 = "gamma",
  var.power = c(2, 2),
  mixing.copula = 1L,
```

```
  progress = FALSE
)

fit_zm$zero_inflation
```

`fit_multizimp` *Fit a Multivariate Zero-Inflated Mixed Poisson Model*

Description

Fits a multivariate Mixed Poisson (MMP) or Zero-Inflated Mixed Poisson (MZIMP) model with factor-copula dependence.

Marginal models may either be fitted directly from `Y`, `X`, and optionally `Z`, or supplied through an object returned by `select_multi_margin()`. After the marginal models have been fitted or supplied, `fit_multizimp()` estimates dependence among the latent mixed-Poisson components and, when requested, dependence among structural-zero indicators.

The marginal and dependence parameters are estimated using an inference-functions-for-margins (IFM) strategy.

Usage

```
fit_multizimp(
  Y = NULL,
  X = NULL,
  Z = NULL,
  margin = NULL,
  mixing.distribution = "gamma",
  var.power = NULL,
  margin.ini.par = NULL,
  parallel = NULL,
  margin.block_size = NULL,
  copula.block_size = 500L,
  d0 = 10,
  J = 1L,
  L = 0L,
  latent.dependence = c("gaussian", "biexp"),
  latent.factor.skewness = 0.5,
  latent.loglike = NULL,
  gh_order = 16L,
  approx.hermite.joint.order = 4L,
  hermite.polar.cycles = 6L,
  hermite.polar.v.iter = 8L,
  hermite.polar.r.iter = 4L,
  hermite.polar.tol = 1e-05,
  hermite.polar.r.max = 0.995,
  genz.conditional.Q = 256L,
```

```
  genz.conditional.antithetic = TRUE,
  qmc.n_rep = 4L,
  qmc.seed = 1L,
  zero.dependence = c("gaussian", "biexp"),
  zero.factor.skewness = 0.5,
  biexp.root_tol = 1e-10,
  psi0_start = NULL,
  rho0_start = NULL,
  gh.progressive = TRUE,
  use.gradient = TRUE,
  progress = TRUE,
  trace = 0,
  eval.max = 1000L,
  iter.max = 100L,
  tiny = 1e-300,
  big_penalty = 1e+12,
  rel.tol = 1e-04
)
```

Arguments

`Y`	Optional $n \times k$ numeric matrix or data frame of multivariate count responses. Required when `margin = NULL`. Entries must be finite, non-negative integer counts and k must be at least two.
`X`	Count-component design used when `margin = NULL`. May be a matrix/data frame shared by all margins or a list of length k containing margin-specific designs. Intercepts must be supplied explicitly when required.
`Z`	Optional zero-component design used when `margin = NULL`. `NULL` specifies ordinary mixed Poisson margins. A matrix/data frame specifies a common zero-component design, while a list may provide margin-specific designs.
`margin`	Optional object returned by `select_multi_margin()`. When supplied, the selected marginal models are reused and the direct marginal-fitting arguments `Y`, `X`, `Z`, `mixing.distribution`, `var.power`, and `margin.ini.par` are not used to reselect the margins.
`mixing.distribution`	Mixing distribution used for direct marginal fitting when `margin = NULL`. A single value may be shared by all margins or a vector may specify one distribution per margin. Supported distributions are `"gamma"`, `"lnorm"`, `"ig"`, `"weibull"`, `"loglogistic"`, and `"betaprime"`.
`var.power`	Optional variance-power specification used for direct marginal fitting.
`margin.ini.par`	Optional initial parameter specification for direct marginal fitting.
`parallel`	`NULL` or a positive integer controlling supported parallel computation. `NULL` or `1` requests serial execution.
`margin.block_size`	Optional positive integer controlling the block size used during parallel marginal fitting.

`copula.block_size`
: Positive integer giving the number of observations processed in each dependence-likelihood block. The default is `500L`.

`d0`
: Integer greater than or equal to `2` specifying the checkerboard resolution used in the count dependence layer. The default is `10`.

`J`
: Positive integer giving the number of factors in the count/intensity dependence layer. It must satisfy `1 <= J < k`. The default is `1L`.

`L`
: Integer specifying the number of factors in the structural-zero dependence layer. The current implementation supports `0` and `1`. `L = 0` disables zero-layer factor dependence; `L = 1` requests a one-factor model when estimable.

`latent.dependence`
: Character string specifying the count-layer factor family. Supported values are `"gaussian"` and `"biexp"`.

`latent.factor.skewness`
: Numeric parameter controlling skewness of the standardized bilateral-exponential factor distribution when `latent.dependence = "biexp"`.

`latent.loglike`
: Likelihood method for Gaussian count factors. May be `NULL`, `"GH"`, or `"hermite"`. If `NULL`, the method is selected automatically according to the factor dimension.

`gh_order`
: Integer greater than or equal to `2` specifying the quadrature order used by dependence calculations where applicable. The default is `16L`.

`approx.hermite.joint.order`
: Hermite approximation order used by the joint count/zero likelihood calculation where applicable. The default is `4L`.

`hermite.polar.cycles`
: Positive integer giving the maximum number of alternating direction/radius cycles used by the polar Hermite optimizer.

`hermite.polar.v.iter`
: Positive integer giving the maximum number of optimizer iterations for the direction step of each polar cycle.

`hermite.polar.r.iter`
: Positive integer giving the maximum number of optimizer iterations for the radius step of each polar cycle.

`hermite.polar.tol`
: Positive numeric convergence tolerance for the polar Hermite optimizer.

`hermite.polar.r.max`
: Numeric value in `(0, 1)` giving the maximum admissible row norm for factor loadings in the polar parameterization.

`genz.conditional.Q`
: Positive integer giving the number of conditional randomized quasi-Monte Carlo paths used for final Genz likelihood evaluation where applicable.

`genz.conditional.antithetic`
: Logical indicating whether antithetic uniforms are used in the conditional RQMC evaluation.

`qmc.n_rep`
: Positive integer giving the number of randomized QMC replicates used for final likelihood evaluation.

`qmc.seed` Integer seed used for randomized QMC calculations and other reproducible stochastic computations within the fitting procedure.

`zero.dependence`
Character string specifying the factor family for the structural-zero layer when `L = 1`. Supported values are `"gaussian"` and `"biexp"`.

`zero.factor.skewness`
Numeric parameter controlling skewness of the standardized bilateral-exponential factor distribution when `zero.dependence = "biexp"`.

`biexp.root_tol` Positive numeric tolerance used in numerical root finding required by the bilateral-exponential factor backend.

`psi0_start` Optional numeric vector of raw starting parameters for the count-layer factor-loading matrix.

`rho0_start` Optional numeric vector of raw starting parameters for the structural-zero factor-loading matrix.

`gh.progressive` Logical. If `TRUE`, supported Gaussian count-factor fits may use progressively fitted lower-dimensional factor models as warm starts.

`use.gradient` Logical indicating whether available analytic gradients are used during dependence optimization.

`progress` Deprecated and ignored. Retained for backward API compatibility; public fitting is currently silent.

`trace` Deprecated and ignored. Retained for backward API compatibility.

`eval.max` Positive integer giving the maximum number of objective-function evaluations for the main dependence optimizer.

`iter.max` Positive integer giving the maximum number of iterations for the main dependence optimizer.

`tiny` Positive numeric lower bound used to protect probabilities before logarithms are evaluated.

`big_penalty` Positive numeric objective value returned for invalid parameter states during optimization.

`rel.tol` Positive numeric relative convergence tolerance for the main dependence optimizer.

Details

For margin $j = 1, \ldots, k$, the observed response is represented as

$$Y_j = B_j N_j,$$

where N_j follows a mixed Poisson model and B_j is an activation indicator when zero inflation is present. For an ordinary mixed Poisson margin, $B_j = 1$ with probability one.

The model separates two sources of cross-margin dependence:

1. dependence among the latent mixed-Poisson components underlying $N_1, \ldots, N_k$; and
2. dependence among the activation indicators for zero-inflated margins.

These two dependence structures are modeled separately through the count layer and the structural-zero layer.

Value

An object of class "multi_zipm_fit" containing the fitted marginal models and dependence structure. Important components include:

method Estimation method. Currently "IFM".

n, k Number of observations and response margins.

marginal List of length k containing the fitted marginal information. Each element contains the margin index, mixing distribution, variance power, zero-inflation status, count coefficients (be), dispersion parameter (phi), zero-component coefficients (gam) when present, and marginal log-likelihood.

marginal.dist Mixing distribution used for each response margin.

zero_inflation Logical vector indicating which margins contain a zero-inflation component.

count_layer Fitted dependence structure for the mixed-Poisson count layer. Important components include dependence, J, psi, A, implied_correlation, and information on the likelihood method. The loading matrix psi has dimension k by J.

zero_layer Structural-zero dependence information. When zero-layer dependence is estimated, this includes the fitted loading matrix rho and its implied correlation matrix. The component also records cases in which no zero-layer dependence is estimated.

copula_fit Dependence-fitting results, including the fitted log-likelihood, optimization objective, convergence information, elapsed time, optimizer output, and available final likelihood-evaluation diagnostics.

summary Ordinary list containing the principal user-facing results, including dimensions, marginal-model summary, count- and zero-layer summaries, log-likelihood, convergence information, elapsed time, and final likelihood diagnostics.

call Matched function call.

Marginal models

Marginal models can be supplied in two ways.

Direct specification.

If margin = NULL, Y and X must be supplied. The function fits exactly one marginal model for each response. No marginal model selection is performed.

mixing.distribution specifies the mixing distribution and var.power specifies its variance-power parameterization.

Z = NULL gives ordinary mixed Poisson margins. Supplying Z gives zero-inflated mixed Poisson margins.

X and Z may be common matrices/data frames shared by all margins or lists containing margin-specific designs. When lists are supplied, their lengths must equal the number of response margins.

Previously selected margins.

Alternatively, margin may be an object returned by select_multi_margin(). In this case the selected marginal distributions, variance powers, regression coefficients, dispersion parameters, and zero-inflation specifications are reused without refitting or reselecting the marginal models.

If compatible checkerboard probability matrices were precomputed by select_multi_margin(), they are reused when possible.

A margin object may contain a mixture of ordinary MP and ZIMP margins.

Count dependence

Dependence among the latent mixed-Poisson components is represented by a `J`-factor copula. The number of factors must satisfy

$$1 \leq J < k.$$

The factor family is controlled by `latent.dependence`:

`"gaussian"` Gaussian factor copula.

`"biexp"` Factor copula based on a standardized bilateral-exponential distribution.

The fitted count-layer loading matrix is denoted by Ψ and has dimension $k \times J$. The parameterization guarantees

$$\sum_{r=1}^{J} \psi_{jr}^2 < 1$$

for every margin j, leaving a positive idiosyncratic variance.

For Gaussian factors, `latent.loglike` controls likelihood evaluation. `"GH"` uses Gauss–Hermite quadrature and `"hermite"` uses the Hermite-based approximation implemented by the package. If `latent.loglike = NULL`, the method is selected automatically.

Structural-zero dependence

Zero-inflated margins may additionally share a dependence structure through their activation indicators.

`L = 0` specifies no factor dependence in the structural-zero layer. `L = 1` requests a one-factor zero-layer dependence model when at least two zero-inflated margins are available.

The zero-layer factor family is specified by `zero.dependence`, with supported values `"gaussian"` and `"biexp"`.

If no margin is zero inflated, or if only one margin is zero inflated, no cross-margin zero-layer dependence can be estimated. Ordinary MP margins do not participate in the structural-zero dependence model.

Thus a multivariate model may contain any combination of MP and ZIMP margins.

Checkerboard probabilities

Dependence estimation uses checkerboard interval probabilities derived from the fitted marginal distributions. The checkerboard resolution is controlled by `d0`.

These marginal quantities are computed before dependence optimization and do not depend on the factor-loading parameters. When a compatible cache is available in an object returned by `select_multi_margin()`, it is reused where possible.

Estimation

The model is fitted using an IFM strategy. Marginal parameters are estimated first, or taken from `margin`, and are subsequently held fixed while the dependence parameters are estimated.

For Gaussian count factors, `gh.progressive = TRUE` allows lower-dimensional fitted factor models to be used as starting values for higher-dimensional models.

`parallel` controls supported parallel computations. `NULL` or `1` requests serial computation; an integer greater than one requests the corresponding level of parallelism where supported.

See Also

`select_multi_margin()` for selecting marginal models before dependence fitting; `fit_zimp()` and `select_zimp()` for univariate models; `fit_bizimp()` for the specialized bivariate model.

Examples

```
set.seed(12345)

n <- 15
k <- 3
x <- rnorm(n)
X <- cbind(Intercept = 1, x = x)

beta <- c(0.5, 0.3)
mu <- exp(X %*% beta)

Y <- cbind(
  y1 = rpois(n, mu),
  y2 = rpois(n, mu),
  y3 = rpois(n, mu)
)

## Directly specified mixed-Poisson margins
fit <- fit_multizimp(
  Y = Y,
  X = X,
  Z = NULL,
  mixing.distribution = "gamma",
  var.power = 2,
  J = 1,
  L = 0,
  d0 = 3,
  parallel = 1
)

fit$summary
fit$count_layer$psi

## Alternatively, select the marginal models first
margins <- select_multi_margin(
  Y = Y,
  X = X,
```

```
  Z = NULL,
  mixing.distribution = c("gamma", "lnorm"),
  var.power = 2,
  compute.I = TRUE,
  I.d0 = 3,
  parallel = 1
)

fit2 <- fit_multizimp(
  margin = margins,
  J = 1,
  L = 0,
  d0 = 3,
  parallel = 1
)
fit2$summary
```

`fit_zimp` *Fit a Univariate Mixed Poisson or Zero-Inflated Mixed Poisson Model*

Description

Fits a single univariate Mixed Poisson (MP) or Zero-Inflated Mixed Poisson (ZIMP) regression model for a specified mixing distribution and variance-power parameterization.

Both formula and matrix interfaces are supported. The count component models the mean of the Poisson mixing distribution, while an optional zero-inflation component models the probability that the count-generating component is active.

Usage

```
fit_zimp(
  formula = NULL,
  data = NULL,
  Y = NULL,
  X = NULL,
  Z = NULL,
  offset.count = NULL,
  mixing.distribution = "gamma",
  var.power = NULL,
  n_gh = 20,
  gh = NULL,
  ini.par = NULL,
  progress = TRUE,
  trace = 0,
  eval.max = 100,
  rel.tol = 1e-05,
  x.tol = 1e-05,
  iter.max = 100
)
```

Arguments

formula	Optional model formula. For an MP model, use a formula such as y ~ 1 + x1 + x2. For a ZIMP model, use a two-part formula such as y ~ 1 + x1 + x2 | 1 + z1 + z2. Intercepts must be included explicitly using 1. When formula is supplied, data must also be supplied.
data	Optional data frame containing the variables referenced in formula.
Y	Numeric vector containing the count response for the matrix interface. Values must be finite, non-negative integers.
X	Numeric matrix or data frame containing the design variables for the count component when using the matrix interface. It must have length(Y) rows. Include a column of ones explicitly if an intercept is required.
Z	Optional numeric matrix or data frame containing the design variables for the zero-inflation component when using the matrix interface. It must have length(Y) rows. Include a column of ones explicitly if an intercept is required. If NULL, an MP model is fitted.
offset.count	Optional numeric offset for the count linear predictor. For the matrix interface, it must be a finite numeric vector of length length(Y). If NULL, no count offset is used. With the formula interface, an offset may be specified in the count component using offset().
mixing.distribution	Character string specifying the mixing distribution. One of "gamma", "lnorm", "ig", "weibull", "loglogistic", or "betaprime". The default is "gamma".
var.power	Optional variance-power specification. If NULL, the native/default parameterization of the selected mixing distribution is used. Otherwise, it must be one of 1, 2, or 3.
n_gh	Positive integer specifying the number of Gauss–Hermite quadrature nodes when Gauss–Hermite integration is required. The default is 20.
gh	Optional precomputed Gauss–Hermite quadrature object. If NULL, the required quadrature rule is constructed internally.
ini.par	Optional numeric vector of initial parameter values. Its length must be ncol(X) + 1 for an MP model and ncol(X) + 1 + ncol(Z) for a ZIMP model. The parameter order is c(beta, log(phi), gamma).
progress	Logical indicating whether fitting progress should be displayed. The default is TRUE.
trace	Numeric optimizer trace level. The default is 0.
eval.max	Positive integer giving the maximum number of objective function evaluations. The default is 100.
rel.tol	Positive numeric value giving the relative convergence tolerance. The default is 1e-5.
x.tol	Positive numeric value giving the parameter convergence tolerance. The default is 1e-5.
iter.max	Positive integer giving the maximum number of optimizer iterations. The default is 100.

Details

`fit_zimp()` fits one specified model. It does not perform model selection over mixing distributions or variance-power specifications. For model selection, see `select_zimp()`.

The conditional count model is

$$Y \mid \Lambda \sim \text{Poisson}(\Lambda),$$

where the distribution of the latent intensity Λ is determined by `mixing.distribution`. The marginal mean is linked to the count covariates through

$$\mu_i = \exp(x_i^\top \beta + o_i),$$

where o_i is an optional count offset.

For a zero-inflated model, let B_i denote the indicator that the count-generating component is active. The model uses

$$Y_i = B_i N_i,$$

with

$$\Pr(B_i = 1) = \pi_i, \qquad \text{logit}(\pi_i) = z_i^\top \gamma.$$

Thus, `Z` models the probability of belonging to the count-generating component rather than the structural-zero component.

The dispersion parameter `phi` is constrained to be positive and is optimized internally on the log scale.

Value

An object of class `"zimp_fit"` containing the fitted model. Important components include:

`be` Estimated regression coefficients $\hat{\beta}$ for the count component.

`phi` Estimated dispersion parameter on its natural scale.

`gam` Estimated zero-component coefficients $\hat{\gamma}$ for a ZIMP model.

`log_lik_value` Maximized log-likelihood.

`neg_log_lik_value` Negative log-likelihood evaluated at the fitted parameters.

`AIC` Akaike information criterion.

`BIC` Bayesian information criterion.

`par` Estimated parameter vector on the internal optimization scale.

`convergence` Optimizer convergence code.

`message` Optimizer convergence message, when available.

`hessian` Estimated Hessian matrix, when available.

`vcov` Estimated covariance matrix, when available.

`coef_table` Coefficient and inference table produced by the fitting engine.

`count_coef_table` Coefficient table for the count component.

`zero_coef_table` Coefficient table for the zero component for a ZIMP model.

`phi_se` Standard error of the reported dispersion parameter, when available.

`y`, `X`, `Z` Response and design matrices used for fitting.

`offset.count` Count offset used in the fitted model.

`mixing.distribution` Mixing distribution used in the model.

`var.power` Variance-power specification used in the model.

`zero_inflation` Logical indicating whether the fitted model contains a zero-inflation component.

`nobs` Number of observations used in the fit.

`npar` Number of estimated parameters.

`call` Matched function call.

`summary` Printable summary containing coefficient tables, dispersion, log-likelihood, AIC, BIC, and convergence information.

Additional components describing the formula, model frames, coefficient names, interface, and missing-data handling may be included when relevant.

Model specification

Two input interfaces are available.

Formula interface

A Mixed Poisson model can be specified as

```
y ~ x1 + x2
```

and a Zero-Inflated Mixed Poisson model as

```
y ~ x1 + x2 | z1 + z2
```

The expression before `|` specifies the count component and the expression after `|` specifies the zero-inflation component.

In the `mzimp` formula interface, intercepts are included only when explicitly requested with `1`. For example,

```
y ~ x1 + x2
```

specifies a count model without an intercept, whereas

```
y ~ 1 + x1 + x2
```

includes an intercept. The same convention applies independently to the zero-inflation component.

Matrix interface

The response and design matrices can instead be supplied directly through `Y`, `X`, and optionally `Z`. If `Z = NULL`, an MP model is fitted. If `Z` is supplied, a ZIMP model is fitted. Intercept columns, when required, must be included explicitly in `X` and `Z`.

Mixing distributions

The following mixing distributions are supported:

`"gamma"` Gamma mixing distribution.

`"lnorm"` Lognormal mixing distribution.

`"ig"` Inverse Gaussian mixing distribution.

`"weibull"` Weibull mixing distribution.

`"loglogistic"` Log-logistic mixing distribution.

`"betaprime"` Beta-prime mixing distribution.

The optional `var.power` argument controls the variance-power parameterization. If `var.power = NULL`, the native/default parameterization of the selected mixing distribution is used. Otherwise, `var.power` must be one of `1`, `2`, or `3`.

Initial values

Initial parameter values may be supplied through `ini.par`. Parameters are ordered internally as

```
c(beta, log(phi), gamma)
```

where `beta` contains the count-component coefficients, `log(phi)` is the log-dispersion parameter, and `gamma` contains the zero-component coefficients. The `gamma` block is omitted for an MP model.

See Also

`select_zimp()` for selecting among multiple univariate MP or ZIMP model specifications.

Examples

```
## Generate a small Mixed Poisson data set
set.seed(12345)
n <- 50
x1 <- rnorm(n)
x2 <- rnorm(n)
X <- cbind(1, x1, x2)
mu <- exp(X %*% c(0.5, 0.3, 0.7))

## Gamma-mixed Poisson data are Negative Binomial marginally.
## This provides a short reproducible example without relying on
## non-exported simulation functions.
phi <- 0.2
y <- rnbinom(n, size = 1 / phi, mu = mu)

dat <- data.frame(y = y, x1 = x1, x2 = x2)

## Formula interface
fit_formula <- fit_zimp(
  y ~ 1 + x1 + x2,
  data = dat,
```

```
  mixing.distribution = "gamma",
  var.power = 2,
  progress = FALSE
)

fit_formula

## Equivalent matrix interface
fit_matrix <- fit_zimp(
  Y = y,
  X = X,
  mixing.distribution = "gamma",
  var.power = 2,
  progress = FALSE
)
```

`print.bizimp_selection_summary` *Print a select_bizimp summary*

Description

Print a select_bizimp summary

Usage

```
## S3 method for class 'bizimp_selection_summary'
print(x, ...)
```

Arguments

`x`	An object of class `bizimp_selection_summary`.
`...`	Additional arguments (currently unused).

Value

The summary object, invisibly.

print.zimp_selection_summary *Print a select_zimp summary*

Description

Print a select_zimp summary

Usage

```
## S3 method for class 'zimp_selection_summary'
print(x, ...)
```

Arguments

x	An object of class zimp_selection_summary.
...	Additional arguments (currently unused).

Value

The summary object, invisibly.

select_bizimp *Select a Bivariate Mixed Poisson or Zero-Inflated Mixed Poisson Model*

Description

Performs model selection for bivariate Mixed Poisson (MP) and Zero-Inflated Mixed Poisson (ZIMP) regression models.

Marginal models are first selected independently using select_zimp(). The best margin.top models from each margin are retained, and dependence structures are then evaluated for every retained marginal-model pair. The final bivariate model is selected by the largest maximized observed-data log-likelihood.

Usage

```
select_bizimp(
  formula1 = NULL,
  formula2 = NULL,
  data = NULL,
  Y = NULL,
  X = NULL,
  Z = NULL,
 mixing.distribution = c("gamma", "lnorm", "ig", "weibull", "loglogistic", "betaprime"),
```

```
  var.power = NULL,
  margin.top = 1L,
  mixing.copula = c(1L),
  d0 = 10L,
  n_gh = 20,
  gh = NULL,
  ini.par = NULL,
  parallel = NULL,
  progress = FALSE,
  show.result = FALSE,
  trace = 0,
  eval.max = 100,
  rel.tol = 1e-05,
  x.tol = 1e-05,
  iter.max = 100,
  gaussian.sign.tol = 1e-06,
  t.df.grid = c(3, 5, 10, 30)
)
```

Arguments

`formula1`	Optional formula specifying the first marginal model. The syntax follows `select_zimp()`. Used together with `formula2` and `data`.
`formula2`	Optional formula specifying the second marginal model. The syntax follows `select_zimp()`. Used together with `formula1` and `data`.
`data`	Optional data frame containing the variables referenced by `formula1` and `formula2`. Required when the formula interface is used.
`Y`	Response for the direct matrix interface. Either an $n \times 2$ numeric matrix/data frame or a list containing two numeric vectors of finite, non-negative integer counts.
`X`	Count-component design for the direct matrix interface. May be one numeric matrix/data frame shared by both margins or a list containing two margin-specific design matrices. Include intercept columns explicitly when required.
`Z`	Optional zero-component design for the direct matrix interface. May be `NULL`, one numeric matrix/data frame shared by both margins, or a list containing two margin-specific designs. A `NULL` list element specifies an MP margin; a non-`NULL` element specifies a ZIMP margin.
`mixing.distribution`	Character vector specifying candidate marginal mixing distributions passed to `select_zimp()`. Supported values are `"gamma"`, `"lnorm"`, `"ig"`, `"weibull"`, `"loglogistic"`, and `"betaprime"`. By default, all supported distributions are considered.
`var.power`	Optional numeric vector specifying candidate variance-power parameterizations passed to `select_zimp()`. Values must be selected from `1`, `2`, and `3`. If `NULL`, each requested mixing distribution uses its native/default parameterization.

`margin.top` Positive integer specifying the maximum number of independently ranked marginal models retained from each margin. The default is `1L`. If `m` models are retained from each margin, up to m^2 marginal pairs are evaluated.

`mixing.copula` Integer vector containing one or more candidate `VineCopula` family codes for dependence between the latent mixing variables. See the "Mixing-copula candidates" section for supported codes. The default is `1L` (Gaussian).

`d0` Positive integer specifying the checkerboard grid size used to discretize the mixing copulas. The default is `10L`.

`n_gh` Positive integer specifying the number of Gauss–Hermite quadrature nodes used when required by the marginal calculations. The default is `20`.

`gh` Optional precomputed Gauss–Hermite quadrature object passed to the marginal-selection and dependence calculations. If `NULL`, the required quadrature rule is constructed internally.

`ini.par` `NULL` or a list of length two containing optional initial parameter vectors for the two marginal selections. The first element is passed to margin 1 and the second to margin 2.

`parallel` `NULL` or a positive integer specifying the number of workers passed to `select_zimp()` for candidate-model fitting within each margin. `NULL` or `1` uses serial marginal selection.

`progress` Logical compatibility argument retained for the public interface. `select_bizimp()` performs model selection silently.

`show.result` Logical compatibility argument retained for the public interface. Results are stored in the returned object rather than printed during fitting.

`trace` Numeric optimizer trace argument retained for compatibility. The current selection interface performs fitting silently.

`eval.max` Positive integer giving the maximum number of objective-function evaluations used by the marginal and dependence optimizers. The default is `100`.

`rel.tol` Positive numeric value giving the relative convergence tolerance passed to the marginal model-selection routines. The default is `1e-5`.

`x.tol` Positive numeric value giving the parameter convergence tolerance passed to the marginal model-selection routines. The default is `1e-5`.

`iter.max` Positive integer giving the maximum number of optimization iterations. The default is `100`.

`gaussian.sign.tol` Non-negative numeric tolerance used to determine whether the Gaussian prescreen provides an informative dependence direction. The default is `1e-6`.

`t.df.grid` Integer vector containing candidate degrees of freedom greater than two for Student-t mixing copulas. The default is `c(3, 5, 10, 30)`.

Details

`select_bizimp()` extends `select_zimp()` from univariate marginal selection to bivariate model selection. In contrast to `fit_bizimp()`, which fits one specified bivariate model, `select_bizimp()` searches over marginal specifications and mixing-copula families.

The selection procedure consists of three stages:

1. Select candidate models independently for each margin using `select_zimp()`.
2. Retain the best `margin.top` models from each margin and perform a Gaussian dependence prescreen for every retained marginal pair.
3. Fit the compatible user-requested mixing copulas, together with the admissible zero-dependence structures, and select the candidate with the largest maximized bivariate log-likelihood.

Value

An object of class `"bizimp_selection"`. Important components include:

`margin_selection_1`, `margin_selection_2` Results from the independent `select_zimp()` selections for margins 1 and 2.

`margin_top` Requested maximum number of marginal models retained from each margin.

`retained_margin_1`, `retained_margin_2` Lists of fitted marginal models retained for the dependence search.

`pair_results` Detailed dependence-search results for every retained marginal pair, including the Gaussian prescreen, compatible mixing-copula families, zero-dependence candidates, and fitted dependence models.

`candidate_table` Data frame containing all fitted bivariate candidates, with columns describing margin 1, margin 2, zero dependence, mixing dependence, maximized log-likelihood, AIC, and BIC. Rows are ordered by decreasing log-likelihood.

`pair_summary` Data frame containing the best dependence model obtained for each retained marginal pair, ordered by decreasing bivariate log-likelihood.

`best_pair` Detailed result for the retained marginal pair containing the globally selected bivariate model.

`best_model` Selected dependence model. It contains the selected mixing-copula family and parameter estimates, zero-dependence parameter when applicable, maximized bivariate log-likelihood, convergence information, total parameter count, AIC, and BIC.

`best_dependence` Complete dependence-engine result associated with `best_model`.

`best_margin_1`, `best_margin_2` Fitted marginal models belonging to the globally selected bivariate model.

`interface` Character string identifying whether the formula or direct matrix interface was used.

`formula1`, `formula2` Marginal formulas when the formula interface was used.

`mixing.copula.candidates` User-requested candidate mixing-copula family codes.

`summary` Object of class `"bizimp_selection_summary"` containing the candidate table, selected candidate, pair summary, and matched call for formatted display.

`call` Matched call to `select_bizimp()`.

Model specification

Two input interfaces are supported.

Formula interface

Supply `formula1`, `formula2`, and `data`. Each formula follows the syntax used by `select_zimp()`. For example,

```
formula1 = y1 ~ 1 + x1 + x2 | z1 + z2
formula2 = y2 ~ 1 + x1 + x2
```

specifies a ZIMP first margin and an MP second margin.

The expression before | specifies the count component and the expression after | specifies the zero-inflation component. Intercepts are included only when explicitly requested with 1.

Both formulas must refer to the same observations. If missing values cause different observations to be retained for the two margins, bivariate model selection is stopped.

Matrix interface

Alternatively, supply Y, X, and optionally Z.

Y may be an $n \times 2$ matrix/data frame or a list containing two count vectors. X may be a design matrix shared by both margins or a list containing two margin-specific design matrices.

Z = NULL specifies two MP margins. A shared matrix/data frame Z specifies two ZIMP margins. A list may be used for mixed configurations; for example, Z = list(Z1, NULL) specifies ZIMP–MP and Z = list(NULL, Z2) specifies MP–ZIMP.

Marginal model selection

Each margin is selected independently using select_zimp() with the candidate mixing distributions supplied in mixing.distribution and the variance-power specifications supplied in var.power.

Supported mixing distributions are:

"gamma" Gamma mixing distribution.

"lnorm" Lognormal mixing distribution.

"ig" Inverse Gaussian mixing distribution.

"weibull" Weibull mixing distribution.

"loglogistic" Log-logistic mixing distribution.

"betaprime" Beta-prime mixing distribution.

If var.power = NULL, each distribution uses its native/default variance-power specification. Otherwise, every requested mixing distribution is crossed with the requested variance powers, following the candidate construction used by select_zimp().

Marginal candidates are ranked by decreasing maximized univariate log-likelihood. At most margin.top models are retained from each margin. If m_1 and m_2 models are retained from margins 1 and 2, respectively, dependence selection is performed for

$$m_1 m_2$$

marginal-model pairs.

Thus, margin.top = 1 restricts the dependence search to the independently best marginal model from each margin.

Mixing-copula candidates

Dependence between the two latent mixing variables is specified using bivariate `VineCopula` family codes. `mixing.copula` may contain one or more candidate families.

Supported family codes are:

`0L` Independence copula.

`1L` Gaussian copula.

`2L` Student-t copula.

`3L` Clayton copula.

`4L` Gumbel copula.

`5L` Frank copula.

`6L` Joe copula.

`7L` BB1 copula.

`8L` BB6 copula.

`9L` BB7 copula.

`10L` BB8 copula.

`13L`, `14L`, `16L–20L` 180-degree rotations of the corresponding Archimedean and BB families.

`23L`, `24L`, `26L–30L` 90-degree rotations of the corresponding Archimedean and BB families.

`33L`, `34L`, `36L–40L` 270-degree rotations of the corresponding Archimedean and BB families.

`104L`, `114L`, `124L`, `134L`, `204L`, `214L`, `224L`, `234L` Supported Tawn families and their rotations.

The default is `mixing.copula = 1L`, corresponding to the Gaussian copula.

For example,

```
mixing.copula = 1L             # Gaussian only
mixing.copula = c(1L, 3L, 5L) # Gaussian, Clayton, and Frank
```

For the Student-t copula (`2L`), the correlation parameter is optimized for each degrees-of-freedom value supplied in `t.df.grid`, and the likelihood-maximizing specification is retained.

Gaussian dependence prescreen

For every retained marginal pair, a Gaussian checkerboard copula is fitted first to the latent mixing variables. This fit is used only as a prescreen for the sign of the mixing dependence.

If the absolute fitted Gaussian correlation exceeds `gaussian.sign.tol`, user-requested copula families incompatible with that dependence direction are removed before the full dependence search. If the fitted correlation is not finite or does not exceed the tolerance, no sign restriction is imposed.

If sign filtering would remove all user-requested families, all requested families are restored. Consequently, the prescreen never introduces a mixing-copula family that was not requested by the user.

The Gaussian family is fitted internally for this prescreen even when `1L` is not included in `mixing.copula`, but it is eligible for final selection only when explicitly requested.

Zero dependence

When both retained margins are ZIMP models, dependence between their Bernoulli activation indicators is also selected. For every compatible mixing copula, the following three zero-dependence specifications are evaluated:

Gaussian Gaussian copula dependence with an estimated correlation parameter.

Frechet upper Comonotone dependence at the upper Frechet bound.

Frechet lower Countermonotone dependence at the lower Frechet bound.

Thus, for a ZIMP–ZIMP marginal pair, the full dependence search crosses each compatible mixing-copula family with all three zero-dependence specifications.

If either margin is an MP model, the zero-dependence layer is inactive.

Final model selection

The complete candidate set consists of the fitted combinations of

retained margin 1 $\times$ retained margin 2 $\times$ compatible mixing copula $\times$ admissible zero dependence.

Candidates are evaluated using the full bivariate observed-data likelihood. `candidate_table` is sorted by decreasing maximized log-likelihood, and its first row corresponds to the selected model.

AIC and BIC are also calculated for each candidate. Their parameter counts include both marginal models, the mixing-copula parameters, and the Gaussian zero-copula correlation parameter when applicable. AIC and BIC are reported for comparison but are not used for model selection.

Parallel computation

`parallel` is passed to `select_zimp()` and controls parallel fitting of candidate marginal models within each margin.

The two marginal selections themselves are performed sequentially. Dependence searches across retained marginal pairs, mixing copulas, and zero-dependence specifications are also performed sequentially by `select_bizimp()`.

See Also

`fit_bizimp()` for fitting one specified bivariate model, `select_zimp()` for univariate marginal model selection, and `fit_zimp()` for fitting one specified univariate marginal model.

Examples

```
## Generate a small bivariate Mixed Poisson data set
set.seed(12345)
n <- 30
x1 <- rnorm(n)
x2 <- rnorm(n)
X <- cbind(1, x1, x2)
```

```
mu1 <- exp(X %*% c(0.5, 0.3, 0.7))
mu2 <- exp(X %*% c(0.3, 0.5, 0.4))

## Gamma-mixed Poisson margins can be generated from negative-binomial
## distributions for this short reproducible example.
y1 <- rnbinom(n, size = 1 / 0.2, mu = mu1)
y2 <- rnbinom(n, size = 1 / 0.3, mu = mu2)
Y <- cbind(y1, y2)

## Compare several mixing-copula families while retaining only the
## independently best marginal model from each margin.
sel <- select_bizimp(
  Y = Y,
  X = X,
  Z = NULL,
  mixing.distribution = "gamma",
  var.power = 2,
  margin.top = 1L,
  mixing.copula = c(1L, 3L, 5L)
)

## Candidate models are ordered by decreasing log-likelihood
sel$candidate_table

## Selected marginal and dependence models
sel$best_margin_1
sel$best_margin_2
sel$best_model
```

`select_multi_margin` *Select Marginal Models for Multivariate Count Data*

Description

Fits and selects a univariate Mixed Poisson (MP) or Zero-Inflated Mixed Poisson (ZIMP) model independently for each margin of a multivariate count response.

Candidate models may differ in mixing distribution, variance-power specification, and, when requested, the presence or absence of zero inflation. The best model for each margin is selected using AIC or BIC.

`select_multi_margin()` is designed primarily as the marginal-selection stage preceding `fit_multizimp()`. It supports common or margin-specific count and zero-component designs, serial or parallel computation, and optional precomputation of the checkerboard interval-probability matrices required by the multivariate dependence model.

Usage

```
select_multi_margin(
  Y,
```

```
  X,
  Z = NULL,
  X.columns = NULL,
  Z.columns = NULL,
  check.zero = FALSE,
  selection.criterion = c("BIC", "AIC"),
 mixing.distribution = c("gamma", "lnorm", "ig", "weibull", "loglogistic", "betaprime"),
  var.power = NULL,
  benchmark.distribution = "lnorm",
  ini.par = NULL,
  parallel = NULL,
  block_size = NULL,
  compute.I = FALSE,
  I.d0 = 10L,
  progress = FALSE,
  show.result = FALSE,
  trace = 0,
  eval.max = 100,
  iter.max = 100,
  rel.tol = 1e-05,
  x.tol = 1e-05,
  n_gh = 20,
  gh = NULL,
  seed = NULL
)
```

Arguments

Y	An $n \times k$ numeric matrix or data frame containing multivariate count responses. All entries must be finite, non-negative, and integer-valued. Columns represent response margins.
X	Count-component design. Either a numeric matrix/data frame shared by all margins or a list of length k containing margin-specific design matrices/data frames. Intercepts must be supplied explicitly.
Z	Optional zero-component design. May be NULL, a matrix/data frame shared by all margins, or a list of length k. For list input, Z[[j]] = NULL specifies an MP margin and a non-NULL element specifies a ZIMP margin.
X.columns	Optional list of length k specifying which columns of a common X are used for each margin. Elements may contain column names or column indices. A NULL element uses all columns. Ignored when X is supplied as a list.
Z.columns	Optional list of length k specifying which columns of a common Z are used for each margin. Elements may contain column names or column indices. A NULL element uses all columns. Ignored when Z is supplied as a list.
check.zero	Logical. If TRUE and Z = NULL, both MP and intercept-only ZIMP specifications are considered for every candidate marginal model. If Z is explicitly supplied, check.zero is ignored. The default is FALSE.

`selection.criterion`
: Character string specifying the information criterion used to select the best model independently for each margin. Either `"BIC"` or `"AIC"`. The default is `"BIC"`.

`mixing.distribution`
: Character vector specifying candidate mixing distributions. Supported values are `"gamma"`, `"lnorm"`, `"ig"`, `"weibull"`, `"loglogistic"`, and `"betaprime"`. By default, all supported distributions are considered.

`var.power`
: `NULL` or a numeric vector containing candidate variance powers. Values must be selected from `1`, `2`, and `3`. If `NULL`, each distribution uses its native/default variance-power parameterization.

`benchmark.distribution`
: Character string specifying the benchmark marginal distribution used for reporting. Supported values are `"lnorm"`, `"gamma"`, and `"ig"`. The default is `"lnorm"`.

`ini.par`
: Optional initial parameter specification. It may be a list of length `k` containing one initial parameter vector per margin, or a named list indexed by mixing distribution whose elements are themselves lists of length `k`.

`parallel`
: `NULL` or a positive integer specifying the requested number of PSOCK workers. `NULL` or `1` uses serial execution.

`block_size`
: `NULL` or a positive integer controlling the number of fitting jobs grouped into each load-balanced parallel block. If `NULL`, the block size is chosen automatically.

`compute.I`
: Logical. If `TRUE`, precompute checkerboard interval-probability matrices for the selected marginal models. The default is `FALSE`.

`I.d0`
: Integer greater than or equal to `2` specifying the checkerboard grid size when `compute.I = TRUE`. The default is `10L`.

`progress`
: Deprecated and ignored. Retained for API compatibility. `select_multi_margin()` is silent.

`show.result`
: Deprecated and ignored. Retained for API compatibility. Selection results are stored in the returned object.

`trace`
: Deprecated and ignored. Retained for API compatibility. `select_multi_margin()` is silent.

`eval.max`
: Positive integer giving the maximum number of objective-function evaluations for each marginal fit.

`iter.max`
: Positive integer giving the maximum number of optimizer iterations for each marginal fit.

`rel.tol`
: Positive numeric value giving the relative convergence tolerance for marginal fitting.

`x.tol`
: Positive numeric value giving the parameter convergence tolerance for marginal fitting.

`n_gh`
: Positive integer giving the number of Gauss–Hermite quadrature nodes used when required by marginal calculations. The default is `20`.

`gh`
: Optional precomputed Gauss–Hermite quadrature object. If `NULL`, the required quadrature rule is constructed internally.

`seed`
: Optional integer seed used to initialize reproducible random-number streams in parallel execution.

Details

Let

$$\mathbf{Y} = (Y_1, \ldots, Y_k)$$

denote an $n \times k$ multivariate count response. The function fits a collection of candidate univariate models separately to each margin Y_j and selects the best marginal model according to `selection.criterion`.

Marginal selection does not estimate dependence between response margins. Dependence estimation is performed subsequently by `fit_multizimp()`.

Value

An object of class `"multi_margin_fit"` containing the marginal-selection results and the information required by `fit_multizimp()`. The public components are:

`Y` Validated $n \times k$ response matrix.

`X`, `Z` Count- and zero-component designs after internal conversion.

`X_is_list`, `Z_is_list` Logical indicators identifying whether the corresponding designs are stored as margin-specific lists.

`X_columns`, `Z_columns` Margin-specific column selections when common designs were supplied.

`selected_margins` List of length `k` containing the selected marginal models. Each element has class `"mzimp_margin_fit"` and contains, among other quantities, the selected mixing distribution, variance power, zero-inflation status, count coefficients (`be`), dispersion parameter (`phi`), zero-component coefficients (`gam`) when applicable, and maximized marginal log-likelihood.

`benchmark.distribution` Mixing distribution used as the benchmark.

`benchmark_by_margin` Benchmark-fit information for individual response margins.

`benchmark` Summary of the comparison between the selected models and the benchmark, including marginal and aggregate log-likelihood gains.

`selected_summary` Data frame with one row per response margin describing the selected distribution, variance power, zero-inflation status, log-likelihood, AIC, BIC, benchmark log-likelihood, and gain relative to the benchmark.

`I_precomputed` Logical indicating whether the checkerboard probability cache was computed.

`I_d0` Checkerboard grid size used for the cache, or `NULL` when `compute.I = FALSE`.

`I_y_list`, `I_0_list` Lists of checkerboard interval-probability matrices evaluated at observed counts and at zero, respectively, when `compute.I = TRUE`; otherwise `NULL`.

`pb_list` List of Bernoulli activation-probability vectors when `compute.I = TRUE`; otherwise `NULL`.

`y0_list` List of observed-zero indicators for ZIMP margins when `compute.I = TRUE`. Elements corresponding to MP margins are `NULL`.

`mixing.distribution`, `var.power` Candidate marginal specifications requested by the user.

`candidate_grid` Data frame describing the candidate marginal specifications participating in model selection.

`check.zero` Logical indicating whether MP versus intercept-only ZIMP comparison was requested when `Z = NULL`.

`selection.criterion` Information criterion used for marginal model selection.

`n`, `k` Number of observations and response margins.

`response_names` Names of the response margins.

`parallel`, `n_workers` Requested parallel setting and the actual number of workers used.

`performance` Timing and, when available, memory-use information for marginal fitting and optional checkerboard-cache computation.

`summary` A regular list containing `selected_summary`, benchmark information, selection criterion, response dimensions and names, and performance information.

Response and design specification

`Y` must be an $n \times k$ numeric matrix or data frame containing finite, non-negative integer counts. Each column represents one response margin.

The count-component design `X` can be supplied either as:

- one matrix or data frame shared by all margins; or
- a list of length `k` containing a separate design for each margin.

When a common `X` is supplied, `X.columns` may be used to specify a different subset of columns for each margin.

The zero-component design `Z` follows the same convention. It may be `NULL`, a common matrix/data frame, or a list of length `k`.

No intercept is added automatically to either `X` or `Z`. Include an intercept column explicitly when required.

Data-frame designs may contain numeric, integer, logical, factor, and character variables. Character variables are converted to factors and categorical variables are expanded internally into numeric design columns.

MP and ZIMP margins

The interpretation of `Z` determines whether a margin is fitted as an ordinary MP model or as a ZIMP model.

`Z = NULL`, `check.zero = FALSE` All margins are fitted as ordinary MP models.

Common matrix/data frame `Z` All margins are fitted as ZIMP models using the supplied zero-component design.

`Z` **supplied as a list** Each margin is specified independently. `Z[[j]] = NULL` gives an MP margin, whereas a non-`NULL` `Z[[j]]` gives a ZIMP margin.

`Z = NULL`, `check.zero = TRUE` For every marginal distribution and variance-power specification, both an MP model and an intercept-only ZIMP model are considered. The presence or absence of zero inflation is therefore selected independently for each margin.

If `Z` is explicitly supplied, `check.zero` is ignored.

For a ZIMP margin, the zero-component regression models the Bernoulli activation probability. If B_i denotes the activation indicator,

$$\Pr(B_i = 1) = \text{logit}^{-1}(z_i^\top \gamma).$$

Thus, $B_i = 1$ corresponds to the count-generating component and $B_i = 0$ to a structural zero.

Candidate marginal models

The following mixing distributions are supported:

`"gamma"` Gamma mixing distribution.

`"lnorm"` Lognormal mixing distribution.

`"ig"` Inverse-Gaussian mixing distribution.

`"weibull"` Weibull mixing distribution.

`"loglogistic"` Log-logistic mixing distribution.

`"betaprime"` Beta-prime mixing distribution.

If `var.power` is supplied, every requested mixing distribution is crossed with every requested variance power. For example,

```
mixing.distribution = c("gamma", "lnorm")
var.power = c(1, 2, 3)
```

defines six distribution–variance-power specifications before any MP/ZIMP expansion.

If `var.power = NULL`, each distribution uses its native/default variance-power parameterization:

Gamma 2

Lognormal 2

Inverse Gaussian 3

Weibull 2

Log-logistic 2

Beta-prime 2

The complete candidate specifications used for marginal selection are returned in `candidate_grid`.

Marginal model selection

Candidate models are fitted independently for each response margin. The selected model minimizes the information criterion specified by `selection.criterion`.

For `selection.criterion = "BIC"`,

$$\text{BIC} = -2\ell + \log(n)q,$$

whereas for `selection.criterion = "AIC"`,

$$\mathrm{AIC} = -2\ell + 2q,$$

where ℓ is the maximized marginal log-likelihood and q is the number of estimated marginal parameters.

Selection is performed separately for every margin. Consequently, different response margins may select different mixing distributions, variance powers, and, when `check.zero = TRUE`, different MP/ZIMP specifications.

Candidate fitting uses the same initialization strategy as `select_zimp()`. Gamma, lognormal, and inverse-Gaussian models use distribution-specific initialization. Weibull, log-logistic, and beta-prime models use fitted Gamma models with the same effective variance power to initialize their regression coefficients, while retaining distribution-specific initialization for dispersion.

Gamma models required only for initialization are internal fits and do not participate in final model selection unless the corresponding Gamma specification was explicitly requested.

Benchmark model

`benchmark.distribution` specifies a reference marginal distribution used to compare the selected marginal models. Supported benchmark distributions are `"lnorm"`, `"gamma"`, and `"ig"`.

The benchmark is fitted using its native/default variance-power parameterization under the corresponding zero-inflation specification. It is used only for reporting and does not affect marginal model selection.

The returned object contains the benchmark log-likelihood for each margin and the log-likelihood gain of the selected model relative to the benchmark.

Parallel computation

Candidate marginal models can be fitted in parallel.

If `parallel = NULL` or `parallel = 1`, fitting is serial. If `parallel` is an integer greater than one, a PSOCK cluster is created and marginal-model fitting jobs are distributed across workers.

Parallelization is across independent margin–candidate fitting jobs. The number of workers actually used is bounded by the number of available fitting tasks.

`block_size` optionally controls how fitting jobs are grouped for load-balanced parallel execution. If `block_size = NULL`, an appropriate block size is selected automatically.

Serial and parallel execution use the same model-selection procedure and return the same public result structure.

Checkerboard probability cache

If `compute.I = TRUE`, checkerboard interval probabilities are precomputed for each selected marginal model for subsequent use by `fit_multizimp()`.

For each margin j, the cache contains:

`I_y_list[[j]]` An $n \times d$ matrix of interval probabilities evaluated at the observed counts, where d is `I.d0`.

`I_0_list[[j]]` An $n \times d$ matrix of interval probabilities evaluated at zero.

`pb_list[[j]]` Bernoulli activation probabilities. For an MP margin this is a vector of ones. For a ZIMP margin it is $\text{logit}^{-1}(Z_j\gamma_j)$.

`y0_list[[j]]` Logical indicator of observed zeros for a ZIMP margin. It is `NULL` for an MP margin.

The checkerboard resolution d is controlled by `I.d0`.

If `compute.I = FALSE`, no cache is created and `I_d0`, `I_y_list`, `I_0_list`, `pb_list`, and `y0_list` are returned as `NULL`.

See Also

`fit_multizimp()` for fitting the multivariate dependence model after marginal selection, `select_zimp()` for selecting a univariate marginal model, and `fit_zimp()` for fitting one specified univariate model.

Examples

```
set.seed(12345)

n <- 30
k <- 3
x1 <- rnorm(n)
x2 <- rnorm(n)

X <- cbind(Intercept = 1, x1 = x1, x2 = x2)

Y <- cbind(
  y1 = rpois(n, exp(0.5 + 0.3 * x1 + 0.2 * x2)),
  y2 = rpois(n, exp(0.3 - 0.2 * x1 + 0.3 * x2)),
  y3 = rpois(n, exp(0.4 + 0.1 * x1 - 0.2 * x2))
)

## Select among Gamma and lognormal marginal models
fit <- select_multi_margin(
  Y = Y,
  X = X,
  Z = NULL,
  mixing.distribution = c("gamma", "lnorm"),
  var.power = 2,
  selection.criterion = "BIC",
  parallel = 1
)

## One selected model per response margin
fit$selected_summary

## Selected marginal model objects
fit$selected_margins
```

`select_zimp` *Select a Univariate Mixed Poisson or Zero-Inflated Mixed Poisson Model*

Description

Fits and compares multiple univariate Mixed Poisson (MP) or Zero-Inflated Mixed Poisson (ZIMP) regression models across candidate mixing distributions and variance-power specifications. Candidate models are ranked by maximized log-likelihood, and the model with the largest log-likelihood is selected.

Usage

```
select_zimp(
  formula = NULL,
  data = NULL,
  Y = NULL,
  X = NULL,
  Z = NULL,
  offset.count = NULL,
 mixing.distribution = c("gamma", "lnorm", "ig", "weibull", "loglogistic", "betaprime"),
  var.power = NULL,
  n_gh = 20,
  gh = NULL,
  ini.par = NULL,
  parallel = NULL,
  progress = TRUE,
  show.result = TRUE,
  trace = 0,
  eval.max = 100,
  rel.tol = 1e-05,
  x.tol = 1e-05,
  iter.max = 100,
  .return.all = FALSE
)
```

Arguments

`formula`	Optional model formula. For an MP model, use a formula such as `y ~ 1 + x1 + x2`. For a ZIMP model, use a two-part formula such as `y ~ 1 + x1 + x2 | 1 + z1 + z2`. Intercepts must be included explicitly using `1`. When `formula` is supplied, `data` must also be supplied.
`data`	Optional data frame containing the variables referenced in `formula`.
`Y`	Numeric vector containing the count response for the matrix interface. Values must be finite, non-negative integers.

`X` Numeric matrix or data frame containing the design variables for the count component when using the matrix interface. It must have `length(Y)` rows. Include a column of ones explicitly if an intercept is required.

`Z` Optional numeric matrix or data frame containing the design variables for the zero-inflation component when using the matrix interface. It must have `length(Y)` rows. Include a column of ones explicitly if an intercept is required. If `NULL`, an MP model is fitted.

`offset.count` Optional numeric offset for the count linear predictor. For the matrix interface, it must be a finite numeric vector of length `length(Y)`. If `NULL`, no count offset is used. With the formula interface, an offset may be specified in the count component using `offset()`.

`mixing.distribution`
Character vector specifying one or more candidate mixing distributions. Supported values are `"gamma"`, `"lnorm"`, `"ig"`, `"weibull"`, `"loglogistic"`, and `"betaprime"`. By default, all supported distributions are considered.

`var.power` Optional numeric vector specifying one or more variance-power parameterizations. Values must be selected from `1`, `2`, and `3`. If `NULL`, each requested mixing distribution is fitted using its native/default variance-power specification.

`n_gh` Positive integer specifying the number of Gauss–Hermite quadrature nodes when Gauss–Hermite integration is required. The default is `20`.

`gh` Optional precomputed Gauss–Hermite quadrature object. If `NULL`, the required quadrature rule is constructed internally.

`ini.par` Optional numeric vector of initial parameter values. Its length must be `ncol(X) + 1` for an MP model and `ncol(X) + 1 + ncol(Z)` for a ZIMP model. The parameter order is `c(beta, log(phi), gamma)`.

`parallel` `NULL` or a positive integer specifying the number of workers requested for parallel candidate-model fitting. `NULL` or `1` uses serial execution. The number of workers actually used in a fitting stage cannot exceed the number of candidate jobs in that stage.

`progress` Logical indicating whether fitting progress should be displayed. The default is `TRUE`.

`show.result` Logical compatibility argument. Selection is performed silently and results are stored in the returned object. The selection report can be displayed by evaluating or printing the `summary` component.

`trace` Numeric optimizer trace level. The default is `0`.

`eval.max` Positive integer giving the maximum number of objective function evaluations. The default is `100`.

`rel.tol` Positive numeric value giving the relative convergence tolerance. The default is `1e-5`.

`x.tol` Positive numeric value giving the parameter convergence tolerance. The default is `1e-5`.

`iter.max` Positive integer giving the maximum number of optimizer iterations. The default is `100`.

`.return.all` Logical. Internal-use option. If `FALSE`, the default, only the compact public selection object is returned. If `TRUE`, candidate fit objects are additionally retained in `all_results`. Users should normally leave this argument at its default value.

Details

`select_zimp()` is the model-selection counterpart of `fit_zimp()`. While `fit_zimp()` fits one fully specified model, `select_zimp()` fits a collection of candidate models and returns the best-fitting specification.

The function supports the same formula and matrix interfaces as `fit_zimp()`.

Value

An object of class `"zimp_selection"`. By default, the object contains:

`summary_table` A data frame containing one row for each requested candidate model, ordered by decreasing maximized log-likelihood. Columns include `distribution`, `var.power`, `log_lik`, `neg_log_lik`, and `phi`.

`best_model` The selected model, returned as an object of class `"zimp_fit"`. It corresponds to the first row of `summary_table` and contains full post-estimation inference.

`summary` An object of class `"zimp_selection_summary"` containing the candidate table and selected model for formatted display.

`call` The matched call to `select_zimp()`.

If `.return.all = TRUE`, an additional internal component `all_results` containing the candidate fit objects is returned.

Model specification

Formula interface

A Mixed Poisson model can be specified as

```
y ~ x1 + x2
```

and a Zero-Inflated Mixed Poisson model as

```
y ~ x1 + x2 | z1 + z2
```

The expression before `|` specifies the count component and the expression after `|` specifies the zero-inflation component.

In the `mzimp` formula interface, intercepts are included only when explicitly requested with `1`. Thus,

```
y ~ x1 + x2
```

specifies a model without an intercept, whereas

```
y ~ 1 + x1 + x2
```

includes an intercept. The same convention applies independently to the zero-inflation component.

Matrix interface

The response and design matrices may instead be supplied through `Y`, `X`, and optionally `Z`. If `Z = NULL`, MP candidate models are fitted. If `Z` is supplied, ZIMP candidate models are fitted. Intercept columns, when required, must be included explicitly in `X` and `Z`.

Candidate models

Candidate models are generated from the requested `mixing.distribution` and `var.power` values.

The supported mixing distributions are `"gamma"`, `"lnorm"`, `"ig"`, `"weibull"`, `"loglogistic"`, and `"betaprime"`.

If `var.power = NULL`, one model is fitted for each requested mixing distribution using its native/default variance-power specification:

`"gamma"` `var.power = 2`

`"lnorm"` `var.power = 2`

`"ig"` `var.power = 3`

`"weibull"` `var.power = 2`

`"loglogistic"` `var.power = 2`

`"betaprime"` `var.power = 2`

If one or more values are supplied to `var.power`, every requested mixing distribution is fitted for each requested variance power. For example,

```
mixing.distribution = c("gamma", "lnorm")
var.power = c(1, 2, 3)
```

defines six candidate models.

Duplicate distribution and variance-power specifications are removed before fitting.

Model fitting and selection

Candidate models are fitted by maximum likelihood using the same distribution-specific likelihood and gradient routines as `fit_zimp()`.

Gamma, lognormal, and inverse-Gaussian candidates are fitted directly. Weibull, log-logistic, and beta-prime candidates use matched Gamma models as initialization anchors. For these candidates, the count coefficients and, for ZIMP models, the zero-component coefficients are initialized from a Gamma model with the same effective variance power. The dispersion parameter is initialized using the target distribution's own initialization procedure.

When a required Gamma anchor is not among the requested candidate models, it is fitted internally for initialization only and is not included in the returned candidate table.

Candidate models are initially fitted without post-estimation inference. They are then ordered by decreasing maximized log-likelihood. The first row of `summary_table` therefore corresponds to the selected model.

Full post-estimation inference is subsequently computed only for the selected model, at its already optimized parameter estimates; the selected model is not re-optimized. The resulting `best_model` is an object of class `"zimp_fit"`.

Selection is based exclusively on maximized log-likelihood. AIC and BIC are available for the selected model through `best_model`, but are not used to rank candidate models.

Parallel computation

Candidate models within the same fitting stage may be evaluated in parallel by setting `parallel` to an integer greater than one.

Parallelization is performed across candidate models; optimization of an individual candidate is not itself parallelized by `select_zimp()`. Dependencies between fitting stages, including construction of Gamma initialization anchors, are preserved.

On Unix-like systems, fork-based parallel processing may be used. On Windows, PSOCK workers are used.

See Also

`fit_zimp()` for fitting one specified univariate MP or ZIMP model.

Examples

```
## Generate a small Mixed Poisson data set
set.seed(12345)
n <- 50
x1 <- rnorm(n)
x2 <- rnorm(n)
X <- cbind(1, x1, x2)
mu <- exp(X %*% c(0.5, 0.3, 0.7))

## Gamma-mixed Poisson data
phi <- 0.2
y <- rnbinom(n, size = 1 / phi, mu = mu)

dat <- data.frame(y = y, x1 = x1, x2 = x2)

## Compare candidate mixing distributions
sel <- select_zimp(
  y ~ 1 + x1 + x2,
  data = dat,
  mixing.distribution = c("gamma", "lnorm", "ig"),
  progress = FALSE
)

## Ranked candidate models
sel$summary_table

## Selected model
sel$best_model

## Compare variance-power specifications
sel_power <- select_zimp(
  Y = y,
  X = X,
  mixing.distribution = c("gamma", "lnorm"),
  var.power = c(1, 2, 3),
  progress = FALSE
)
```

```
sel_power$summary_table
```

Index

fit_bizimp, 2
fit_bizimp(), *15*, *24*, *28*
fit_multizimp, 9
fit_multizimp(), *29*, *32*, *35, 36*
fit_zimp, 16
fit_zimp(), *2*, *4–8*, *15*, *28*, *36*, *39–41*

print.bizimp_selection_summary, 21
print.zimp_selection_summary, 22

select_bizimp, 22
select_bizimp(), *4*, *8*
select_multi_margin, 29
select_multi_margin(), *9, 10*, *13–15*
select_zimp, 37
select_zimp(), *15*, *18*, *20*, *22–26*, *28*, *35, 36*