Package: RmlxStats (via r-universe)

June 3, 2026

Title MLX-Accelerated Statistical Models

Version 0.3.0

Description Fast statistical routines on Apple Silicon using the Rmlx package.

License GPL (>= 2)

URL https://hughjonesd.github.io/RmlxStats, https://github.com/hughjonesd/RmlxStats

BugReports https://github.com/hughjonesd/RmlxStats/issues

Imports Rmlx (>= 0.4.0), generics

Remotes hughjonesd/Rmlx

Suggests glmnet, knitr, lmtest, sandwich, testthat (>= 3.0.0), withr

Config/testthat/edition 3

Config/Needs/website knitr, rmarkdown, ggplot2, nycflights13, bench, boot, bigstatsr, car, fixest, irlba, RcppEigen, rsvd, speedglm, fastglm, lmboot, huxtable

Encoding UTF-8

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.0.0

Config/pak/sysreqs cmake

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

Date/Publication 2026-05-26 15:03:33 UTC

RemoteUrl https://github.com/hughjonesd/RmlxStats

RemoteRef v0.3.0

RemoteSha 36963b9c305b2487d6f93264c34d1c37bd0d5ea5

Contents

`generics-reexports` *Re-export generics*

Description

These generics are re-exported from the generics package for convenience.

Usage

```
tidy(x, ...)

glance(x, ...)

augment(x, ...)
```

Arguments

`x, ...`	Passed to the generic.

mlxs-glm-methods *mlxs_glm method utilities*

Description

Support functions that provide a familiar S3 surface for `mlxs_glm` fits by delegating to equivalent base `glm` behaviour where helpful.

Usage

```
## S3 method for class 'mlxs_glm'
weights(object, type = c("prior", "working"), ...)

## S3 method for class 'mlxs_glm'
predict(
  object,
  newdata = NULL,
  type = c("link", "response"),
  se.fit = FALSE,
  ...
)

## S3 method for class 'mlxs_glm'
fitted(object, ...)

## S3 method for class 'mlxs_glm'
residuals(object, type = c("deviance", "pearson", "working", "response"), ...)

## S3 method for class 'mlxs_glm'
vcov(object, ...)

## S3 method for class 'mlxs_glm'
confint(
  object,
  parm,
  level = 0.95,
  ...,
  bootstrap = FALSE,
 bootstrap_args = list(B = 200L, seed = NULL, progress = FALSE, bootstrap_type = "case")
)

## S3 method for class 'mlxs_glm'
print(x, digits = max(3, getOption("digits") - 3), ...)

## S3 method for class 'mlxs_glm'
summary(
  object,
```

```
  bootstrap = FALSE,
 bootstrap_args = list(B = 200L, seed = NULL, progress = FALSE, bootstrap_type = "case"),
  confint = FALSE,
  level = 0.95,
  ...
)

## S3 method for class 'summary.mlxs_glm'
print(x, digits = max(3, getOption("digits") - 3), ...)

## S3 method for class 'mlxs_glm'
anova(object, ...)

## S3 method for class 'mlxs_glm'
model.frame(formula, ...)

## S3 method for class 'mlxs_glm'
model.matrix(object, ...)

## S3 method for class 'mlxs_glm'
terms(x, ...)

## S3 method for class 'mlxs_glm'
nobs(object, ...)

## S3 method for class 'mlxs_glm'
tidy(x, ...)

## S3 method for class 'mlxs_glm'
glance(x, ...)

## S3 method for class 'mlxs_glm'
augment(
  x,
  data = x$model,
  newdata = NULL,
  type.predict = c("response", "link"),
  type.residuals = c("response", "deviance"),
  se_fit = FALSE,
  output = c("data.frame", "mlx"),
  ...
)

## S3 method for class 'mlxs_glm'
estfun(x, ..., output = c("matrix", "mlx"))

## S3 method for class 'mlxs_glm'
hatvalues(model, ..., output = c("matrix", "mlx"))
```

```
## S3 method for class 'mlxs_glm'
bread(x, ...)
```

Arguments

object, model	An mlxs_glm model fit.
type	Character string indicating the scale of the prediction or residuals to return.
...	Additional arguments passed to underlying methods.
newdata	Optional data frame used for prediction.
se.fit	Logical. Should standard errors of the fit be returned when supported?
parm	Parameter specification for confidence intervals.
level	Confidence level for intervals.
bootstrap	Logical; should bootstrap standard errors or confidence intervals be computed?
bootstrap_args	List of bootstrap configuration options. See mlxs_boot().
x	An mlxs_glm model fit (for methods with a leading x argument).
digits	Number of significant digits to print for summaries.
confint	Logical; should confidence intervals be included in the summary object?
formula, data	Optional formula and data overrides used by augment.mlxs_glm().
type.predict, type.residuals	Character strings controlling the scale of fitted values and residuals returned by augment.mlxs_glm().
se_fit	Logical; standard-error analogue for augment.
output	Character string; return format ("data.frame" or "mlx").

---

mlxs-lm-methods *mlxs_lm method utilities*

Description

These helpers provide the familiar S3 surface for mlxs_lm fits.

Usage

```
## S3 method for class 'mlxs_lm'
predict(object, newdata = NULL, ...)

## S3 method for class 'mlxs_lm'
fitted(object, ...)

## S3 method for class 'mlxs_lm'
residuals(object, ...)
```

```
## S3 method for class 'mlxs_lm'
vcov(object, ...)

## S3 method for class 'mlxs_lm'
confint(
  object,
  parm,
  level = 0.95,
  ...,
  bootstrap = FALSE,
 bootstrap_args = list(B = 200L, seed = NULL, progress = FALSE, bootstrap_type = "case")
)

## S3 method for class 'mlxs_lm'
anova(object, ...)

## S3 method for class 'mlxs_anova'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)

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

## S3 method for class 'mlxs_anova'
tidy(x, ...)

## S3 method for class 'mlxs_lm'
summary(
  object,
  bootstrap = FALSE,
 bootstrap_args = list(B = 200L, seed = NULL, progress = FALSE, bootstrap_type = "case"),
  confint = FALSE,
  level = 0.95,
  ...
)

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

## S3 method for class 'summary.mlxs_lm'
print(x, ...)

## S3 method for class 'mlxs_lm'
model.frame(formula, ...)

## S3 method for class 'mlxs_lm'
model.matrix(object, ...)

## S3 method for class 'mlxs_lm'
```

```
terms(x, ...)

## S3 method for class 'mlxs_lm'
nobs(object, ...)

## S3 method for class 'mlxs_lm'
tidy(x, ...)

## S3 method for class 'mlxs_lm'
glance(x, ...)

## S3 method for class 'mlxs_lm'
augment(
  x,
  data = model.frame(x),
  newdata = NULL,
  se_fit = FALSE,
  output = c("data.frame", "mlx"),
  ...
)

## S3 method for class 'mlxs_lm'
estfun(x, ..., output = c("matrix", "mlx"))

## S3 method for class 'mlxs_lm'
hatvalues(model, ..., output = c("matrix", "mlx"))

## S3 method for class 'mlxs_lm'
bread(x, ...)
```

Arguments

object, model	An mlxs_lm model fit.
newdata	Optional data frame for prediction.
...	Additional arguments passed to underlying methods.
parm	Parameter specification for confidence intervals.
level	Confidence level for intervals.
bootstrap	Logical; should bootstrap standard errors or confidence intervals be computed?
bootstrap_args	List of bootstrap configuration options. See mlxs_boot().
x	An mlxs_lm model fit (for methods with a leading x argument).
row.names	Optional row names for data frame conversion.
optional	Logical; passed to as.data.frame.
confint	Logical; should confidence intervals be included in the summary object?
formula	An mlxs_lm object used in place of formula for model.frame.
data	Optional data frame for augment.

se_fit	Logical; should standard errors of fit be included?
output	Character string; return format ("data.frame", "matrix", "vector" or "mlx"). To make methods from other packages work, the usual default is to return a base R object.

---

mlxs-model-methods *Shared mlxs model methods*

Description

Methods for behavior shared by `mlxs_lm` and `mlxs_glm` through their `mlxs_model` superclass.

Usage

```
## S3 method for class 'mlxs_model'
update(object, ..., evaluate = TRUE)

## S3 method for class 'mlxs_model'
coef(object, ..., output = c("vector", "mlx"))
```

Arguments

object	An `mlxs_model` fit.
...	Additional arguments passed to underlying methods.
evaluate	Logical; evaluate the updated call?

---

mlxs-prcomp-methods *PCA methods for* `mlxs_prcomp`

Description

`predict.mlxs_prcomp()` returns MLX scores. `summary()` and `plot()` only materialize standard deviations for base-style output; `print()` and `biplot()` materialize rotations and scores as needed for display.

Usage

```
## S3 method for class 'mlxs_prcomp'
predict(object, newdata, ...)

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

## S3 method for class 'mlxs_prcomp'
summary(object, ...)
```

```
## S3 method for class 'mlxs_prcomp'
plot(x, main = deparse1(substitute(x)), ...)

## S3 method for class 'mlxs_prcomp'
biplot(x, ...)

## S3 method for class 'mlxs_prcomp'
nobs(object, ...)

## S3 method for class 'mlxs_prcomp'
tidy(x, ...)

## S3 method for class 'mlxs_prcomp'
augment(x, data = NULL, newdata = NULL, output = c("data.frame", "mlx"), ...)
```

Arguments

object, x	A fitted mlxs_prcomp object.
newdata	Optional new observations to project.
...	Passed through to the corresponding base method.
data	Optional original data to append PCA scores to in augment.mlxs_prcomp().
output	Output format for augment.mlxs_prcomp(): either a data frame with appended score columns or the MLX score matrix directly.

Value

Method-specific output. predict.mlxs_prcomp() returns an MLX matrix. augment.mlxs_prcomp() returns either a data frame or MLX matrix.

---

mlxs_binomial *MLX-friendly binomial family*

Description

Construct a binomial GLM family whose core link and deviance helpers are implemented in R so they work with MLX arrays as well as base R vectors. This avoids calling into compiled C routines that only handle base types.

Usage

```
mlxs_binomial(link = "logit")
```

Arguments

link	a specification for the model link function. This can be a name/expression, a literal character string, a length-one character vector, or an object of class "link-glm" (such as generated by make.link) provided it is not specified *via* one of the standard names given next. The gaussian family accepts the links (as names) identity, log and inverse; the binomial family the links logit, probit, cauchit, (corresponding to logistic, normal and Cauchy CDFs respectively) log, identity and cloglog (complementary log-log); the Gamma family the links inverse, identity and log; the poisson family the links log, identity, and sqrt; and the inverse.gaussian family the links 1/mu^2, inverse, identity and log. The quasi family accepts the links logit, probit, cloglog, identity, inverse, log, 1/mu^2 and sqrt, and the function power can be used to create a power link function.

Details

Currently the logit, log, cloglog, and cauchit links are supported. For other link specifications, fall back to stats::binomial().

Value

A family object compatible with stats::glm() and mlxs_glm().

---

mlxs_boot *Bootstrap MLX arrays along the first dimension*

Description

mlxs_boot() resamples observations from one or more MLX arrays, calls a user-supplied function on each resampled batch, and returns the collected results. Every argument supplied via ... must share the same size in its first dimension (number of observations). Arguments that do not need resampling should be captured in the environment of fun instead of being passed through ....

Usage

```
mlxs_boot(fun, ..., B = 200L, seed = NULL, progress = FALSE, compile = FALSE)
```

Arguments

fun	Function called on each bootstrap draw. It must accept the same named arguments as supplied through ....
...	Arrays, matrices, or vectors that should be resampled along the first dimension before being passed to fun.
B	Number of bootstrap iterations.
seed	Optional integer seed for reproducibility.

progress	Logical; if TRUE, show a text progress bar.
compile	Logical; compile fun once via Rmlx::mlx_compile() before entering the resampling loop. Defaults to FALSE.

Value

A list with elements samples (the raw results from fun), B, and seed.

---

mlxs_cv_glmnet *Cross-validated MLX elastic net regression*

Description

Cross-validation wrapper around mlxs_glmnet() that mirrors the core glmnet::cv.glmnet() workflow for the families currently supported by mlxs_glmnet().

Usage

```
mlxs_cv_glmnet(
  x,
  y,
  weights = NULL,
  offset = NULL,
  lambda = NULL,
  type.measure = c("default", "mse", "deviance", "class", "mae", "auc", "C"),
  nfolds = 10,
  foldid = NULL,
  alignment = c("lambda", "fraction"),
  grouped = TRUE,
  keep = FALSE,
  parallel = FALSE,
  gamma = c(0, 0.25, 0.5, 0.75, 1),
  relax = FALSE,
  trace.it = 0,
  family = mlxs_gaussian(),
  ...
)
```

Arguments

x	Numeric matrix of predictors (observations in rows).
y	Numeric response vector.
weights	Optional observation weights. Currently unsupported.
offset	Optional offset. Currently unsupported.
lambda	Optional decreasing lambda sequence. If NULL, the full-data fit chooses the path and the same path is reused inside each fold.

type.measure	Loss used to score the holdout predictions.
nfolds	Number of folds.
foldid	Optional integer vector giving the fold assignment for each observation.
alignment	Alignment mode. Only "lambda" is currently supported.
grouped	Should cross-validation be aggregated fold-by-fold? Only TRUE is currently supported.
keep	Should out-of-fold predictions be stored?
parallel	Logical. Parallel refits are currently unsupported.
gamma, relax	Relaxed fits are currently unsupported.
trace.it	Progress tracing. Currently unsupported.
family	MLX-aware family object, e.g. mlxs_gaussian() or mlxs_binomial().
...	Additional arguments passed to mlxs_glmnet(), such as alpha, nlambda, lambda_min_ratio, standardize, intercept, maxit, and tol.

Details

The full-data fit defines a master lambda path. Each fold is then refit on the same lambda values and scored on its holdout set.

Current limitations relative to glmnet::cv.glmnet():

- only Gaussian and binomial families are supported
- weights, offset, alignment != "lambda", grouped = FALSE, parallel = TRUE, relax = TRUE, and non-zero trace.it are not implemented
- type.measure = "auc" and type.measure = "C" are not implemented

Value

An object of class mlxs_cv_glmnet.

---

mlxs_gaussian *MLX-friendly Gaussian family*

Description

MLX-friendly Gaussian family

Usage

```
mlxs_gaussian(link = "identity")
```

Arguments

link	a specification for the model link function. This can be a name/expression, a literal character string, a length-one character vector, or an object of class "link-glm" (such as generated by make.link) provided it is not specified *via* one of the standard names given next. The gaussian family accepts the links (as names) identity, log and inverse; the binomial family the links logit, probit, cauchit, (corresponding to logistic, normal and Cauchy CDFs respectively) log, identity and cloglog (complementary log-log); the Gamma family the links inverse, identity and log; the poisson family the links log, identity, and sqrt; and the inverse.gaussian family the links 1/mu^2, inverse, identity and log. The quasi family accepts the links logit, probit, cloglog, identity, inverse, log, 1/mu^2 and sqrt, and the function power can be used to create a power link function.

Value

A family object compatible with mlxs_glm().

---

mlxs_glm *MLX-backed generalized linear model*

Description

Fit generalized linear models using iterative reweighted least squares (IRLS) with MLX providing the heavy lifting for weighted least squares solves. Final convergence is done at double precision on the cpu.

Usage

```
mlxs_glm(
  formula,
  family = mlxs_gaussian(),
  data,
  subset,
  weights,
  na.action = stats::na.exclude,
  start = NULL,
  control = list(),
  ...
)
```

Arguments

formula	Model formula.
family	A mlxs family object (e.g., mlxs_gaussian(), mlxs_binomial(), mlxs_poisson()). You can use "gaussian" etc.

`data`	Optional data frame, tibble, or environment containing the variables in the model.
`subset`	Optional expression for subsetting observations.
`weights`	Optional non-negative observation weights.
`na.action`	How to handle missing values.
`start`	Starting values for the parameters in the linear predictor.
`control`	Optional list of control parameters passed to `mlxs_glm_control()`. Control parameters can include `epsilon`, `epsilon_f64`, `maxit`, `trace`, and `rank_tol`.
`...`	Additional arguments passed to the family function when `family` is supplied as a function or string.

Value

An object of class `c("mlxs_glm", "mlxs_model")` containing elements similar to the result of `stats::glm()`. Unlike `stats::glm()`, rank-deficient model matrices are rejected rather than fit with aliased coefficients.

Examples

```
fit <- mlxs_glm(mpg ~ cyl + disp, family = mlxs_gaussian(), data = mtcars)
coef(fit)
```

`mlxs_glm_control` *Control parameters*

Description

Control parameters

Usage

```
mlxs_glm_control(
  epsilon = 1e-08,
  epsilon_f64 = 1e-06,
  maxit = 25,
  trace = FALSE,
  rank_tol = NULL
)
```

Arguments

`epsilon`	Convergence tolerance parameter, interpreted as in `stats::glm.control()`. Iterations converge when `abs(deviance - deviance_old)/(abs(deviance) + 0.1) < epsilon`.
`epsilon_f64`	Move operations to float64 on the cpu when convergence is this close (using the same expression as above). Doing this allows more precision but slows computation.

maxit	Maximum number of IWLS iterations.
trace	Logical: trace each iteration?
rank_tol	Optional relative tolerance used to detect rank-deficient systems. NULL uses the package default, which varies by dtype and is 1e-6 for float32 matrices. Set to FALSE to skip rank checks entirely. Note that higher numbers indicate *lower* tolerance.

Value

A list with default values filled in.

`mlxs_glmnet` *MLX-backed elastic net regression*

Description

Fit lasso or elastic-net penalised regression paths using MLX arrays for the heavy linear algebra. Dense Gaussian and binomial paths stay on the MLX backend throughout the iterative updates, with repeated chunk updates traced through `Rmlx::mlx_compile()` to reduce host overhead.

Usage

```
mlxs_glmnet(
  x,
  y,
  family = mlxs_gaussian(),
  alpha = 1,
  lambda = NULL,
  nlambda = 100,
  lambda_min_ratio = 1e-04,
  standardize = TRUE,
  intercept = TRUE,
  use_strong_rules = TRUE,
  maxit = 1000,
  tol = 1e-06
)
```

Arguments

x	Numeric R or mlx matrix of predictors.
y	Numeric R or mlx response vector.
family	MLX-aware family object, e.g. `mlxs_gaussian()` or `mlxs_binomial()`.
alpha	Elastic-net mixing parameter (1 = lasso, currently alpha must be strictly positive).
lambda	Optional decreasing sequence of penalty values. If NULL, a sequence of length nlambda is generated from lambda_max down to lambda_max * lambda_min_ratio.

`nlambda`	Length of automatically generated lambda path.
`lambda_min_ratio`	Smallest lambda as a fraction of `lambda_max`.
`standardize`	Should columns of `x` be scaled before fitting?
`intercept`	Should an intercept be fit?
`use_strong_rules`	Retained for API compatibility. The dense MLX solver keeps all coefficients on device, so this flag currently does not change the computation.
`maxit`	Maximum proximal-gradient iterations per lambda value.
`tol`	Convergence tolerance on the coefficient updates.

Value

An object of class `mlxs_glmnet` containing the fitted coefficient path, intercepts, lambda sequence, and scaling information. The `beta` and `a0` components are stored as MLX arrays so downstream methods can keep path computations on the MLX backend. Use `coef()` or `predict()` for ordinary base-R matrix outputs.

Note

`glmnet::glmnet()` is faster on smaller problems. Very roughly as of April 2026, `mlxs_glmnet()` gets competitive at n x p = 5,000,000 or greater.

`mlxs_lm` *MLX-backed linear regression*

Description

Fit a linear model via QR decomposition using MLX arrays on Apple Silicon devices. The interface mirrors `stats::lm()` for the common arguments.

Usage

```
mlxs_lm(
  formula,
  data,
  subset,
  weights,
  na.action = stats::na.exclude,
  rank_tol = NULL
)
```

Arguments

formula	Model formula.
data	Optional data frame, tibble, or environment containing the variables in the model.
subset	Optional expression for subsetting observations.
weights	Optional non-negative observation weights.
na.action	How to handle missing values.
rank_tol	Optional relative tolerance used to detect rank-deficient systems. NULL uses the package default, which varies by dtype and is 1e-6 for float32 matrices. Set to FALSE to skip rank checks entirely. Note that higher numbers indicate *lower* tolerance.

Value

An object of class c("mlxs_lm", "mlxs_model") containing components similar to an "lm" fit, along with MLX intermediates stored in the mlx element. Note that MLX currently operates in single precision, so fitted values and diagnostics may differ from stats::lm() at around the 1e-6 level. Unlike stats::lm(), rank-deficient model matrices are rejected rather than fit with aliased coefficients.

Examples

```
fit <- mlxs_lm(mpg ~ cyl + disp, data = mtcars)
coef(fit)
```

---

mlxs_lm_fit *Fit an MLX linear model from design matrices*

Description

mlxs_lm_fit() powers mlxs_lm() by wrapping the QR-based solver that runs entirely on MLX arrays.

Usage

```
mlxs_lm_fit(x, y, weights = NULL, rank_tol = NULL)
```

Arguments

x	MLX design matrix (or object coercible via Rmlx::as_mlx()) whose rows represent observations and columns represent predictors.
y	MLX column vector (or object coercible via Rmlx::as_mlx()) holding the response values.
weights	Optional MLX column vector or numeric vector of non-negative observation weights. When supplied, weighted least squares are fit via the standard square-root weighting.

rank_tol — Optional relative tolerance used to detect rank-deficient systems. NULL uses the package default, which varies by dtype and is 1e-6 for float32 matrices. Set to FALSE to skip rank checks entirely. Note that higher numbers indicate *lower* tolerance.

Details

Inputs that are not already MLX objects are converted with Rmlx::as_mlx() or Rmlx::mlx_matrix() so callers can provide base-R matrices or vectors. Weighted fits are performed by applying the standard square-root weight transform before solving the QR system.

Value

A list with components coefficients, fitted.values, residuals, effects, and qr, mirroring the corresponding pieces of stats::lm(). Array-valued components remain MLX matrices to keep downstream GPU pipelines in device memory.

Examples

```
x <- Rmlx::as_mlx(cbind(1, as.matrix(mtcars[c("cyl", "disp")])))
y <- Rmlx::mlx_matrix(mtcars$mpg, ncol = 1)
fit <- mlxs_lm_fit(x, y)
drop(as.matrix(fit$coefficients))
```

---

mlxs_poisson *MLX-friendly Poisson family*

Description

MLX-friendly Poisson family

Usage

```
mlxs_poisson(link = "log")
```

Arguments

link — a specification for the model link function. This can be a name/expression, a literal character string, a length-one character vector, or an object of class "link-glm" (such as generated by make.link) provided it is not specified *via* one of the standard names given next.

The gaussian family accepts the links (as names) identity, log and inverse; the binomial family the links logit, probit, cauchit, (corresponding to logistic, normal and Cauchy CDFs respectively) log, identity and cloglog (complementary log-log); the Gamma family the links inverse, identity and log; the poisson family the links log, identity, and sqrt; and the inverse.gaussian family the links 1/mu^2, inverse, identity and log.

The quasi family accepts the links logit, probit, cloglog, identity, inverse, log, 1/mu^2 and sqrt, and the function power can be used to create a power link function.

Value

A family object compatible with mlxs_glm().

---

mlxs_prcomp *MLX-backed principal components analysis*

Description

Perform principal components analysis with MLX arrays, keeping the centred and scaled data on device throughout the decomposition.

Usage

```
mlxs_prcomp(
  x,
  retx = TRUE,
  center = TRUE,
  scale. = FALSE,
  tol = NULL,
  rank. = NULL,
  oversample = NULL,
  n_iter = 2L,
  seed = 1L,
  ...
)
```

Arguments

x	Numeric matrix-like object or MLX array with observations in rows.
retx	Should the rotated scores be returned?
center, scale.	Passed to base::scale(). User-supplied vectors are supported.
tol	Optional tolerance for omitting components with small standard deviations, relative to the leading component.
rank.	Optional maximal rank. If smaller than min(n, p), the fit uses the randomized truncated PCA path.
oversample	Oversampling added to the randomized subspace dimension. If NULL, randomized fits use min(rank., max(10, ceiling(rank. / 2))). Ignored for exact fits.
n_iter	Number of randomized power iterations. Ignored for exact fits.
seed	Seed used for the randomized projection basis. Ignored for exact fits.
...	Additional arguments are rejected for compatibility with stats::prcomp().

Details

The interface follows `stats::prcomp()` closely. Full-rank fits use an exact decomposition. When `rank.` is supplied and smaller than `min(nrow(x), ncol(x))`, a randomized truncated PCA path is used instead.

The randomized path first sketches a slightly larger subspace than the requested rank, then compresses back down to the requested components. The `oversample` parameter controls how much extra space is used in that sketch: larger values make it less likely that the random sketch misses part of the leading principal subspace. The `n_iter` parameter applies additional power iterations, which improve accuracy when the singular values decay slowly but require extra passes over the matrix.

By default, `oversample` is chosen as `min(rank., max(10, ceiling(rank. / 2)))`, which keeps the usual constant-size oversampling for small target ranks while allowing more slack for larger truncated fits. This follows common randomized SVD guidance to start with modest oversampling, often around 5 to 10, and to increase oversampling before increasing the number of power iterations.

Value

An object of class `c("mlxs_prcomp", "prcomp")`.

References

Halko, N., Martinsson, P.-G., and Tropp, J. A. (2011). Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. *SIAM Review*, 53(2), 217-288.

Musco, C. and Musco, C. (2015). Randomized Block Krylov Methods for Stronger and Faster Approximate Singular Value Decomposition. *NeurIPS 2015*.

`mlxs_quasibinomial` *MLX-friendly quasibinomial family*

Description

MLX-friendly quasibinomial family

Usage

```
mlxs_quasibinomial(link = "logit")
```

Arguments

`link`	a specification for the model link function. This can be a name/expression, a literal character string, a length-one character vector, or an object of class `"link-glm"` (such as generated by `make.link`) provided it is not specified *via* one of the standard names given next.

The gaussian family accepts the links (as names) identity, log and inverse; the binomial family the links logit, probit, cauchit, (corresponding to logistic, normal and Cauchy CDFs respectively) log, identity and cloglog (complementary log-log); the Gamma family the links inverse, identity and log; the poisson family the links log, identity, and sqrt; and the inverse.gaussian family the links 1/mu^2, inverse, identity and log.

The quasi family accepts the links logit, probit, cloglog, identity, inverse, log, 1/mu^2 and sqrt, and the function power can be used to create a power link function.

Value

A family object compatible with mlxs_glm().

---

mlxs_quasipoisson *MLX-friendly quasipoisson family*

Description

MLX-friendly quasipoisson family

Usage

```
mlxs_quasipoisson(link = "log")
```

Arguments

link	a specification for the model link function. This can be a name/expression, a literal character string, a length-one character vector, or an object of class "link-glm" (such as generated by make.link) provided it is not specified *via* one of the standard names given next. The gaussian family accepts the links (as names) identity, log and inverse; the binomial family the links logit, probit, cauchit, (corresponding to logistic, normal and Cauchy CDFs respectively) log, identity and cloglog (complementary log-log); the Gamma family the links inverse, identity and log; the poisson family the links log, identity, and sqrt; and the inverse.gaussian family the links 1/mu^2, inverse, identity and log. The quasi family accepts the links logit, probit, cloglog, identity, inverse, log, 1/mu^2 and sqrt, and the function power can be used to create a power link function.

Value

A family object compatible with mlxs_glm().

Index