Package 'MEMWAS'

August 22, 2026

Type Package

Title Mixed-Effects Models with Autocorrelation Structures

Version 0.9.5

Date 2026-08-20

Description Fits longitudinal mixed-effects models through a registered 'C++' numerical backend. Supported serial covariance structures include first-order autoregressive (AR(1)), exponential or Ornstein-Uhlenbeck, higher-order autoregressive (AR(p)), first-order autoregressive moving-average (ARMA(1,1)), compound symmetry, Toeplitz, and unstructured covariance. Serial processes can be unified or attached independently to numeric predictor loadings. Candidate temporal structures can be ranked on a common sample by dependence-component grouped cross-validation, the Akaike information criterion, the Bayesian information criterion, or log-likelihood. Clustered, crossed, and nested random intercepts and slopes are assembled jointly with diagonal or term-specific unstructured covariance. Available approximation methods include Laplace, saddlepoint likelihood with latent Laplace integration, adaptive Gaussian quadrature, full-covariance Gaussian variational inference, and penalized quasi-likelihood. Penalized smooth mean terms include ordinary and cyclic P-splines, factor-by and varying-coefficient terms, tensor products, shrinkage smooths, and whole-term selection. Term-specific penalties, grouped fold-local smoothing selection, null-space constraints, and smooth effective degrees of freedom remain separate from elastic-net coefficient shrinkage while the smooth mean and serial covariance are fitted jointly. Bootstrap resampling preserves the declared dependence components. The mixed-effects framework is inspired by Laird and Ware (1982) <doi:10.2307/2529876>; generalized-model approximations are inspired by Breslow and Clayton (1993) <doi:10.1080/01621459.1993.10594284>; and serial covariance formulations are inspired by Pinheiro and Bates (2000) <doi:10.1007/b98882>. The run-time fitting interface imports no third-party 'R' packages.

License GPL (>= 3)

Depends R (>= 4.1.0)

Imports stats, utils

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

NeedsCompilation yes

Config/roxygen2/version 8.0.0

VignetteBuilder knitr, rmarkdown

Config/testthat/edition 3

Language en-US

Encoding UTF-8

Author Enoch Kang [aut, cre, trl] (ORCID:
<https://orcid.org/0000-0002-4903-942X>)

Maintainer Enoch Kang <y.enoch.kang@gmail.com>

Repository CRAN

Date/Publication 2026-08-22 15:40:02 UTC

Contents

`check_MEMWAS_assumptions` *Check assumptions for a fitted MEMWAS model*

Description

Runs selected diagnostic screens for temporal autocorrelation, response distribution and link adequacy, residual conditional independence, random-effect normality, random-effect–predictor independence, and variance homogeneity. Diagnostics are separate from ordinary model fitting and can be requested after inspecting the fitted model.

Usage

```
check_MEMWAS_assumptions(
  object,
  autocorrelation_check = "All",
  distribution_link_check = "All",
  conditional_independence_check = "All",
  random_effects_normality_check = "All",
  random_effects_predictor_independence_check = "All",
  homogeneity_variance_check = "All",
  alpha = 0.05,
  ...
)

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

Arguments

`object`	Object of class ‘MEMWAS_fit‘. Must be operationally converged and retain finite residuals, fitted values, response data, primary clusters, times, fitted random-effect contributions, and the fixed-effect design. No default.
`autocorrelation_check`	Character vector. ‘"All"‘, ‘"None"‘, or any of ‘"WithinClusterDifferenceRatioScreen"‘, ‘"PooledWithinClusterPortmanteauScreen"‘, ‘"WithinClusterLag1CorrelationScreen"‘, and ‘"WithinClusterSignRunsScreen"‘. Default is ‘"All"‘.
`distribution_link_check`	Character vector. ‘"All"‘, ‘"None"‘, or any of ‘"PearsonDispersionScreen"‘, ‘"DevianceDispersionScreen"‘, ‘"PearsonResidualJarqueBeraScreen"‘, ‘"SquaredFittedPartialCorrelationScreen"‘, and ‘"BinomialGroupedCalibrationScreen"‘. Default is ‘"All"‘.
`conditional_independence_check`	Character vector. ‘"All"‘, ‘"None"‘, or any of ‘"WithinClusterLag1ResidualScreen"‘, ‘"ClusterMeanResidualHeterogeneityScreen"‘, and ‘"WithinClusterSignRunsScreen"‘. Default is ‘"All"‘.
`random_effects_normality_check`	Character vector. ‘"All"‘, ‘"None"‘, or any of ‘"PooledRandomModeNormalScoreScreen"‘, ‘"PooledRandomModeJarqueBeraScreen"‘, and ‘"PooledRandomModeSkewKurtosisScreen"‘. Default is ‘"All"‘.
`random_effects_predictor_independence_check`	Character vector. ‘"All"‘, ‘"None"‘, or any of ‘"MaxGroupMeanRandomContributionCorrelationScreen"‘, ‘"MaxPredictorRandomContributionPearsonScreen"‘, and ‘"MaxPredictorRandomContributionSpearmanScreen"‘. Default is ‘"All"‘.
`homogeneity_variance_check`	Character vector. ‘"All"‘, ‘"None"‘, or any of ‘"SquaredResidualFittedLinearScreen"‘, ‘"SquaredResidualFittedQuadraticScreen"‘, ‘"MedianSplitAbsoluteResidualScreen"‘, and ‘"GroupAbsoluteResidualHeterogeneityScreen"‘. Default is ‘"All"‘.
`alpha`	Numeric scalar. Significance threshold in ‘(0, 0.5]‘ applied to Holm-adjusted p-values. Default is ‘0.05‘.

...	Additional arguments. No additional arguments are accepted.
x	Object of class 'MEMWAS_assumption_checks'. No default.
digits	Integer scalar. Number of significant digits used for numeric output. Default is 'max(3L, getOption("digits") - 3L)'.

Details

Holm adjustment is applied jointly to every requested diagnostic with an available finite p-value. Structurally unavailable diagnostics remain in the output with 'available = FALSE'; an unavailable row is not evidence that its assumption holds. Method names describe the calculations actually performed and should not be interpreted as exact implementations of similarly named textbook tests unless the result detail explicitly says so.

Value

Object of class 'MEMWAS_assumption_checks'. It contains the normalized selector specification, requested-method table, full results, flagged rows, unavailable rows with explanations, temporal-autocorrelation subset, alpha, multiplicity method, settings, and call.

The input object, invisibly.

Examples

```
set.seed(1L)
sim_data <- MEMWAS:::.simulate_panel_data(
  n_id = 6L, n_time = 3L, beta = c(x1 = 0.6),
  cor_matrix = matrix(1, 1L, 1L), intercept = 1,
  sigma_eps = 0.2, sigma_b = 0, autocor = "NONE"
)
fit <- fit_MEMWAS(
  y ~ x1, data = sim_data, id = "id", time = "time",
  random = NULL, autocor = "NONE", se_method = "none",
  control = list(n_starts = 1L, cold_start_verification = FALSE),
  verbose = FALSE
)
check_MEMWAS_assumptions(
  fit,
  autocorrelation_check = "WithinClusterLag1CorrelationScreen",
  distribution_link_check = "None",
  conditional_independence_check = "None",
  random_effects_normality_check = "None",
  random_effects_predictor_independence_check = "None",
  homogeneity_variance_check = "None"
)
```

`coef.MEMWAS_fit` *Methods for fitted MEMWAS models*

Description

Extracts coefficient versions, covariance matrices, likelihood/objective values, information criteria, fitted values, residuals, observation counts, summaries, and printed representations from fitted MEMWAS models.

Extracts selected coefficients or penalty parameters, creates a structured tuning summary, and prints tuning or summary objects.

Usage

```
## S3 method for class 'MEMWAS_fit'
coef(object, type = c("adjusted", "original", "both"), ...)

## S3 method for class 'MEMWAS_family_fit'
coef(object, ...)

## S3 method for class 'MEMWAS_fit'
vcov(object, full = FALSE, ...)

## S3 method for class 'MEMWAS_family_fit'
vcov(object, full = FALSE, ...)

## S3 method for class 'MEMWAS_fit'
logLik(object, ...)

## S3 method for class 'MEMWAS_family_fit'
logLik(object, ...)

## S3 method for class 'MEMWAS_fit'
AIC(object, ..., k = 2)

## S3 method for class 'MEMWAS_family_fit'
AIC(object, ..., k = 2)

## S3 method for class 'MEMWAS_fit'
BIC(object, ...)

## S3 method for class 'MEMWAS_family_fit'
BIC(object, ...)

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

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

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

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

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

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

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

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

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

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

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

## S3 method for class 'tune_MEMWAS'
predict(object, ...)

## S3 method for class 'tune_MEMWAS'
coef(object, ...)

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

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

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

Arguments

object	Object of class ‘tune_MEMWAS‘. No default.

type	Character scalar. For ‘coef()‘, options are ‘"adjusted"‘ for the final regularized coefficients, ‘"original"‘ for coefficients from the same selected model refitted without elastic-net L1/L2 penalties (while retaining any smooth roughness penalties), and ‘"both"‘ for the complete comparison table. For ‘residuals()‘, options are ‘"response"‘ and ‘"pearson"‘. The method-specific default is the first listed option.
...	Additional arguments. ‘coef()‘ forwards them to the final fitted model when one exists; other methods reject unused arguments.
full	Logical scalar. Whether ‘vcov()‘ returns the full parameter covariance matrix instead of the fixed-effect block. Default is ‘FALSE‘.
k	Numeric scalar. Positive multiplier used by ‘AIC()‘; ‘2‘ gives the ordinary Akaike criterion. Default is ‘2‘.
x	Object of class ‘tune_MEMWAS‘ or ‘summary.tune_MEMWAS‘. No default.
digits	Integer scalar. Number of significant digits used for numeric output. Default is ‘max(3L, getOption("digits") - 3L)‘.

Details

‘coef(type = "adjusted")‘ returns the coefficients used for fitted values and predictions. ‘coef(type = "original")‘ returns the elastic-net-unpenalized reference coefficients for the same selected formula, smooth penalties, and dependence specification. ‘coef(type = "both")‘ reports both vectors, their signed adjustment, and a regularization indicator. For approximation-specific objectives that are not marginal likelihoods, ‘logLik()‘ returns a labelled objective value and ‘AIC()‘ and ‘BIC()‘ return ‘NA‘ with a warning rather than treating unlike objectives as likelihoods. Penalized AIC and BIC are likewise unavailable unless the fit has a valid generalized-trace effective degrees of freedom; an active-count fallback is retained only as a diagnostic. BIC additionally requires at least two independent connected dependence components. Printed fit summaries separate parametric fixed-effect inference from one smooth-term table containing term-specific smoothing parameters, penalty ranks, null-space dimensions, effective degrees of freedom, and inferential status when available. Internal smooth-basis coordinates remain accessible through ‘coef()‘ but are not repeated as parametric Wald rows or expanded in the displayed model equation. When a coefficient/smooth/serial penalty or a whole-term smooth-selection step suppresses ordinary Wald inference, ‘vcov()‘ also fails explicitly instead of exposing an invalid Hessian covariance; dependence-preserving bootstrap inference is required.

When a final refit exists, ‘coef()‘ supports the coefficient ‘type‘ choices documented for ‘coef.MEMWAS_fit()‘. Without a final refit it returns the selected penalty parameters. The summary includes grouped-fold performance, stability, convergence, selected elastic-net penalties, separately reported smoothing parameters and effective degrees of freedom, one concise table of fold-local smoothing selections, and distinct reference and final equations when available. Printed output does not repeat internal smooth-basis coordinates as parametric coefficients.

Value

The requested numeric vector, matrix, scalar, summary object, or the input object invisibly for print methods. ‘coef(type = "both")‘ returns a data frame containing original coefficients, adjusted coefficients, and regularization adjustments.

'coef()' returns a coefficient vector, coefficient-version table, or selected penalty vector. 'summary()' returns an object of class 'summary.tune_MEMWAS'. Print methods return their input invisibly.

Examples

```
set.seed(1L)
sim_data <- MEMWAS:::.simulate_panel_data(
  n_id = 6L, n_time = 3L, beta = c(x1 = 0.6),
  cor_matrix = matrix(1, 1L, 1L), intercept = 1,
  sigma_eps = 0.2, sigma_b = 0, autocor = "NONE"
)
fit <- fit_MEMWAS(
  y ~ x1, data = sim_data, id = "id", time = "time",
  random = NULL, autocor = "NONE", se_method = "none",
  control = list(n_starts = 1L, cold_start_verification = FALSE),
  verbose = FALSE
)
coef(fit)
fitted(fit)
residuals(fit)
set.seed(1L)
sim_data <- MEMWAS:::.simulate_panel_data(
  n_id = 6L, n_time = 3L, beta = c(x1 = 0.6),
  cor_matrix = matrix(1, 1L, 1L), intercept = 1,
  sigma_eps = 0.2, sigma_b = 0, autocor = "NONE"
)
tuned <- suppressWarnings(MEMWAS:::.tune_MEMWAS(
  y ~ x1, data = sim_data, id = "id", time = "time",
  random = NULL, autocor = "NONE", K = 2L,
  n_initial = 5L, max_iter = 0L, use_stability_penalty = FALSE,
  se_method = "none", refit_final = FALSE,
  control = list(n_starts = 1L, cold_start_verification = FALSE),
  verbose = FALSE
))
coef(tuned)
```

`compare_approximations` *Compare MEMWAS approximation requests*

Description

Fits the same model under each requested approximation and reports the dispatched method, computational kernel, objective semantics, information criteria, and convergence state.

Usage

```
compare_approximations(approximations = c("laplace"), ...)
```

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

Arguments

approximations	Character vector. One or more of ‘"auto"‘, ‘"laplace"‘, ‘"saddlepoint"‘, ‘"adaptive_gaussian_quadrature"‘, ‘"variational_inference"‘, and ‘"pql"‘. Default is ‘c("laplace")‘.
...	Named arguments passed to ‘fit_MEMWAS()‘ by ‘compare_approximations()‘. At minimum these normally include ‘formula‘, ‘data‘, ‘id‘, and ‘time‘; the wrapper introduces no separate defaults. For the print method, additional arguments are passed to ‘print.data.frame()‘.
x	Object of class ‘MEMWAS_approximation_comparison‘. Comparison object to print. No default.

Value

Object of class ‘MEMWAS_approximation_comparison‘ containing a summary table and the fitted model for every requested approximation.

Examples

```
set.seed(1L)
sim_data <- MEMWAS:::.simulate_panel_data(
  n_id = 6L, n_time = 3L, beta = c(x1 = 0.6),
  cor_matrix = matrix(1, 1L, 1L), intercept = 1,
  sigma_eps = 0.2, sigma_b = 0, autocor = "NONE"
)
one_method <- compare_approximations(
  "laplace", formula = y ~ x1, data = sim_data,
  id = "id", time = "time",
  random = NULL, autocor = "NONE", se_method = "none",
  control = list(n_starts = 1L, cold_start_verification = FALSE)
)
one_method$table
```

---

diagnose_approximation

Diagnose a MEMWAS approximation

Description

Summarizes objective semantics, optimizer stationarity, boundary conditions, design rank, covariance-Jacobian rank, and approximation-specific convergence.

Usage

```
diagnose_approximation(object)

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

Arguments

object	Object of class 'MEMWAS_fit'. Fitted model to diagnose. No default.
x	Object of class 'MEMWAS_approximation_diagnostics'. Diagnostic object to print. No default.
...	Additional arguments. No additional arguments are accepted by the diagnostic print method.

Value

Object of class 'MEMWAS_approximation_diagnostics' containing objective, optimizer, gradient, step, KKT, boundary, stationarity, rank, and approximation metadata.

Examples

```
set.seed(1L)
sim_data <- MEMWAS:::.simulate_panel_data(
  n_id = 6L, n_time = 3L, beta = c(x1 = 0.6),
  cor_matrix = matrix(1, 1L, 1L), intercept = 1,
  sigma_eps = 0.2, sigma_b = 0, autocor = "NONE"
)
fit <- fit_MEMWAS(
  y ~ x1, data = sim_data, id = "id", time = "time",
  random = NULL, autocor = "NONE", se_method = "none",
  control = list(n_starts = 1L, cold_start_verification = FALSE),
  verbose = FALSE
)
diagnose_approximation(fit)
```

---

fit_MEMWAS *Fit a MEMWAS mixed-effects model*

Description

Fits longitudinal generalized mixed-effects models with clustered, crossed, or nested random effects and unified or predictor-specific serial processes. Nonlinearity screening and assumption diagnostics are separate optional stages and are disabled by default so that ordinary fitting avoids their additional computation.

Usage

```
fit_MEMWAS(
  formula,
  family = "gaussian",
  data,
  id,
  time,
  random = ~1,
  autocor = "AR(1)",
  serial = NULL,
  residual_autocor = NULL,
  predictor_autocor = NULL,
  L1_penalty = 0,
  L2_penalty = 0,
  control = list(),
  method = "ML",
  random_cov = "unstructured",
  approximation = "laplace",
  init_approximation = "native_fixed_effect",
  quadrature_points = 7L,
  se_method = "hessian",
  link = NULL,
  offset = NULL,
  weights = NULL,
  subset = NULL,
  na.action = NULL,
  theta = NULL,
  shape = NULL,
  start = NULL,
  dot_predictors = NULL,
  dot_alternative = NULL,
  dot_threshold = 0,
  dot_alpha = 0.05,
  nonlinear_alpha = 0.05,
  spline_knots = 4L,
  spline_probs = NULL,
  nonlinear_predictors = NULL,
  screen_nonlinear = FALSE,
  nonlinear_multiplicity = "holm",
  nonlinear_screening_method = "nuisance_adjusted_score",
  nonlinear_bootstrap_reps = 100L,
  nonlinear_bootstrap_seed = NULL,
  check_assumptions = FALSE,
  run_checks_and_screening = NULL,
  bootstrap_inference = FALSE,
  bootstrap_reps = 100L,
  bootstrap_conf_level = 0.95,
  bootstrap_seed = NULL,
```

```
  bootstrap_min_success_fraction = 0.8,
  post_selection_inference = "none",
  prediction_inference = FALSE,
  prediction_level = 0.95,
  prediction_interval = "confidence",
  autocorrelation_check = "All",
  distribution_link_check = "All",
  conditional_independence_check = "All",
  random_effects_normality_check = "All",
  random_effects_predictor_independence_check = "All",
  homogeneity_variance_check = "All",
  verbose = TRUE,
  smooth = NULL,
  smooth_control = list(),
  nonlinear_spline = "restricted_cubic",
  ...
)
```

Arguments

formula	Formula object. A two-sided model formula defining the response and fixed effects. No default.
family	Character scalar or family object. Supported families are ‘"gaussian"‘, ‘"binomial"‘, ‘"poisson"‘, ‘"negative_binomial"‘, ‘"gamma"‘, and ‘"exponential"‘. Default is ‘"gaussian"‘.
data	Data frame. Contains every variable used by the model, grouping, time, offset, weight, and subset expressions. No default.
id	Character scalar. Name of the primary independent-cluster column in ‘data‘. No default.
time	Character scalar. Name of the measurement-time column in ‘data‘. Values must be orderable within primary clusters. No default.
random	One-sided formula, character specification, structured term, list of terms, or ‘NULL‘. Defines random intercepts and slopes; crossed and nested terms are allowed. Default is ‘~ 1‘, interpreted at the primary ID.
autocor	Character scalar, named character vector, or ‘NULL‘. A scalar defines one outcome-loaded serial process; a named vector defines predictor-specific processes. Options include ‘"NONE"‘, ‘"AR(1)"‘, ‘"OU"‘, ‘"expOU"‘, ‘"AR(p)"‘, ‘"ARMA(1,1)"‘, ‘"CS"‘, ‘"TOEP"‘, and ‘"UN"‘. Default is ‘"AR(1)"‘; use ‘NULL‘ when supplying ‘serial‘.
serial	Object of class ‘MEMWAS_serial_component‘, structured serial specification, list of components, or ‘NULL‘. Provides explicit unified or predictor-specific serial components. Default is ‘NULL‘.
residual_autocor	Character scalar, object of class ‘MEMWAS_serial_component‘, structured component, or ‘NULL‘. Explicit outcome-loaded residual autocorrelation structure.

When this argument or ‘predictor_autocor‘ is supplied, do not also supply a non-NULL ‘autocor‘ or ‘serial‘. ‘NULL‘ adds no outcome-loaded serial component. Default is ‘NULL‘.

`predictor_autocor`
: Named character vector, named list of structure specifications, or ‘NULL‘. Each name identifies a numeric predictor whose values load an independently parameterized residual serial covariance component. Entries may use the structures accepted by ‘serial_component()‘; ‘"NONE"‘ omits that predictor component. These components are conditional outcome-covariance terms, not joint stochastic models for the predictors. Default is ‘NULL‘.

`L1_penalty`
: Numeric scalar. Non-negative L1 penalty applied to non-intercept fixed effects. Default is ‘0‘.

`L2_penalty`
: Numeric scalar. Non-negative L2 penalty applied to non-intercept fixed effects. Default is ‘0‘.

`control`
: Named list. Computational and model controls. Common entries include ‘maxit‘, ‘evalmax‘, ‘reltol‘, ‘mode_maxit‘, ‘mode_tol‘, ‘optimizer‘, ‘n_starts‘, ‘parallel_starts‘, ‘optimizer_threads‘, ‘compute_hessian‘, ‘hessian_strategy‘, ‘quadrature_max_dimension‘, ‘quadrature_max_nodes‘, ‘vi_maxit‘, ‘pql_maxit‘, ‘autocor_regularization‘, ‘serial_time_scale‘, and ‘allow_negative_autocor‘. With native OpenMP support, ‘parallel_starts = TRUE‘ evaluates distinct optimizer starts concurrently using at most ‘optimizer_threads‘ workers; otherwise the same starts are evaluated serially. Set ‘dense_fallback = FALSE‘ to forbid compatibility fallback that materializes the global random-effects design; the native pivoted rank diagnostic still operates directly on compressed row pointers, column indices, and nonzero values and retains pivot and alias reporting. ‘block_factorization = FALSE‘ explicitly selects the dense reference path, so it is incompatible with that prevention request. Default is ‘list()‘.

`method`
: Character scalar. Estimation option ‘"ML"‘ or, for an unpenalized Gaussian identity-link model, ‘"REML"‘. Default is ‘"ML"‘.

`random_cov`
: Character scalar or term-wise character specification. Options are ‘"diagonal"‘ and ‘"unstructured"‘. Default is ‘"unstructured"‘.

`approximation`
: Character scalar. Final approximation option: ‘"auto"‘, ‘"native_fixed_effect"‘, ‘"laplace"‘, ‘"saddlepoint"‘, ‘"adaptive_gaussian_quadrature"‘, ‘"variational_inference"‘, or ‘"pql"‘. Default is ‘"laplace"‘.

`init_approximation`
: Character scalar. Initialization option accepted by the approximation dispatcher. Default is ‘"native_fixed_effect"‘ for using least-squares approach.

`quadrature_points`
: Integer scalar. Positive Gauss–Hermite order used by adaptive quadrature and predictive-density integration. Default is ‘7L‘.

`se_method`
: Character scalar. Options are ‘"hessian"‘ and ‘"none"‘. Default is ‘"hessian"‘.

`link`
: Character scalar or ‘NULL‘. Optional supported link override for ‘family‘. ‘NULL‘ uses the family default. Default is ‘NULL‘.

`offset`
: Numeric vector, model-frame expression, or ‘NULL‘. Additive fixed offset on the linear-predictor scale. Default is ‘NULL‘.

`weights` Numeric vector, model-frame expression, or ‘NULL‘. Non-negative observation weights; for binomial proportions, positive integer values are interpreted as trial totals. Default is ‘NULL‘.

`subset` Logical/integer vector, model-frame expression, or ‘NULL‘. Selects analysis rows before missing-data handling. Default is ‘NULL‘.

`na.action` Function, character name, or ‘NULL‘. Missing-data action used by model-frame preparation. ‘NULL‘ uses the package default. Default is ‘NULL‘.

`theta` Numeric scalar or ‘NULL‘. Positive negative-binomial dispersion parameter; ‘NULL‘ estimates it when applicable. Default is ‘NULL‘.

`shape` Numeric scalar or ‘NULL‘. Positive gamma shape parameter; ‘NULL‘ estimates it when applicable. Default is ‘NULL‘.

`start` Named numeric vector, complete numeric vector, or ‘NULL‘. Optional parameter starting values. Default is ‘NULL‘.

`dot_predictors` Character vector, named specification, data frame, or ‘NULL‘. Selects fixed-effect terms for directional tests. ‘NULL‘ requests no directional tests. Default is ‘NULL‘.

`dot_alternative`
Character scalar/vector, named specification, data frame, or ‘NULL‘. Options are ‘"greater"‘, ‘"less"‘, and ‘"two.sided"‘. Default is ‘NULL‘, which uses two-sided testing for selected terms.

`dot_threshold` Numeric scalar/vector or named specification. Defines the non-negative minimally important coefficient magnitude for directional tests. Default is ‘0‘.

`dot_alpha` Numeric scalar/vector or named specification. Significance level in ‘(0, 1)‘ for directional tests. Default is ‘0.05‘.

`nonlinear_alpha`
Numeric scalar. Selection threshold in ‘(0, 1)‘ for nonlinear screening. Default is ‘0.05‘.

`spline_knots` Integer scalar. Number of restricted-cubic-spline knots or the initial P-spline basis dimension for nonlinear screening; at least three are required. Default is ‘4L‘.

`spline_probs` Numeric vector or ‘NULL‘. Strictly increasing knot probabilities in ‘[0, 1]‘; the length must equal ‘spline_knots‘. ‘NULL‘ uses the package probabilities. This argument is restricted to the restricted-cubic engine. Default is ‘NULL‘.

`nonlinear_predictors`
Character vector or ‘NULL‘. Numeric fixed-effect predictors eligible for screening. ‘NULL‘ uses all eligible predictors. Default is ‘NULL‘.

`screen_nonlinear`
Logical scalar. Whether to call ‘screen_MEMWAS_nonlinearity()‘ before the final fit. Default is ‘FALSE‘.

`nonlinear_multiplicity`
Character scalar. P-value adjustment used by likelihood-ratio screening. Options are ‘"holm"‘, ‘"hochberg"‘, ‘"hommel"‘, ‘"bonferroni"‘, ‘"BH"‘, ‘"BY"‘, and ‘"none"‘. Default is ‘"holm"‘.

`nonlinear_screening_method`
Character scalar. Options are ‘"nuisance_adjusted_score"‘ and ‘"likelihood_ratio"‘. Default is ‘"nuisance_adjusted_score"‘.

`nonlinear_bootstrap_reps`
: Integer scalar. Number of primary-cluster multiplier-bootstrap replicates for score screening; at least 99 are required. Default is ‘100L‘.

`nonlinear_bootstrap_seed`
: Integer scalar or ‘NULL‘. Non-negative seed for a package-local native score-screening stream; it does not alter the global random-number state. ‘NULL‘ uses and advances the current ’R’ stream. Default is ‘NULL‘.

`check_assumptions`
: Logical scalar. Whether to call ‘check_MEMWAS_assumptions()‘ after fitting. Default is ‘FALSE‘.

`run_checks_and_screening`
: Logical scalar or ‘NULL‘. ‘FALSE‘ disables both optional stages; ‘TRUE‘ activates either stage whose individual flag was omitted; ‘NULL‘ respects ‘screen_nonlinear‘ and ‘check_assumptions‘. Default is ‘NULL‘.

`bootstrap_inference`
: Logical scalar. Whether to run the automatic dependence-aware bootstrap after fitting. Primary-ID cluster case resampling is used only when every random-effect grouping partition equals the primary-ID partition. Its sampling frame contains only positive-weight rows and IDs having at least one such row; zero-weight rows are ignored. Models with nested or crossed grouping levels instead use joint parametric simulation of random effects, serial processes, and responses on the fitted incidence graph. That joint parametric route requires every positive observation/frequency weight to equal one; zero- weight rows are ignored and a positive non-unit likelihood weight fails explicitly because it does not define a response-generating mechanism. Default is ‘FALSE‘.

`bootstrap_reps`
: Integer scalar. Number of dependence-preserving bootstrap replicates; at least two are required when bootstrap inference is active. Default is ‘100L‘.

`bootstrap_conf_level`
: Numeric scalar. Bootstrap confidence level in ‘(0, 1)‘. Default is ‘0.95‘.

`bootstrap_seed`
: Integer scalar or ‘NULL‘. Non-negative seed for the package-local bootstrap stream used by the automatically selected case or joint-parametric scheme; it does not alter the global random-number state. ‘NULL‘ uses and advances the current ’R’ stream. Default is ‘NULL‘.

`bootstrap_min_success_fraction`
: Numeric scalar. Minimum acceptable fraction of successful bootstrap refits in ‘(0, 1]‘. Default is ‘0.80‘.

`post_selection_inference`
: Character scalar. Compatibility selector with options ‘"none"‘ and ‘"bootstrap_after_selection"‘. The latter turns on the same dependence-preserving bootstrap requested by ‘bootstrap_inference = TRUE‘; it is not a different refitting algorithm. Every bootstrap of a penalized fit repeats the fitted penalty-induced active-set operation. When retained, the preselection smooth specification also repeats automatic smoothing and grouped-CV whole-term selection; nonlinear-screen candidate selection remains conditional on the screened formula. Coordinate-wise percentile intervals are reported only for invariant parametric coefficients. If primary-ID case resampling rebuilds a global smooth projection, those nominal coefficients are resample-local, or if whole-term replay can add/remove such a projec-

tion, the coefficient table is explicitly unavailable and only refit/selection diagnostics are retained; common-scale function or prediction contrasts are required for numerical intervals. Bootstrap output is not reported as universal selection-corrected p-values. Default is ‘"none"‘.

`prediction_inference`
: Logical scalar. Whether to calculate fitted-row Hessian/delta-method uncertainty after an unpenalized fit. This inference is unavailable when a fixed-effect, smooth, or serial-covariance penalty or whole-term smooth-selection step is active; use a dependence-preserving bootstrap workflow for penalized or post-selection uncertainty. Default is ‘FALSE‘.

`prediction_level`
: Numeric scalar. Prediction confidence level in ‘(0, 1)‘. Default is ‘0.95‘.

`prediction_interval`
: Character scalar. Options are ‘"confidence"‘ and ‘"prediction"‘. Default is ‘"confidence"‘.

`autocorrelation_check`
: Character vector. ‘"All"‘, ‘"None"‘, or method names accepted by ‘check_MEMWAS_assumptions()‘. Default is ‘"All"‘; active only when ‘check_assumptions = TRUE‘.

`distribution_link_check`
: Character vector. ‘"All"‘, ‘"None"‘, or distribution/link method names accepted by ‘check_MEMWAS_assumptions()‘. Default is ‘"All"‘; active only when ‘check_assumptions = TRUE‘.

`conditional_independence_check`
: Character vector. ‘"All"‘, ‘"None"‘, or conditional-independence method names accepted by ‘check_MEMWAS_assumptions()‘. Default is ‘"All"‘; active only when ‘check_assumptions = TRUE‘.

`random_effects_normality_check`
: Character vector. ‘"All"‘, ‘"None"‘, or random-effect normality method names accepted by ‘check_MEMWAS_assumptions()‘. Default is ‘"All"‘; active only when ‘check_assumptions = TRUE‘.

`random_effects_predictor_independence_check`
: Character vector. ‘"All"‘, ‘"None"‘, or random-effect–predictor method names accepted by ‘check_MEMWAS_assumptions()‘. Default is ‘"All"‘; active only when ‘check_assumptions = TRUE‘.

`homogeneity_variance_check`
: Character vector. ‘"All"‘, ‘"None"‘, or variance-homogeneity method names accepted by ‘check_MEMWAS_assumptions()‘. Default is ‘"All"‘; active only when ‘check_assumptions = TRUE‘.

`verbose`
: Logical scalar. Whether to report progress. Default is ‘TRUE‘.

`smooth`
: ‘NULL‘, one named smooth-term list, or a list of named smooth-term lists. Each term names ‘variable‘ (or ‘variables‘ for a tensor product) and may set ‘name‘/‘label‘, ‘type‘, ‘by‘, ‘k‘, ‘degree‘, ‘difference_order‘, ‘lambda‘, ‘lambda_initial‘, ‘period‘, ‘boundary_knots‘, ‘rank_tolerance‘, ‘select‘, ‘selection‘, ‘whole_term_selection‘, ‘selection_penalty‘, and ‘extrapolation‘. Term-level ‘lambda_initial‘ and ‘rank_tolerance‘ override the corresponding ‘smooth_control‘ defaults. Available types are ordinary and cyclic P-splines, factor-by and treatment-specific smooths, numeric-by

varying-coefficient smooths, tensor products, and shrinkage smooths. ‘lambda = NULL‘ or ‘"auto"‘ requests grouped-CV smoothing; a finite non-negative value fixes the term-specific smoothing parameter (one per tensor margin). Subject-specific random smooths are deliberately unavailable pending a redesign of sparse latent integration. Default is ‘NULL‘.

`smooth_control` Named list controlling P-spline construction and smoothing selection. Basis defaults are ‘k = 10L‘, ‘degree = 3L‘, ‘difference_order = 2L‘, ‘lambda_initial = 1‘, ‘selection_penalty = 1e-4‘, ‘extrapolation = "constant"‘, and ‘rank_tolerance = sqrt(.Machine$double.eps)‘. Automatic smoothing uses dependence-component grouped cross-validation and accepts ‘optimizer = "grouped_cv"‘, ‘cv_folds = 5L‘, ‘log_lambda_range = c(-4, 4)‘, ‘lambda_base = 10‘, ‘grid_size = 5L‘, ‘max_iter = 2L‘, ‘tolerance = 1e-4‘, ‘metric‘ equal to ‘"RMSE"‘, ‘"MAE"‘, or ‘"MSE"‘, ‘seed‘, and ‘selection_tolerance‘. Default is ‘list()‘.

`nonlinear_spline`
Character scalar. Nonlinearity-screening engine: ‘"restricted_cubic"‘ or ‘"pspline"‘. The latter builds a native univariate P-spline for each candidate, keeps its smoothness separate from elastic-net shrinkage, and reports smooth-specific effective rather than raw-basis degrees of freedom. Screening remains inactive unless ‘screen_nonlinear = TRUE‘; ‘run_checks_and_screening = FALSE‘ also prevents it from running. Default is ‘"restricted_cubic"‘.

`...` Additional arguments. No additional arguments are accepted.

Details

A smooth model extends the fixed mean as

$$g\{E(Y_{it} \mid b_i, u_i)\} = X_{it}\beta + \sum_j B_j(x_{it})\gamma_j + Z_{it}b_i + S_{it}u_i + o_{it}.$$

Smooth coefficients are kept in the profiled fixed-coefficient block, so the smooth mean, random effects, and serial covariance parameters are fitted jointly rather than in a two-stage residual smoother. The implementation is family-agnostic: the same augmented fixed design is used for every supported response family and approximation, subject to their usual numerical and inferential restrictions.

Each smooth has its own penalty matrix and smoothing parameter,

$$\frac{1}{2}\sum_j \lambda_{s,j}\gamma_j^T S_j \gamma_j,$$

with the default univariate P-spline using $S_j = D_{d_j}^T D_{d_j}$ for a coefficient-difference matrix of order ‘difference_order‘. This quadratic roughness penalty is not the elastic-net L1/L2 penalty: smooth coordinates are excluded from L1 shrinkage and their diagonalized penalty weights are stored and optimized separately. MEMWAS eigendecomposes each term penalty, explicitly partitions its penalized range from its null space, and applies term-appropriate alias and identifiability constraints before keeping the resulting coordinates in the profiled fixed block. Ordinary, cyclic, and tensor terms are centered/projected against the parametric design. For factor-by and varying-coefficient terms, that residualization is used only to diagnose exact aliases: the fitted designs remain $B(x)I(\text{level})$ and $zB(x)$, respectively, and cannot acquire unrequested interactions with

arbitrary parametric covariates. ‘select = TRUE‘ (or ‘type = "shrinkage"‘) adds a small null-space penalty; alternatively, ‘selection = "whole_term"‘ compares complete-term removal by grouped CV. The two selection mechanisms are mutually exclusive.

Ordinary, cyclic, factor-by/treatment-specific, varying-coefficient, tensor-product, shrinkage, and whole-term-selection smooths use this common representation. A factor ‘by‘ creates level-specific smooths; a numeric ‘by‘ creates a varying coefficient. Tensor products accept one basis dimension, degree, difference order, boundary pair, extrapolation rule, and smoothing parameter per margin. Cyclic P-splines impose periodic construction and prediction; noncyclic prediction can reject, clamp, or linearly extrapolate beyond the training boundary. Subject-specific random smooths are rejected explicitly until sparse latent integration is redesigned.

With ‘lambda = NULL‘/‘"auto"‘, preprocessing, boundaries, basis construction, null-space constraints, and any nonlinear screening are rebuilt from each training partition using only positive-frequency-weight rows. The fold dependence graph is also constructed only from those rows, so a zero-weight row cannot bridge otherwise independent components or create a fold. Folds contain complete connected components induced by the primary ID and every nested or crossed grouping factor. Thus validation rows cannot determine a training basis or share a latent grouping level with it. Smoothing candidates re-estimate the smooth mean and serial covariance jointly in every fold. Fixed lambdas bypass this internal search. Reported smooth inference uses term-specific effective degrees of freedom, reference degrees of freedom, statistics, p-values, and inferential status, rather than treating raw basis dimension as the test degrees of freedom.

The fitted fixed-effect vector is the adjusted coefficient vector used by the final penalized objective. When a fixed-effect penalty is positive, the same selected model is also refitted without elastic-net L1/L2 penalties to obtain the original coefficient vector; any smooth roughness penalty remains. The fit stores both vectors, their differences, and two equations:

$$g\{E(Y_{it} \mid b_i, u_i)\} = X_{it}\beta^{(0)} + \sum_j f_j(x_{it}) + Z_{it}b_i + S_{it}u_i + o_{it}$$

for the Original Model equation and

$$g\{E(Y_{it} \mid b_i, u_i)\} = X_{it}\tilde{\beta} + \sum_j f_j(x_{it}) + Z_{it}b_i + S_{it}u_i + o_{it}$$

for the Final equation, where ‘beta^(0)‘ is the L1/L2-unpenalized reference and ‘tilde(beta)‘ is the adjusted vector from the selected penalized fit. Smooth terms are rendered by their labels rather than by internal basis coordinates.

Score-based nonlinearity screening uses one unpenalized ML–Laplace null fit, nuisance residualization, and a shared primary-cluster maxT multiplier bootstrap. Likelihood-ratio screening rebuilds and refits each candidate design. Both methods can screen restricted cubic splines or native univariate P-splines. Assumption diagnostics are evaluated only when requested. Neither P-spline screening nor any other nonlinear screen is automatically run when ‘screen_nonlinear = FALSE‘ or ‘run_checks_and_screening = FALSE‘; an explicit ‘smooth‘ specification is a model term, not a request to screen unrelated predictors. When explicit smooths and screening are both requested, the screening reference excludes those penalized terms to preserve an unpenalized parametric null; requested and screen-selected smooths are then combined, and jointly fitted in the final model.

Information criteria require a finite marginal log-likelihood. For a penalized fit, AIC and BIC are reported only when effective degrees of freedom are available from the generalized-trace calculation; an active-parameter fallback remains diagnostic and is not used as an information-criterion dimension. BIC uses the number of independent connected components among positive-weight rows in

the joint graph induced by the primary ID and every nested or crossed random-effect grouping factor. At least two such components are required, and frequency weights do not create additional BIC units. Fixed parameter coordinates are excluded from information-criterion degrees of freedom. For an unpenalized Hessian covariance, only the free-coordinate block is inverted and fixed-coordinate rows and columns are exactly zero.

Inferential bootstrap refits preserve the fitted dependence specification. Ordinary primary-ID case resampling is valid only for primary-ID-only random grouping. Its dependence analysis and native cluster draws use only positive-weight rows, so zero-weight-only IDs are ineligible and zero-weight rows within otherwise eligible IDs are ignored. Nested or crossed random-effect models use a joint parametric bootstrap on the original incidence graph so shared grouping levels and longitudinal serial dependence are not broken by one-way resampling. Joint parametric simulation is available only when every positive observation/frequency weight is one; zero-weight rows are ignored. MEMWAS does not reinterpret positive non-unit likelihood weights as replicate counts or known inverse response variances. When any fixed-effect, smooth, or serial-covariance penalty is active, the penalized Hessian is retained only as optimization curvature and is not reported as a sampling covariance. Ordinary Wald standard errors, tests, confidence intervals, and Hessian/delta prediction inference are therefore unavailable. The dependence-preserving bootstrap repeats the complete fitted penalized estimation operation. It reports empirical standard errors and percentile intervals only for coefficient coordinates that retain a common parameterization across refits, and deliberately does not turn sign counts into post-selection p-values. In particular, primary-ID case resampling rebuilds a global smooth projection from each resample; its nominal parametric coefficients are therefore not compared coordinate-wise. The same suppression applies when whole-term replay can add or remove a global projection. The fit records successful refits and selection replay, marks the coefficient table unavailable, and directs numerical inference to common-scale function or prediction contrasts.

Value

Object of class 'MEMWAS_fit'. It contains adjusted and original fixed-effect coefficients, coefficient adjustments, Original Model and Final equations, covariance parameters, random and serial modes, normalized smooth specifications, term-specific penalty and smoothing metadata, smooth effective degrees of freedom and inference summaries, approximation metadata, diagnostics, and convergence details.

Examples

```
set.seed(1L)
sim_data <- MEMWAS:::.simulate_panel_data(
  n_id = 6L, n_time = 3L, beta = c(x1 = 0.6),
  cor_matrix = matrix(1, 1L, 1L), intercept = 1,
  sigma_eps = 0.2, sigma_b = 0, autocor = "NONE"
)
fit <- fit_MEMWAS(
  y ~ x1, data = sim_data, id = "id", time = "time",
  random = NULL, autocor = "NONE", se_method = "none",
  control = list(n_starts = 1L, cold_start_verification = FALSE),
  verbose = FALSE
)
coef(fit)
```

MEMWAS_capabilities *Report MEMWAS fitting capabilities*

Description

Returns the model and approximation contract used by ‘fit_MEMWAS()‘, ‘tune_MEMWAS()‘, and ‘rank_autocorrelation_structures()‘.

Usage

```
MEMWAS_capabilities()
```

Details

Grouped tuning and covariance ranking assign connected dependence components intact to folds. Bootstrap inference selects primary-ID case resampling only when it preserves every fitted grouping partition. Case resampling draws only positive-weight rows from IDs having at least one positive-weight row; zero-weight rows and zero-weight-only IDs are ignored; otherwise it uses joint parametric simulation on the nested or crossed incidence graph. That joint parametric route requires every positive observation/frequency weight to equal one; zero-weight rows are ignored.

Value

Data frame with one row per approximation and columns describing supported families and links, random effects, covariance structures, serial processes, coefficient and smooth penalties, supported smooth classes, smoothness selection, estimation, initialization, standard-error methods, prediction modes, computational kernels, and unavailable combinations. The ‘registry_version‘ attribute records the contract version. Subject-specific random smooths are deliberately outside this contract pending redesign of sparse latent integration.

Examples

```
capabilities <- MEMWAS_capabilities()
capabilities[, c("approximation", "active", "kernels")]
```

predict.MEMWAS_fit *Predict from a fitted MEMWAS model*

Description

Computes conditional, zero-latent, population-marginal, or new-cluster predictions from a fitted MEMWAS model or a tuned object with a final refit.

Usage

```
## S3 method for class 'MEMWAS_fit'
predict(
  object,
  newdata = NULL,
  type = c("link", "response"),
  mode = NULL,
  include_random = is.null(newdata),
  include_serial = is.null(newdata),
  allow_new_levels = TRUE,
  offset = NULL,
  interval = NULL,
  level = NULL,
  se.fit = FALSE,
  trials = NULL,
  joint = FALSE,
  ...
)

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

Arguments

object	Object of class ‘MEMWAS_fit‘, ‘MEMWAS_family_fit‘, or ‘tune_MEMWAS‘. A tuned object must contain a final refit. No default.
newdata	Data frame or ‘NULL‘. Prediction rows; ‘NULL‘ uses retained fitted rows. Default is ‘NULL‘.
type	Character scalar. Options are ‘"link"‘ and ‘"response"‘. Population-marginal modes and predictive intervals for every family/link other than Gaussian identity require ‘"response"‘. Default is ‘"link"‘.
mode	Character scalar or ‘NULL‘. Options are ‘"fitted_cluster_conditional"‘, ‘"zero_random_effect"‘, ‘"population_marginal_mean"‘, and ‘"new_cluster_predictive_distribution"‘. ‘NULL‘ uses the legacy switches ‘include_random‘ and ‘include_serial‘. Default is ‘NULL‘.
include_random	Logical scalar. Whether fitted random-effect contributions are included when ‘mode = NULL‘. Default is ‘is.null(newdata)‘.
include_serial	Logical scalar. Whether fitted serial-mode contributions are included when ‘mode = NULL‘. Default is ‘is.null(newdata)‘.
allow_new_levels	Logical scalar. Whether unseen grouping levels are accepted and assigned zero fitted-mode contribution. Default is ‘TRUE‘.
offset	Numeric vector, scalar, or ‘NULL‘. Explicit additive offset for ‘newdata‘, in addition to formula offsets. Default is ‘NULL‘.
interval	Character scalar or ‘NULL‘. Options are ‘"none"‘, ‘"confidence"‘, and ‘"prediction"‘. ‘NULL‘ requests point predictions, except that a new-cluster predictive mode selects a prediction interval. Confidence intervals are unavailable in

population-marginal and new-cluster modes because their required parameter-and-variance-component delta propagation is not represented. Predictive intervals for every family/link other than Gaussian identity use actual-family mixture quantiles on the response scale; link-scale predictive intervals are unavailable. Default is 'NULL'.

`level` Numeric scalar or 'NULL'. Confidence level in '(0, 1)'; 'NULL' uses the fitted prediction setting or '0.95'. Default is 'NULL'.

`se.fit` Logical scalar. Whether to return delta-method estimation standard errors when no interval is requested. It is unavailable in population-marginal and new-cluster modes; their 'predictive_sd' is aleatory response variation, not an estimation standard error. Default is 'FALSE'.

`trials` Numeric vector, scalar, or 'NULL'. Positive integer binomial trial totals for 'newdata'; fitted-row totals are reused automatically. Grouped-binomial newdata require explicit totals for prediction intervals or new-cluster distributional output. Default is 'NULL'.

`joint` Logical scalar. Whether to return the full cross-row predictive covariance for a Gaussian-identity population-marginal or new-cluster prediction. The covariance is assembled separately for fixed-parameter, nested/crossed random-effect, serial, and observation components. Default is 'FALSE'.

`...` Additional arguments. Fit methods reject unused arguments; the tuned-object method forwards them to its final fit.

Details

For models containing penalized smooths, 'newdata' is transformed through the stored training blueprint: boundary knots, marginal bases, tensor products, penalty diagonalizations, centering/projection constraints, retained columns, and scaling are never re-estimated from prediction rows. Cyclic terms wrap periodically. Noncyclic terms follow their fitted 'extrapolation' rule ('"error"', '"constant"', or '"linear"'), and factor-by terms reject unseen levels. Smooth contributions use the fitted coefficients and term-specific smoothing parameters; prediction does not rerun smoothing selection. If coefficient penalties or whole-term smooth selection were active, direct Hessian/delta standard errors and confidence intervals are suppressed; use dependence-preserving bootstrap inference. A joint marginal predictive covariance remains available conditionally on the fitted coefficients, with a zero fixed-parameter component explicitly marked unavailable.

Value

A numeric prediction vector when neither an interval nor standard errors are requested. Estimation-uncertainty output contains 'fit', 'se.fit', 'lower', and 'upper'; distributional prediction output uses 'predictive_sd' instead of labeling aleatory variation as 'se.fit' and includes 'lower' and 'upper'. For non-Gaussian-identity conditional predictive intervals, 'fit' is the mean of the actual-family mixture over asymptotic fixed-parameter uncertainty and 'plug_in_fit' retains the inverse-link prediction at the fitted coefficients. With 'joint = TRUE', returns a 'MEMWAS_joint_prediction' list containing rowwise summaries, the full covariance matrix, and its named components.

Examples

```
set.seed(1L)
```

```
sim_data <- MEMWAS:::.simulate_panel_data(
  n_id = 6L, n_time = 3L, beta = c(x1 = 0.6),
  cor_matrix = matrix(1, 1L, 1L), intercept = 1,
  sigma_eps = 0.2, sigma_b = 0, autocor = "NONE"
)
fit <- fit_MEMWAS(
  y ~ x1, data = sim_data, id = "id", time = "time",
  random = NULL, autocor = "NONE", se_method = "none",
  control = list(n_starts = 1L, cold_start_verification = FALSE),
  verbose = FALSE
)
predict(fit, type = "response")
```

`rank_autocorrelation_structures` *Rank temporal autocorrelation structures*

Description

Fits and ranks candidate temporal covariance specifications by dependence-component grouped cross-validation, AIC, BIC, or reported log-likelihood. Candidate specifications may combine an outcome-loaded residual process with separate predictor-loaded residual processes. All viable candidates are evaluated on the same complete-case rows. Model approximation, covariance construction, likelihood evaluation, and optimization are performed by MEMWAS's registered C++ backend; grouped fold assignment and validation metrics are also native.

Usage

```
rank_autocorrelation_structures(
  formula,
  family = "gaussian",
  data,
  id,
  time,
  random = ~1,
  candidates = c("NONE", "AR(1)", "OU", "CS"),
  criterion = "grouped_cv",
  K = 5L,
  metric = "RMSE",
  cv_aggregate = c("median", "mean"),
  cv_prediction_mode = c("population_marginal_mean", "zero_random_effect"),
  fold_accept = c("operational", "strict"),
  random_cov = "diagonal",
  control = list(),
  method = "ML",
  approximation = "laplace",
  init_approximation = "variational_inference",
  quadrature_points = 7L,
```

```
  link = NULL,
  offset = NULL,
  weights = NULL,
  subset = NULL,
  na.action = NULL,
  theta = NULL,
  shape = NULL,
  L1_penalty = 0,
  L2_penalty = 0,
  mape_epsilon = 1e-08,
  seed = NULL,
  refit_best = TRUE,
  refit_se_method = c("hessian", "none"),
  keep_fits = FALSE,
  keep_fold_fits = FALSE,
  verbose = TRUE,
  ...
)

## S3 method for class 'MEMWAS_autocorrelation_ranking'
print(x, ..., digits = 4L)

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

## S3 method for class 'summary.MEMWAS_autocorrelation_ranking'
print(x, ..., digits = 4L)
```

Arguments

formula	Formula object. Two-sided response and fixed-effect formula. No default.
family	Character scalar or family object. Family accepted by ‘fit_MEMWAS()‘. Default is ‘"gaussian"‘.
data	Data frame. Contains all model, grouping, time, offset, weight, subset, and serial-loading variables. No default.
id	Character scalar. Primary longitudinal-cluster column. Grouped CV keeps every primary ID and every connected random-effect grouping level in one fold. No default.
time	Character scalar. Measurement-time column. No default.
random	Random-effects specification accepted by ‘fit_MEMWAS()‘. Default is ‘~ 1‘.
candidates	Character vector or list. A character vector ranks outcome-loaded structures. List entries may be structure names, ‘serial_component()‘ objects, structured components, or lists containing ‘autocor‘, ‘serial‘, ‘residual_autocor‘, and/or ‘predictor_autocor‘. Named entries supply display labels. At least two are required. Default is ‘c("NONE", "AR(1)", "OU", "CS")‘.
criterion	Character scalar. One of ‘"grouped_cv"‘, ‘"AIC"‘, ‘"BIC"‘, or ‘"logLik"‘. Default is ‘"grouped_cv"‘.

`K` Integer scalar. Number of dependence-component folds. It must not exceed the number of connected components induced jointly by the primary ID and every random-effect grouping factor. Default is ‘5L‘.

`metric` Character scalar. Grouped-CV metric: ‘"RMSE"‘, ‘"MAE"‘, ‘"MSE"‘, ‘"MAPE"‘, or ‘"SMAPE"‘. Default is ‘"RMSE"‘.

`cv_aggregate` Character scalar. Use the fold ‘"median"‘ or ‘"mean"‘ as the grouped-CV ranking value. Default is ‘"median"‘.

`cv_prediction_mode`
Character scalar. Held-out dependence-component prediction mode: ‘"population_marginal_mean"‘ or ‘"zero_random_effect"‘. Default is ‘"population_marginal_mean"‘.

`fold_accept` Character scalar. Accept ‘"operational"‘ or only ‘"strict"‘ convergence in each fold and criterion fit. Default is ‘"operational"‘.

`random_cov` Character scalar or term-wise covariance specification. Default is ‘"diagonal"‘.

`control` Named list. Computational controls accepted by ‘fit_MEMWAS()‘. Ranking disables optional screening and diagnostics. Default is ‘list()‘.

`method` Character scalar. ‘"ML"‘ or an eligible ‘"REML"‘ fit. Default is ‘"ML"‘.

`approximation` Character scalar. Final approximation accepted by ‘fit_MEMWAS()‘. Default is ‘"laplace"‘.

`init_approximation`
Character scalar. Initialization approximation. Default is ‘"variational_inference"‘.

`quadrature_points`
Integer scalar. Positive quadrature order. Default is ‘7L‘.

`link` Character scalar or ‘NULL‘. Optional link override. Default is ‘NULL‘.

`offset` Numeric vector, expression, column name, or ‘NULL‘. Explicit linear-predictor offset. Default is ‘NULL‘.

`weights` Numeric vector, expression, column name, or ‘NULL‘. Non-negative observation weights. Default is ‘NULL‘.

`subset` Logical/integer expression or ‘NULL‘. Initial row selection. Default is ‘NULL‘.

`na.action` Function, character scalar, or ‘NULL‘. ‘"na.fail"‘ rejects incomplete candidate data; other supported actions use one common omitted sample across viable candidates. Default is ‘NULL‘.

`theta` Numeric scalar or ‘NULL‘. Negative-binomial dispersion. Default is ‘NULL‘.

`shape` Numeric scalar or ‘NULL‘. Gamma shape. Default is ‘NULL‘.

`L1_penalty` Numeric scalar. Common non-negative L1 penalty. Default is ‘0‘.

`L2_penalty` Numeric scalar. Common non-negative L2 penalty. Default is ‘0‘.

`mape_epsilon` Numeric scalar. Positive MAPE/SMAPE safeguard. Default is ‘1e-8‘.

`seed` Integer scalar or ‘NULL‘. Seed for a package-local native grouped-fold stream; it does not alter the global random-number state. ‘NULL‘ uses and advances the current ’R’ stream. Default is ‘NULL‘.

`refit_best` Logical scalar. Whether to refit the selected candidate on all common rows. Default is ‘TRUE‘.

refit_se_method	Character scalar. ‘"hessian"‘ or ‘"none"‘ for the selected full-data refit. Default is ‘"hessian"‘.
keep_fits	Logical scalar. Whether to retain full criterion fits. Default is ‘FALSE‘.
keep_fold_fits	Logical scalar. Whether to retain accepted fold fits. Default is ‘FALSE‘.
verbose	Logical scalar. Whether to report progress. Default is ‘TRUE‘.
...	Additional arguments. No additional arguments are accepted.
x	Object of class ‘MEMWAS_autocorrelation_ranking‘. No default.
digits	Integer scalar. Number of significant digits printed. Default is ‘4L‘.
object	Object of class ‘MEMWAS_autocorrelation_ranking‘. No default.

Details

For subject ‘i‘, the native conditional covariance may contain

$$V_i = K_{0i}(\theta_0) + \sum_j \operatorname{diag}(x_{ij}) K_{ji}(\theta_j) \operatorname{diag}(x_{ij}),$$

where ‘K0‘ is an optional outcome-loaded residual process and each ‘Kj‘ is an independently parameterized predictor-loaded residual process. This models residual temporal heterogeneity associated with observed predictor loadings; it does not fit a joint stochastic time-series model for the predictors.

AIC, BIC, and log-likelihood are accepted only when the fit reports the requested criterion as available. Penalized AIC and BIC additionally require effective degrees of freedom from a valid generalized trace; an active-count fallback is diagnostic only. BIC uses the number of positive-weight connected components in the joint primary-ID/random-grouping graph and requires at least two components. Consequently, a fully connected crossed model can have an available AIC but an unavailable BIC. Grouped CV forms the same connected components of the row-level dependence graph induced by the primary ID and all nested or crossed random-effect grouping factors. Each component is assigned intact to one fold, so no shared latent grouping level can leak between training and validation rows. The same native assignment is used for every candidate, and every fold must fit, predict, and return a finite metric.

Value

Object of class ‘MEMWAS_autocorrelation_ranking‘. It contains a sorted ‘ranking‘ table, fold-level results, common analysis-row indices, the selected candidate, an optional selected refit, and optionally retained criterion/fold fits.

‘print()‘ returns ‘x‘ invisibly. ‘summary()‘ returns an object of class ‘summary.MEMWAS_autocorrelation_ranking‘ containing ranking and fold-failure summaries.

Examples

```
set.seed(1L)
sim_data <- MEMWAS:::.simulate_panel_data(
  n_id = 6L, n_time = 3L, beta = c(x1 = 0.6),
  cor_matrix = matrix(1, 1L, 1L), intercept = 1,
```

```
  sigma_eps = 0.2, sigma_b = 0, autocor = "NONE"
)
ranking <- rank_autocorrelation_structures(
  y ~ x1, data = sim_data, id = "id", time = "time", random = NULL,
  candidates = c(independent = "NONE", ar1 = "AR(1)"),
  criterion = "AIC", refit_best = FALSE,
  control = list(n_starts = 1L, cold_start_verification = FALSE),
  verbose = FALSE
)
ranking$ranking
```

`screen_MEMWAS_nonlinearity` *Screen MEMWAS fixed effects for nonlinearity*

Description

Evaluates restricted-cubic-spline or native univariate P-spline additions to an unpenalized ML–Laplace null model. The function can be called independently of 'fit_MEMWAS()' so that screening is performed only when explicitly requested.

Usage

```
screen_MEMWAS_nonlinearity(
  object,
  nonlinear_predictors = NULL,
  nonlinear_alpha = 0.05,
  spline_knots = 4L,
  spline_probs = NULL,
  nonlinear_screening_method = "nuisance_adjusted_score",
  nonlinear_multiplicity = "holm",
  nonlinear_bootstrap_reps = 499L,
  nonlinear_bootstrap_seed = NULL,
  verbose = TRUE,
  nonlinear_spline = "restricted_cubic",
  smooth_control = list(),
  ...
)

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

Arguments

`object`	Object of class 'MEMWAS_fit'. The null model must be operationally converged, fitted by ML with the Laplace approximation, retain a marginal log-likelihood, use no elastic-net or serial-covariance penalty, contain no smooth

with a positive or automatic smoothing parameter, must not result from grouped-CV whole-term smooth selection, and must not already contain terms selected by this nonlinear screening workflow. A pre-existing smooth is admissible only when every lambda is fixed at zero. No default.

`nonlinear_predictors`
: Character vector or 'NULL'. Names of numeric fixed-effect predictors to screen. 'NULL' selects all eligible numeric predictors having at least five finite unique values. Default is 'NULL'.

`nonlinear_alpha`
: Numeric scalar. Family-wise selection threshold in '(0, 1)'. Default is '0.05'.

`spline_knots`
: Integer scalar. Number of restricted-cubic-spline knots or initial P-spline basis dimension; at least three are required. Default is '4L'.

`spline_probs`
: Numeric vector or 'NULL'. Strictly increasing probabilities in '[0, 1]'; the length must equal 'spline_knots'. 'NULL' uses the package probabilities. It is unavailable for P-spline screening. Default is 'NULL'.

`nonlinear_screening_method`
: Character scalar. Options are '"nuisance_adjusted_score"', which uses one shared primary-cluster maxT multiplier bootstrap, and '"likelihood_ratio"', which refits each candidate model. Default is '"nuisance_adjusted_score"'.

`nonlinear_multiplicity`
: Character scalar. P-value adjustment for the likelihood-ratio option. Options are '"holm"', '"hochberg"', '"hommel"', '"bonferroni"', '"BH"', '"BY"', and '"none"'. It is not used by the shared maxT option. Default is '"holm"'.

`nonlinear_bootstrap_reps`
: Integer scalar. Number of Rademacher multiplier-bootstrap replicates for score screening; at least 99 are required. Default is '499L'.

`nonlinear_bootstrap_seed`
: Integer scalar or 'NULL'. Non-negative seed for a package-local native score-screening stream; it does not alter the global random-number state. 'NULL' uses and advances the current 'R' stream. Default is 'NULL'.

`verbose`
: Logical scalar. Whether to report predictor-level screening progress. Default is 'TRUE'.

`nonlinear_spline`
: Character scalar. Candidate engine: '"restricted_cubic"' or '"pspline"'. The P-spline path constructs a native univariate B-spline basis with a coefficient-difference penalty, keeps its smoothing parameter separate from elastic-net shrinkage, and reports penalty-adjusted effective degrees of freedom. Default is '"restricted_cubic"'.

`smooth_control`
: Named list used only by P-spline screening. Basis controls are scalar 'degree', scalar 'difference_order', a strictly positive scalar 'lambda_initial', and the other construction defaults documented by 'fit_MEMWAS()'. Terms that pass the screen retain 'lambda = "auto"'; the subsequent fitted model selects each smoothing parameter by dependence-component grouped cross-validation using 'optimizer', 'cv_folds', 'log_lambda_range', 'lambda_base', 'grid_size', 'max_iter', 'tolerance', 'metric', 'seed', and 'selection_tolerance'. Default is 'list()'.

...	Additional arguments. No additional arguments are accepted.
x	Object of class 'MEMWAS_nonlinearity_screen'. No default.
digits	Integer scalar. Number of significant digits used for numeric output. Default is 'max(3L, getOption("digits") - 3L)'.

Details

Screening never runs merely because the P-spline engine exists. Direct use requires this function call; integrated use requires 'screen_nonlinear = TRUE', and 'run_checks_and_screening = FALSE' suppresses it. An explicit 'smooth' term in 'fit_MEMWAS()' is fitted as requested but does not trigger screening of other predictors.

Internally, 'fit_MEMWAS()' may supply the same estimator output through a lightweight 'MEMWAS_numerical_fit' baseline. This internal path uses the identical objective, optimizer, convergence rules, and retained preparation, while deliberately omitting Hessian and covariance-rank construction.

The score method forms candidate spline blocks from retained analysis rows, residualizes them against the null fixed-effect design using model-based Fisher weights, and calibrates all candidate statistics with one shared maxT distribution. Its single-cluster bootstrap requires every random-effect grouping factor to be nested within the primary ID. For crossed grouping structures, use '"likelihood_ratio"'. With the P-spline engine, the displayed candidate degrees of freedom are the generalized trace of the penalized smoother rather than its raw basis count.

The likelihood-ratio method reconstructs the fixed-effect design for every augmented formula and refits each nested candidate under ML–Laplace estimation. A P-spline candidate uses the term-specific roughness penalty at 'smooth_control$lambda_initial', and its comparison uses the smooth effective degrees of freedom. Selected P-spline terms are returned as smooth specifications with automatic smoothing for the final fit. Binomial trial totals, offsets, subsets, and missing-data handling retained by the null fit are reused during reconstruction. Smooth coefficients remain in the fixed-coefficient block, so the candidate mean and serial covariance are estimated jointly.

Value

Object of class 'MEMWAS_nonlinearity_screen'. It contains the complete screening table, selected predictors, knot or smooth definitions, candidate names, selected formula, selected data, call, and method settings. P-spline rows report effective, not raw-basis, degrees of freedom.

The input object, invisibly.

Examples

```
set.seed(1L)
sim_data <- MEMWAS:::.simulate_panel_data(
  n_id = 6L, n_time = 3L, beta = c(x1 = 0.6),
  cor_matrix = matrix(1, 1L, 1L), intercept = 1,
  sigma_eps = 0.2, sigma_b = 0, autocor = "NONE"
)
intercept_fit <- fit_MEMWAS(
  y ~ 1, data = sim_data, id = "id", time = "time",
  random = NULL, autocor = "NONE", se_method = "none",
  control = list(n_starts = 1L, cold_start_verification = FALSE),
  verbose = FALSE
```

```
)
empty_screen <- screen_MEMWAS_nonlinearity(intercept_fit, verbose = FALSE)
empty_screen$summary
```

`serial_component` *Define a MEMWAS serial covariance component*

Description

Describes one latent serial coefficient process for use in ‘fit_MEMWAS()‘ or ‘tune_MEMWAS()‘. Components may load the outcome uniformly or load one numeric predictor/design column.

Usage

```
serial_component(
  structure = "AR(1)",
  predictor = NULL,
  design = NULL,
  order = NULL,
  name = NULL,
  control = list()
)

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

Arguments

`structure`	Character scalar. Options are ‘"NONE"‘, ‘"AR(1)"‘, ‘"OU"‘, ‘"expOU"‘, ‘"AR(p)"‘, ‘"ARMA(1,1)"‘, ‘"CS"‘, ‘"TOEP"‘, and ‘"UN"‘. Orders may be encoded as ‘"AR(3)"‘ or ‘"TOEP(4)"‘. Default is ‘"AR(1)"‘.
`predictor`	Character scalar or ‘NULL‘. Name of one numeric predictor whose values load the serial coefficient process. Supply either ‘predictor‘ or ‘design‘. Default is ‘NULL‘.
`design`	One-sided formula, character scalar, or ‘NULL‘. Its model matrix must contain exactly one numeric loading column. Supply either ‘design‘ or ‘predictor‘. Default is ‘NULL‘.
`order`	Integer scalar or ‘NULL‘. Positive order for ‘"AR(p)"‘ or ‘"TOEP"‘; it must agree with any order encoded in ‘structure‘. Default is ‘NULL‘.
`name`	Character scalar or ‘NULL‘. Optional non-empty component name used for covariance parameters, contributions, and latent modes. Default is ‘NULL‘.
`control`	Named list. Supported entries are ‘serial_time_scale‘ (alias ‘time_scale‘), ‘allow_negative_autocor‘ (alias ‘allow_negative‘), and ‘max_unstructured_times‘. Default is ‘list()‘.
`x`	Object of class ‘MEMWAS_serial_component‘. The component to print. No default.
`...`	Additional arguments. No additional arguments are accepted by the print method.

Details

Except for compound symmetry, one latent state is created for each distinct ‘(id, time)‘ pair, so duplicate rows at the same time share the same state. Compound symmetry is observation-indexed: its diagonal is one and every pair of distinct rows, including equal-time rows, has the fitted off-diagonal correlation. With the default non-negative AR(1) convention, ‘rho = tanh(raw_rho)^2‘; OU and expOU use ‘range = raw_range^2‘. Thus a raw dependence coordinate of zero is exact independence. AR(1) uses integer scaled lags; when negative correlation is enabled, ‘rho = tanh(raw_rho)‘.

Value

Object of class ‘MEMWAS_serial_component‘ containing the normalized structure, loading specification, order, component name, and controls.

Examples

```
serial_component("AR(1)", name = "within_subject")
```

---

summary.MEMWAS_nonlinearity_screen

Summarize a MEMWAS nonlinearity screen

Description

Creates and prints a compact screening summary without copying the retained analysis data or internal basis matrices into the summary object. Candidate results retain smooth class, penalty, smoothing-parameter, effective-degree- of-freedom, and selection columns when the screening engine supplied them.

Usage

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

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

Arguments

`object`	Object of class ‘MEMWAS_nonlinearity_screen‘. No default.
`...`	Additional arguments. No additional arguments are accepted.
`x`	Object of class ‘summary.MEMWAS_nonlinearity_screen‘. No default.
`digits`	Integer scalar. Number of significant digits used for numeric output. Default is ‘max(3L, getOption("digits") - 3L)‘.

Value

‘summary()‘ returns an object of class ‘summary.MEMWAS_nonlinearity_screen‘; its print method returns that object invisibly.

Index