Package 'magp'

September 24, 2026

Title Mapping-Based Additive Gaussian Process Models

Version 0.12.0

Description Fits mapping-based additive Gaussian process models for experiments in which each component has both a quantitative level and a position in an ordered sequence. Two model structures are available: a compact two-dimensional mapping and a full mapping with one fewer dimension than the number of components. Both models support parameter estimation, point prediction, and plug-in predictive uncertainty. Input checks validate the sequence data and apply consistent scaling to the quantitative inputs. Computationally intensive covariance and gradient calculations are implemented in C++ with 'Rcpp'. Initial-design functions combine a space-filling Latin hypercube with sequence permutations. The sequence portion can be generated randomly or optimized with simulated annealing or space-filling threshold accepting. Expected improvement can be optimized over both parts of the input, and a sequential interface supports Bayesian optimization of an expensive user-supplied objective. An integrated workflow can generate the initial design, evaluate the objective, and continue the sequential search in one call. The model was introduced by Xiao et al.
(2024)
<doi:10.1080/01621459.2022.2123335>.

License MIT + file LICENSE

URL https://github.com/twskr/magp,
https://CRAN.R-project.org/package=magp

BugReports https://github.com/twskr/magp/issues

Encoding UTF-8

Imports Rcpp, nloptr, parallel, stats

LinkingTo Rcpp

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation yes

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

Author Tony Wang [aut, cre, cph],
Qian Xiao [aut, cph],
Yaping Wang [cph],
Abhyuday Mandal [cph],
Xinwei Deng [cph]

Maintainer Tony Wang <wangtony883@gmail.com>

Repository CRAN

Date/Publication 2026-09-24 16:10:07 UTC

Contents

`magp2d_fit` *Fit a MaGP model with a two-dimensional sequence map*

Description

Fits an additive Gaussian process for data that combine component amounts with a component order. Sequence positions are represented by points in a compact two-dimensional latent map.

Usage

```
magp2d_fit(
  X,
  y = NULL,
  q = NULL,
  tau = 0.001,
  maxeval = 500,
  xtol_rel = 1e-05,
  lb_sigma = 10,
  ub_sigma = 1000,
  lb_theta = 0.5,
  ub_theta = 1000,
  lb_delta = -1,
  ub_delta = 1,
  seed = NULL,
  n_starts = 1,
  workers = 1
)
```

Arguments

`X`	A numeric matrix or data frame. The first `q` input columns contain quantitative levels and the next `q` columns contain sequence positions. Each sequence row must be a permutation of `1:q`. A column named `y` may be included as the response.
`y`	An optional numeric response vector. It may be omitted when `X` contains a response column named `y`.
`q`	The number of components. It is inferred from the number of input columns when omitted. The two-dimensional model requires at least three components; the full model requires at least two.
`tau`	A fixed nonnegative nugget variance added to the covariance diagonal.
`maxeval`	Maximum number of objective evaluations used by `nloptr`.
`xtol_rel`	Relative parameter tolerance used by `nloptr`.
`lb_sigma`, `ub_sigma`	Lower and upper bounds for the additive variance parameters.
`lb_theta`, `ub_theta`	Lower and upper bounds for the quantitative correlation parameters.
`lb_delta`, `ub_delta`	Lower and upper bounds for the mapping parameters.
`seed`	An optional nonnegative integer used to generate the initial parameter vectors. The first start retains the result produced by this seed when `n_starts = 1`.
`n_starts`	Number of independent parameter starts. The fitted object contains the result with the lowest objective among the converged starts.
`workers`	Number of local worker processes. Values greater than one use a socket cluster and are capped at two or `n_starts`, whichever is lower.

Details

The covariance is a sum of component-specific terms. Each term combines the distance between two quantitative levels with the distance between their mapped sequence positions. The variance, correlation, and mapping parameters are estimated together with bounded optimization.

Quantitative columns outside `[0, 1]` are transformed with column-wise min-max scaling. Their training ranges are stored in the fitted object and reused for prediction. Columns already in `[0, 1]` are left unchanged.

When several starts are requested, each start receives a separate seed. Supplying `seed` makes the starts and the selected fit reproducible for both sequential and parallel execution. Start-level objective values, convergence codes, warnings, and errors are stored in `fit$multistart$starts`.

Value

An object of class `magp2d`.

Examples

```
train <- read.table(
  system.file("extdata", "example_train.txt", package = "magp"),
  header = TRUE
)
fit <- magp2d_fit(train, seed = 1, n_starts = 2)
fit
```

---

magp2d_rmse *Calculate root-mean-squared prediction error*

Description

Calculates the root-mean-squared error between predicted and observed values.

Usage

```
magp2d_rmse(pred, actual)
```

Arguments

`pred`	Numeric vector of predictions.
`actual`	Numeric vector of observed values with the same length as `pred`.

Value

A single nonnegative numeric value.

`magpfull_fit` *Fit a MaGP model with a full sequence map*

Description

Fits the quantitative-sequence model with `q - 1` latent mapping dimensions, giving the sequence positions a less constrained coordinate representation.

Usage

```
magpfull_fit(
  X,
  y = NULL,
  q = NULL,
  tau = 0.001,
  maxeval = 500,
  xtol_rel = 1e-05,
  lb_sigma = 10,
  ub_sigma = 1000,
  lb_theta = 0.5,
  ub_theta = 1000,
  lb_delta = -1,
  ub_delta = 1,
  seed = NULL,
  n_starts = 1,
  workers = 1
)
```

Arguments

`X`	A numeric matrix or data frame. The first `q` input columns contain quantitative levels and the next `q` columns contain sequence positions. Each sequence row must be a permutation of `1:q`. A column named `y` may be included as the response.
`y`	An optional numeric response vector. It may be omitted when `X` contains a response column named `y`.
`q`	The number of components. It is inferred from the number of input columns when omitted. The two-dimensional model requires at least three components; the full model requires at least two.
`tau`	A fixed nonnegative nugget variance added to the covariance diagonal.
`maxeval`	Maximum number of objective evaluations used by `nloptr`.
`xtol_rel`	Relative parameter tolerance used by `nloptr`.
`lb_sigma, ub_sigma`	Lower and upper bounds for the additive variance parameters.

`lb_theta`, `ub_theta`
: Lower and upper bounds for the quantitative correlation parameters.

`lb_delta`, `ub_delta`
: Lower and upper bounds for the mapping parameters.

`seed`
: An optional nonnegative integer used to generate the initial parameter vectors. The first start retains the result produced by this seed when `n_starts = 1`.

`n_starts`
: Number of independent parameter starts. The fitted object contains the result with the lowest objective among the converged starts.

`workers`
: Number of local worker processes. Values greater than one use a socket cluster and are capped at two or `n_starts`, whichever is lower.

Details

The data layout, quantitative scaling, and covariance construction are the same as in `magp2d_fit()`. The difference is the number of latent coordinates used to represent the sequence positions. With more than one start, the function keeps the converged result with the lowest objective and records the outcome of every start in `fit$multistart$starts`.

Value

An object of class `magpfull`.

Examples

```
train <- read.table(
  system.file("extdata", "example_train.txt", package = "magp"),
  header = TRUE
)
fit <- magpfull_fit(train, seed = 1, n_starts = 2)
fit
```

---

magp_bayes_optimize *Continue Bayesian optimization from completed experiments*

Description

Use this function when initial experiments and their responses are already available. It fits a MaGP model, selects a new input with expected improvement, evaluates `FUN`, and adds the new result to the data. This process repeats for at most `n_iter` new evaluations.

Usage

```
magp_bayes_optimize(
  FUN,
  X,
  y = NULL,
```

```
  model = c("2d", "full"),
  direction = c("minimize", "maximize"),
  n_iter = 10L,
  xi = 0,
  stop_ei = 0,
  stop_patience = 3L,
  seed = NULL,
  fit_control = list(),
  acquisition_control = list(),
  objective_args = list(),
  verbose = TRUE
)
```

Arguments

FUN	Function that evaluates one experiment. Its argument names must match the columns of X. It must return one finite number, or a list with one finite number named Score or Value.
X	Initial inputs as a numeric matrix or data frame. The first half of the columns contains quantitative values. The second half contains the sequence positions, with each row forming a permutation of 1:q. X may also contain a response column named y.
y	Numeric responses for the rows of X. Leave this as NULL when X contains a column named y.
model	MaGP mapping to fit. Use "2d" for the compact mapping or "full" for the full mapping.
direction	Use "minimize" when smaller responses are better and "maximize" when larger responses are better.
n_iter	Maximum number of new experiments to evaluate.
xi	Nonnegative expected-improvement offset. The default, 0, uses the current best response as the improvement target. Larger values require a candidate to exceed that target by more.
stop_ei	Nonnegative early-stopping threshold for expected improvement.
stop_patience	Number of consecutive selected points with expected improvement less than or equal to stop_ei required to stop early.
seed	Optional nonnegative whole-number seed for reproducible model starts and acquisition searches. The caller's random-number state is preserved.
fit_control	Optional named list passed to magp2d_fit() or magpfull_fit(). Common choices include tau, maxeval, n_starts, and workers. This function supplies X, y, and seed.
acquisition_control	Optional named list passed to magp_next_point(). Common choices include lower, upper, sequences, n_starts, workers, and maxit. This function supplies the fitted model, direction, xi, current best response, and seed.
objective_args	Optional named list of fixed arguments passed to FUN in addition to the input columns.

`verbose` If `TRUE`, print the observed response and expected improvement after each new evaluation.

Value

An object of class `magp_bayes_opt`. The final fitted model includes every completed evaluation.

How the loop works

The initial rows of `X` are treated as completed experiments and are not evaluated again. Each iteration performs four steps:

1. fit the selected MaGP model to all results collected so far;
2. use `magp_next_point()` to select an unobserved input;
3. call `FUN` at that input; and
4. add the response and refit the model.

`FUN` receives one named argument for each input column. For columns `A`, `B`, `a`, and `b`, for example, the call is equivalent to `FUN(A = ..., B = ..., a = ..., b = ...)`.

Reading the result

The returned object contains:

- `call`: the function call;
- `best_point`: the input row with the best observed response;
- `best_value`: the best observed response;
- `best_index`: the row number of the best result in `X` and `y`;
- `history`: the initial and newly evaluated rows in evaluation order;
- `model`: the final fitted MaGP model;
- `X` and `y`: all inputs and responses used by the final model;
- `acquisitions`: details from each call to `magp_next_point()`;
- `mapping` and `direction`: the model and optimization direction;
- `iterations_requested` and `iterations_completed`: the requested and completed numbers of new evaluations;
- `stop_reason`: why the loop ended;
- `xi`, `stop_ei`, and `stop_patience`: the acquisition and stopping settings; and
- `fit_control` and `acquisition_control`: the control lists used in the search.

References

Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient Global Optimization of Expensive Black-Box Functions. Journal of Global Optimization, 13, 455-492. doi:10.1023/A:1008306431147.

Examples

```
design <- magp_initial_design(n = 6, q = 3, seed = 1)
objective <- function(quantity_1, quantity_2, quantity_3,
                      sequence_1, sequence_2, sequence_3) {
  quantities <- c(quantity_1, quantity_2, quantity_3)
  positions <- c(sequence_1, sequence_2, sequence_3)
  -sum((quantities - c(0.2, 0.6, 0.8))^2) -
    0.02 * sum((positions - c(1, 3, 2))^2)
}
initial_y <- apply(design$design, 1L, function(row) {
  do.call(objective, as.list(row))
})
result <- magp_bayes_optimize(
  FUN = objective,
  X = design$design,
  y = initial_y,
  direction = "maximize",
  n_iter = 1,
  seed = 2,
  fit_control = list(maxeval = 100),
  acquisition_control = list(n_starts = 2, maxit = 20),
  verbose = FALSE
)
result$best_point
result$best_value
result$history
```

---

magp_bayes_optimize_from_scratch

Start Bayesian optimization before any experiments have been run

---

Description

Use this function when no initial data are available. It creates an initial quantitative-sequence design, evaluates FUN at those runs, and then uses MaGP expected improvement to select later runs.

Usage

```
magp_bayes_optimize_from_scratch(
  FUN,
  n_initial,
  q,
  model = c("2d", "full"),
  direction = c("minimize", "maximize"),
  n_iter = 10L,
  xi = 0,
  stop_ei = 0,
```

```
  stop_patience = 3L,
  seed = NULL,
  design_control = list(),
  fit_control = list(),
  acquisition_control = list(),
  objective_args = list(),
  verbose = TRUE
)
```

Arguments

FUN	Function that evaluates one experiment. It must accept the generated arguments quantity_1, ..., quantity_q, sequence_1, ..., sequence_q. It must return one finite number, or a list with one finite number named Score or Value.
n_initial	Number of experiments in the generated initial design.
q	Number of components. It must be at least three.
model	MaGP mapping to fit. Use "2d" for the compact mapping or "full" for the full mapping.
direction	Use "minimize" when smaller responses are better and "maximize" when larger responses are better.
n_iter	Maximum number of new experiments selected after the initial design has been evaluated.
xi	Nonnegative expected-improvement offset. The default, 0, uses the current best response as the improvement target.
stop_ei	Nonnegative early-stopping threshold for expected improvement.
stop_patience	Number of consecutive selected points with expected improvement less than or equal to stop_ei required to stop early.
seed	Optional nonnegative whole-number seed. It makes the initial design, model starts, acquisition searches, and random values produced by FUN reproducible while preserving the caller's random-number state.
design_control	Optional named list passed to magp_initial_design(). Use sequence_method to choose "random", "sfta", or "sann". Other common choices include sequence_maxit, sfta_control, quantity_maxit, and alignment_maxit. This function supplies n, q, and seed.
fit_control	Optional named list passed to magp2d_fit() or magpfull_fit(). Common choices are maxeval, n_starts, and workers. This function supplies X, y, and seed.
acquisition_control	Optional named list passed to magp_next_point(). Common choices are lower, upper, sequences, n_starts, workers, and maxit. This function supplies the fitted model, direction, xi, current best response, and seed.
objective_args	Optional named list of fixed arguments passed to FUN in addition to the generated input columns.
verbose	If TRUE, print one line after each initial and sequential evaluation.

Value

An object of class `magp_bayes_opt` containing the initial design, all completed evaluations, and the final fitted model.

How to use this function

Define `FUN`, choose `n_initial` and `q`, and state whether the response should be minimized or maximized. The function then:

1. creates an initial design;
2. calls `FUN` once for each initial row;
3. fits the selected MaGP model; and
4. selects and evaluates as many as `n_iter` additional rows.

The generated columns are named `quantity_1` through `quantity_q` and `sequence_1` through `sequence_q`. Use `magp_bayes_optimize()` instead when initial experiments and responses already exist.

Reading the result

The result has the same main components as `magp_bayes_optimize()`, including `best_point`, `best_value`, `history`, the final `model`, and all recorded search settings. It also contains:

- `initial_design`: the generated quantitative-sequence design;
- `initial_response`: the response from each initial run;
- `initial_evaluations`: the number of initial runs; and
- `design_control`: the initial-design settings that were used; and
- `started_from_initial_design`: `TRUE`, marking that the workflow generated its own starting design.

Examples

```
objective <- function(quantity_1, quantity_2, quantity_3,
                      sequence_1, sequence_2, sequence_3) {
  quantities <- c(quantity_1, quantity_2, quantity_3)
  sequence <- c(sequence_1, sequence_2, sequence_3)
  -sum((quantities - c(0.2, 0.6, 0.8))^2) -
    0.01 * sum((sequence - c(1, 3, 2))^2)
}
result <- magp_bayes_optimize_from_scratch(
  FUN = objective,
  n_initial = 6,
  q = 3,
  direction = "maximize",
  n_iter = 1,
  seed = 1,
  design_control = list(
    sequence_maxit = 100,
    quantity_maxit = 100,
```

```
    alignment_maxit = 100
  ),
  fit_control = list(maxeval = 100),
  acquisition_control = list(n_starts = 2, maxit = 20),
  verbose = FALSE
)
result$initial_design$design
result$best_point
result$best_value
result$history
```

`magp_expected_improvement` *Score candidate experiments with expected improvement*

Description

Calculates expected improvement for one or more candidate rows. Higher values indicate candidates that offer a better combination of predicted improvement and uncertainty.

Usage

```
magp_expected_improvement(
  object,
  newdata,
  direction = c("minimize", "maximize"),
  best = NULL,
  xi = 0
)
```

Arguments

`object`	A fitted `magp2d` or `magpfull` model.
`newdata`	A numeric matrix or data frame accepted by the model's `stats::predict()` method.
`direction`	Use `"minimize"` when smaller responses are better and `"maximize"` when larger responses are better.
`best`	Optional response that a new point should improve upon. When it is omitted, the function uses the best observed response in `object`.
`xi`	Nonnegative improvement offset. The default is `0`. Larger values require a candidate to exceed `best` by more.

Details

Expected improvement uses the model's latent predictive mean and standard error. A candidate can receive a high value because its predicted response is good, its uncertainty is large, or both. Rows already present in the training data receive a value of zero. Predictions with variance below `1e-8` also receive zero.

Value

A nonnegative numeric vector with one expected-improvement value per row of `newdata`.

References

Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient Global Optimization of Expensive Black-Box Functions. Journal of Global Optimization, 13, 455-492. doi:10.1023/A:1008306431147.

Examples

```
train <- read.table(
  system.file("extdata", "example_train.txt", package = "magp"),
  header = TRUE
)
test <- read.table(
  system.file("extdata", "example_test.txt", package = "magp"),
  header = TRUE
)
fit <- magp2d_fit(train, seed = 1)
magp_expected_improvement(
  fit,
  test[1:3, ],
  direction = "minimize"
)
```

`magp_initial_design` *Construct a quantitative-sequence initial design*

Description

Builds an initial design in three steps. It first generates the sequence permutations, then constructs a maximin-style Latin hypercube for the quantitative levels, and finally aligns the two fixed designs by permuting whole quantitative rows. The final alignment preserves both the Latin hypercube and every sequence permutation.

Usage

```
magp_initial_design(
  n,
  q,
  pair_weight = 0.2,
  sequence_space_weight = 0.8,
  quantity_weight = 0.5,
  sequence_weight = 0.5,
  p = 15L,
  sequence_maxit = 10000L,
  quantity_maxit = 10000L,
  alignment_maxit = 10000L,
  sequence_temp = 0.1,
  quantity_temp = 0.01,
  alignment_temp = 0.001,
  tmax = 10L,
  initial_sequence = NULL,
  initial_quantity = NULL,
  seed = NULL,
  sequence_method = c("sann", "random", "sfta"),
  sfta_control = list()
)
```

Arguments

n	Number of design runs. Must be at least two.
q	Number of components. Must be at least three.
pair_weight	Nonnegative weight for ordered adjacent-pair balance in the sequence search.
sequence_space_weight	Nonnegative weight for Hamming-distance space filling in the sequence search.
quantity_weight	Nonnegative quantitative-distance weight in the joint criterion.
sequence_weight	Nonnegative sequence-distance weight in the joint criterion.
p	Positive whole-number exponent used in all three criteria.
sequence_maxit	Positive whole number of sequence-search iterations.
quantity_maxit	Positive whole number of quantitative-search iterations.
alignment_maxit	Positive whole number of row-alignment iterations.
sequence_temp	Positive initial temperature for the sequence search.
quantity_temp	Positive initial temperature for the quantitative search.
alignment_temp	Positive initial temperature for the alignment search.
tmax	Positive whole number of evaluations at each temperature.
initial_sequence	Optional n by q matrix of sequence positions.

`initial_quantity`
: Optional `n` by `q` Latin hypercube in `[0, 1]`.

`seed`
: Optional nonnegative whole-number seed. Separate deterministic seeds are used for the three stages, and the caller's random-number state is preserved.

`sequence_method`
: Method for the sequence portion. Choose `"random"`, `"sfta"` for space-filling threshold accepting, or `"sann"` for simulated annealing. The default is `"sann"` for backward compatibility.

`sfta_control`
: Named list of SFTA settings, passed to `magp_sequence_design()`. Used only for `sequence_method = "sfta"`.

Details

The sequence method does not change how quantitative levels are generated. With a fixed `seed`, all three methods use the same quantitative design before the final row-alignment step. The alignment can change its row order but not its values or Latin-hypercube structure.

Value

An object of class `magp_initial_design`. Its `design` element is an `n` by `2*q` matrix ready to use as the input to a MAGP fitting function. The first `q` columns contain quantitative levels and the last `q` columns contain sequence positions. The component searches and their criterion values are retained for inspection.

References

Xiao, Q., Wang, Y., Mandal, A., and Deng, X. (2024). Modeling and Active Learning for Experiments with Quantitative-Sequence Factors. Journal of the American Statistical Association. doi:10.1080/01621459.2022.2123335.

Examples

```
design <- magp_initial_design(
  n = 8,
  q = 4,
  sequence_maxit = 500,
  quantity_maxit = 500,
  alignment_maxit = 500,
  seed = 1
)
design$design
design$criteria

random <- magp_initial_design(
  8, 4, sequence_method = "random", seed = 1,
  quantity_maxit = 100, alignment_maxit = 100
)
sfta <- magp_initial_design(
  8, 4, sequence_method = "sfta", seed = 1, sequence_maxit = 200,
  quantity_maxit = 100, alignment_maxit = 100,
```

```
  sfta_control = list(nstarts = 2, ncalibrate = 50)
)
rbind(random = random$criteria, sfta = sfta$criteria)
```

`magp_joint_criterion` *Evaluate a complete quantitative-sequence initial design*

Description

Combines Euclidean distance for the quantitative portion with Hamming distance for the sequence portion. Smaller values indicate better joint separation of the design runs.

Usage

```
magp_joint_criterion(
  quantity,
  sequence,
  quantity_weight = 0.5,
  sequence_weight = 0.5,
  p = 15L
)
```

Arguments

`quantity`	Numeric matrix or data frame with values in `[0, 1]`.
`sequence`	Numeric matrix or data frame with the same dimensions as `quantity`. Every row must be a permutation of `1:q`.
`quantity_weight`	Nonnegative weight for quantitative distance.
`sequence_weight`	Nonnegative weight for sequence distance.
`p`	Positive whole-number exponent controlling emphasis on the least separated pairs of runs.

Value

One numeric criterion value. Smaller values are preferred.

References

Xiao, Q., Wang, Y., Mandal, A., and Deng, X. (2024). Modeling and Active Learning for Experiments with Quantitative-Sequence Factors. Journal of the American Statistical Association. doi:10.1080/01621459.2022.2123335.

Examples

```
quantity <- cbind(
  c(0.125, 0.375, 0.625, 0.875),
  c(0.625, 0.125, 0.875, 0.375),
  c(0.375, 0.875, 0.125, 0.625),
  c(0.875, 0.625, 0.375, 0.125)
)
sequence <- rbind(
  c(1, 2, 3, 4),
  c(3, 1, 4, 2),
  c(2, 4, 1, 3),
  c(4, 3, 2, 1)
)
magp_joint_criterion(quantity, sequence)
```

---

magp_next_point *Select the next quantitative-sequence experiment*

Description

Searches the allowed quantitative values and sequence permutations, then returns the unobserved point with the largest expected improvement.

Usage

```
magp_next_point(
  object,
  direction = c("minimize", "maximize"),
  xi = 0,
  best = NULL,
  lower = NULL,
  upper = NULL,
  sequences = NULL,
  max_sequences = 120L,
  n_starts = 5L,
  workers = 1L,
  maxit = 100L,
  factr = 1e+07,
  pgtol = 0,
  exclude_observed = TRUE,
  duplicate_tolerance = sqrt(.Machine$double.eps),
  seed = NULL
)
```

Arguments

object	A fitted magp2d or magpfull model.
direction	Use "minimize" when smaller responses are better and "maximize" when larger responses are better.
xi	Nonnegative improvement offset used in expected improvement.
best	Optional response that a new point should improve upon. When it is omitted, the function uses the best observed response in object.
lower, upper	Optional lower and upper bounds for the quantitative inputs. Supply one value for all components or one value per component. The defaults use the prediction ranges stored in the fitted model.
sequences	Optional matrix of candidate sequence permutations. When omitted, every permutation is used if their number does not exceed max_sequences; otherwise a reproducible sample is searched.
max_sequences	Largest number of sequence candidates generated when sequences is not supplied.
n_starts	Number of quantitative starting points searched for each sequence candidate.
workers	Number of local worker processes. Use 1 for sequential execution. At most two processes are used.
maxit	Maximum optimization iterations for each quantitative start.
factr, pgtol	Advanced convergence settings passed to stats::optim() for its "L-BFGS-B" method.
exclude_observed	If TRUE, do not return an input that is already in the training data.
duplicate_tolerance	Nonnegative absolute tolerance used to identify an observed input after the model's quantitative scaling is applied.
seed	Optional nonnegative whole-number seed for sequence sampling and quantitative starting points.

Value

An object of class magp_next_point containing the selected point, its prediction, and search diagnostics.

Sequence search

If sequences is supplied, only those rows are searched. Otherwise, the function searches every permutation when there are no more than max_sequences. For a larger sequence space, it searches a reproducible sample of max_sequences permutations when seed is supplied.

Reading the result

The returned object contains:

- call: the function call;

- point: the selected input as a one-row data frame;
- expected_improvement: the score of the selected point;
- predicted_mean and predicted_standard_error: the MaGP prediction;
- direction, best_observed, and xi: the expected-improvement settings;
- bounds: the quantitative lower and upper bounds;
- sequences and sequence_source: the permutations searched and how they were obtained;
- selected_sequence and selected_start: the winning search indices;
- n_starts: the number of quantitative starts per sequence;
- workers_requested, workers_used, and execution: the parallel-search settings actually used; and
- diagnostics: the result of every sequence and starting-point search.

Examples

```
train <- read.table(
  system.file("extdata", "example_train.txt", package = "magp"),
  header = TRUE
)
fit <- magp2d_fit(train, seed = 1)
next_run <- magp_next_point(
  fit,
  direction = "minimize",
  n_starts = 2,
  maxit = 20,
  seed = 2
)
next_run$point
next_run$expected_improvement
```

---

magp_quantitative_criterion

Evaluate a quantitative Latin hypercube

Description

Measures the separation of design rows under Euclidean distance. The criterion is an inverse-distance p-norm, so smaller values indicate a more space-filling design.

Usage

```
magp_quantitative_criterion(quantity, p = 15L)
```

Arguments

quantity	Numeric matrix or data frame with values in [0, 1].
p	Positive whole-number exponent controlling emphasis on the shortest pairwise distances.

Value

One numeric criterion value. Smaller values are preferred. A design with duplicate rows has value Inf.

Examples

```
quantity <- cbind(
  c(0.125, 0.375, 0.625, 0.875),
  c(0.625, 0.125, 0.875, 0.375)
)
magp_quantitative_criterion(quantity)
```

---

magp_quantitative_design

Construct the quantitative portion of an initial design

Description

Uses simulated annealing to improve a Latin hypercube under a maximin-style Euclidean-distance criterion. A candidate is created by swapping two values within one column, which preserves the Latin hypercube property.

Usage

```
magp_quantitative_design(
  n,
  q,
  p = 15L,
  maxit = 10000L,
  temp = 0.01,
  tmax = 10L,
  initial = NULL,
  seed = NULL
)
```

Arguments

n	Number of design runs. Must be at least two.
q	Number of quantitative variables. Must be positive.
p	Positive whole-number exponent controlling emphasis on the shortest pairwise distances.
maxit	Positive whole number of simulated-annealing iterations.
temp	Positive initial temperature passed to stats::optim().
tmax	Positive whole number of evaluations at each temperature.
initial	Optional n by q Latin hypercube with values in [0, 1].
seed	Optional nonnegative whole-number seed. When supplied, the function restores the caller's random-number state before returning.

Value

An object of class magp_quantitative_design, containing the optimized Latin hypercube, the starting design, criterion values, minimum distances, and search settings.

References

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by Simulated Annealing. Science, 220, 671-680. doi:10.1126/science.220.4598.671.

Examples

```
design <- magp_quantitative_design(
  n = 8,
  q = 4,
  maxit = 500,
  seed = 1
)
design$quantity
design$criterion
```

---

magp_sequence_criterion

Evaluate a sequence initial design

Description

Measures two properties of a sequence design: how evenly ordered adjacent component pairs are represented and how well separated the design rows are under Hamming distance. Smaller values indicate a better design.

Usage

```
magp_sequence_criterion(
  sequence,
  pair_weight = 0.2,
  space_weight = 0.8,
  p = 15L
)
```

Arguments

sequence	Numeric matrix or data frame. Every row must be a permutation of 1:q.
pair_weight	Nonnegative weight for ordered adjacent-pair balance.
space_weight	Nonnegative weight for Hamming-distance space filling.
p	Positive whole-number exponent controlling emphasis on the weakest pair counts and the smallest distances.

Details

Each row uses the same sequence format as the fitting functions. The value in column j is the position assigned to component j, and every row must be a permutation of 1:q.

Value

One numeric criterion value. Smaller values are preferred.

References

Xiao, Q., Wang, Y., Mandal, A., and Deng, X. (2024). Modeling and Active Learning for Experiments with Quantitative-Sequence Factors. Journal of the American Statistical Association. doi:10.1080/01621459.2022.2123335.

Examples

```
sequence <- rbind(
  c(1, 2, 3, 4),
  c(3, 1, 4, 2),
  c(2, 4, 1, 3),
  c(4, 3, 2, 1)
)
magp_sequence_criterion(sequence)
```

magp_sequence_design *Construct the sequence portion of an initial design*

Description

Constructs random sequences, or searches for a sequence design with balanced ordered adjacent pairs and well-separated rows using simulated annealing or space-filling threshold accepting (SFTA). A neighbor is generated by selecting one design row and exchanging two component positions, so every candidate remains a valid permutation.

Usage

```
magp_sequence_design(
  n,
  q,
  pair_weight = 0.2,
  space_weight = 0.8,
  p = 15L,
  maxit = 10000L,
  temp = 0.1,
  tmax = 10L,
  initial = NULL,
  seed = NULL,
  method = c("sann", "random", "sfta"),
  sfta_control = list()
)
```

Arguments

n	Number of design runs. Must be at least two.
q	Number of components. Must be at least three.
pair_weight	Nonnegative weight for ordered adjacent-pair balance.
space_weight	Nonnegative weight for Hamming-distance space filling.
p	Positive whole-number exponent controlling emphasis on the weakest pair counts and the smallest distances.
maxit	Positive whole number of simulated-annealing iterations, or the total number of Phase-II neighbor proposals for SFTA (across all rounds). Unused by method = "random".
temp	Positive initial temperature passed to stats::optim(). The default is scaled to the sequence-design criterion used here.
tmax	Positive whole number of evaluations at each temperature.
initial	Optional n by q matrix of component positions. Every row must be a permutation of 1:q.
seed	Optional nonnegative whole-number seed. When supplied, the function restores the caller's random-number state before returning.

`method` Sequence generation method: `"random"` samples permutations without optimizing them; `"sfta"` uses the two-phase search described below; `"sann"` retains the original simulated annealing and is the default for backward compatibility. With `"random"`, a supplied `initial` matrix is returned unchanged.

`sfta_control` Named list used only with `method = "sfta"`. Positive integer settings are `nstarts` (default 5 candidate designs in Phase I), `ncalibrate` (200 trial swaps for threshold calibration), `nrounds` (`min(10, maxit)` threshold rounds, at most `maxit`), and `max_proposals` (`max(1000, 50*n)`, capped at the integer limit; rejection-sampling budget per Phase-I design). Unknown or duplicated names are errors.

Details

This function constructs only the sequence portion of a quantitative- sequence initial design. Use `magp_initial_design()` to combine it with a quantitative Latin hypercube and improve the pairing of the two portions.

The SFTA option applies threshold accepting to the complete sequence-design criterion returned by `magp_sequence_criterion()`. It does not require responses and does not change the expected-improvement search used later in Bayesian optimization.

Phase I constructs `nstarts` complete designs by accepting candidate rows with probability equal to their minimum Hamming distance to already accepted rows divided by `q`. The best complete design, including the original starting design, enters Phase II. Sampling is bounded; when its budget or all `q!` permutations are exhausted, remaining rows are sampled uniformly. Repeated sequences are allowed (and unavoidable when `n > q!`). The diagnostics record these fallback counts.

Phase II swaps two entries of one execution-order row at a time. Thresholds use type-7 empirical quantiles of absolute criterion differences at the fixed Phase-I winner, at probabilities `0.5 * (1 - r/nrounds)`. The last threshold is explicitly set to zero for a final greedy round; a move is accepted only when its criterion increase is strictly below the threshold. The best visited design is retained, so the result cannot be worse than `initial`. Adjacent-pair counts and Hamming-distance histograms are updated in C++ after each swap. Sequence columns always contain component positions; execution-order conversion is internal.

`temp` and `tmax` only affect `"sann"`. Random generation avoids duplicate rows when `n <= q!`. Neither optimized method guarantees unique rows or a global optimum.

Value

An object of class `magp_sequence_design`, containing the optimized sequence matrix, the starting matrix, criterion values, and search settings. The sequence matrix is ready to use as the sequence half of a `magp` input matrix. `method_key` records the method selector. For SFTA, `sfta` contains resolved controls, Phase-I criteria and sampling counts, thresholds, accepted moves, best-criterion history, and evaluation counts (including threshold calibration and the final independent check).

References

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by Simulated Annealing. Science, 220, 671-680. doi:10.1126/science.220.4598.671.

Xiao, Q., Wang, Y., Mandal, A., and Deng, X. (2024). Modeling and Active Learning for Experiments with Quantitative-Sequence Factors. Journal of the American Statistical Association. doi:10.1080/01621459.2022.2123335.

Examples

```
design <- magp_sequence_design(
  n = 8,
  q = 4,
  maxit = 500,
  seed = 1
)
design$sequence
design$criterion

random <- magp_sequence_design(8, 4, method = "random", seed = 1)
sfta <- magp_sequence_design(
  8, 4, method = "sfta", maxit = 200, seed = 1,
  sfta_control = list(nstarts = 2, ncalibrate = 50)
)
c(random = random$criterion, sfta = sfta$criterion)
```

`predict.magp` *Predict outcomes from a fitted MaGP model*

Description

Returns predictions for new quantitative-sequence inputs. Plug-in standard errors and variances can be returned with the predictions.

Usage

```
## S3 method for class 'magp2d'
predict(
  object,
  newdata,
  se.fit = FALSE,
  type = c("script", "response", "latent"),
  ...
)

## S3 method for class 'magpfull'
predict(
  object,
  newdata,
  se.fit = FALSE,
  type = c("script", "response", "latent"),
  ...
)
```

Arguments

object	A fitted magp2d or magpfull object.
newdata	A numeric matrix or data frame with the same quantitative and sequence inputs used for fitting. A response column named y is ignored.
se.fit	Logical; if TRUE, return predictive standard errors and variances in addition to fitted values.
type	Prediction convention. "script" uses the fitted training-row convention when newdata exactly matches a training row. "response" includes the nugget variance for a future response, and "latent" returns uncertainty for the noise-free surface.
...	Additional arguments, currently unused.

Value

If se.fit = FALSE, a numeric vector of predictions. Otherwise, a list with components fit, se.fit, variance, and type.

Examples

```
train <- read.table(
  system.file("extdata", "example_train.txt", package = "magp"),
  header = TRUE
)
test <- read.table(
  system.file("extdata", "example_test.txt", package = "magp"),
  header = TRUE
)
fit <- magp2d_fit(train, seed = 1)
predict(fit, test[1:3, ], se.fit = TRUE, type = "response")
```

---

print.magp2d *Summarize a fitted two-dimensional MaGP model*

Description

Prints the model size, nugget variance, fitted mean, objective value, and optimizer status.

Usage

```
## S3 method for class 'magp2d'
print(x, ...)
```

Arguments

x	A magp2d model returned by magp2d_fit().
...	Unused.

Value

x, invisibly.

---

print.magpfull *Summarize a fitted full-mapping MaGP model*

Description

Prints the mapping dimension, model size, nugget variance, fitted mean, objective value, and optimizer status.

Usage

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

Arguments

x	A magpfull model returned by magpfull_fit().
...	Unused.

Value

x, invisibly.

---

print.magp_bayes_opt *Print a MaGP Bayesian optimization result*

Description

Print a MaGP Bayesian optimization result

Usage

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

Arguments

x	A magp_bayes_opt object returned by magp_bayes_optimize().
...	Unused.

Value

x, invisibly.

print.magp_initial_design *Print a quantitative-sequence initial design*

Description

Print a quantitative-sequence initial design

Usage

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

Arguments

x	A magp_initial_design object.
...	Additional arguments, currently unused.

Value

x, invisibly.

print.magp_next_point *Print a MaGP acquisition-search result*

Description

Print a MaGP acquisition-search result

Usage

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

Arguments

x	A magp_next_point object returned by magp_next_point().
...	Unused.

Value

x, invisibly.

print.magp_quantitative_design *Print a quantitative initial design*

Description

Print a quantitative initial design

Usage

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

Arguments

x	A magp_quantitative_design object.
...	Additional arguments, currently unused.

Value

x, invisibly.

print.magp_sequence_design *Print a sequence initial design*

Description

Print a sequence initial design

Usage

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

Arguments

x	A fitted magp_sequence_design object.
...	Additional arguments, currently unused.

Value

x, invisibly.

Index