Package: MOFAT (via r-universe)

October 26, 2026

Title Maximum One-Factor-at-a-Time Designs

Version 2.0

Imports gtools, Rcpp, SFDesign, scales, SLHD, spacefillr, stats, MaxPro, RcppHungarian, withr

LinkingTo Rcpp, RcppArmadillo

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Description Identifying important factors from a large number of potentially important factors in a highly nonlinear and computationally expensive black-box model is a difficult problem. Xiao, Joseph, and Ray (2022) <doi:10.1080/00401706.2022.2141897> proposed Maximum One-Factor-at-a-Time (MOFAT) designs for this purpose. A MOFAT design is a collection of one-factor-at-a-time (OFAT) designs constructed to achieve maximal screening capability. Song and Joseph (2026) <doi:10.48550/arXiv.2609.02087> extended the MOFAT framework to incorporate nominal and discrete numeric factors. The resulting designs are implemented in the mofatQQ function. Yu and Joseph (2026) <doi:10.48550/arXiv.2606.02533> proposed Space-Filling One-Factor-at-a-Time (SOFT) designs, which achieve optimal space-filling properties while retaining the screening capability. Therefore, SOFT designs are useful for both factor screening and surrogate modeling. The work on MOFAT is supported by a U.S. National Science Foundation (NSF) grant CMMI-1921646 <https://www.nsf.gov/awardsearch/show-award/?AWD_ID=1921646>, and the work on MOFATQQ and SOFT is supported by a U.S. NSF grant DMS-2310637.

License GPL (>= 2)

Encoding UTF-8

NeedsCompilation yes

SystemRequirements On Linux, optional NVIDIA CUDA Toolkit (12.x or 13.x) and a supported host compiler; Metal is optional on macOS

Config/roxygen2/version 8.1.1

Author Difan Song [aut, cre], Wei-Yang Yu [aut], Qian Xiao [aut], V. Roshan Joseph [aut]

Maintainer Difan Song <dfsong96@gmail.com>

Config/pak/sysreqs nvidia-cuda-dev

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

Date/Publication 2026-10-26 13:10:20 UTC

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

RemoteRef HEAD

RemoteSha 5c9aed30a05e0dc62609b8c8e558425e35ad0aa3

Contents

MOFAT-package 2
a_soft 3
measure 5
mofat 6
mofat_backend_info 7
mofatQQ 8
Q 10
qmc_generate 11
soft 12

Index **14**

`MOFAT-package` *MOFAT: Maximum One-Factor-at-a-Time Designs*

Description

Identifying important factors from a large number of potentially important factors in a highly nonlinear and computationally expensive black-box model is a difficult problem. Xiao, Joseph, and Ray (2022) doi:10.1080/00401706.2022.2141897 proposed Maximum One-Factor-at-a-Time (MOFAT) designs for this purpose. A MOFAT design is a collection of one-factor-at-a-time (OFAT) designs constructed to achieve maximal screening capability. Song and Joseph (2026) doi:10.48550/arXiv.2609.02087 extended the MOFAT framework to incorporate nominal and discrete numeric factors. The resulting designs are implemented in the mofatQQ function. Yu and Joseph (2026) doi:10.48550/arXiv.2606.02533 proposed Space-Filling One-Factor-at-a-Time (SOFT) designs, which achieve optimal space-filling properties while retaining the screening capability. Therefore, SOFT designs are useful for both factor screening and surrogate modeling. The work on MOFAT is supported by a U.S. National Science Foundation (NSF) grant CMMI-1921646 `https://www.nsf.gov/awardsearch/show-award/?AWD_ID=1921646`, and the work on MOFATQQ and SOFT is supported by a U.S. NSF grant DMS-2310637.

Author(s)

Maintainer: Difan Song <dfsong96@gmail.com>

Authors:

- Difan Song <dfsong96@gmail.com>
- Wei-Yang Yu <wyu322@gatech.edu>
- Qian Xiao <qian.xiao@sjtu.edu.cn>
- V. Roshan Joseph <roshan@gatech.edu>

---

a_soft *Augmented SOFT (A-SOFT) Designs*

Description

Construct an augmented SOFT (A-SOFT) design by combining a SOFT component with sequentially selected unrestricted space-filling points. The resulting design has the form

$$D = \begin{pmatrix} D_{\text{soft}} \\ D_{\text{aug}} \end{pmatrix},$$

where D_{soft} is the initial SOFT design and D_{aug} contains the augmented points.

Usage

```
a_soft(
  p,
  l,
  structure = "standard",
  n_aug,
  m = 500L,
  N = 10000L,
  backend = "auto",
  ofat_design = NULL
)
```

Arguments

p	number of factors; an integer of at least 2. Not required when ofat_design is supplied; inferred as ncol(ofat_design), overriding any supplied value.
l	even number of base runs; an integer of at least 2. Not required when ofat_design is supplied; inferred as nrow(ofat_design) / (p + 1), overriding any supplied value. Also determines the augmentation kernel scale.
structure	indicating the OFAT structure. Choose from "standard" or "strict". The default is "standard". Ignored when ofat_design is supplied.
n_aug	number of points to augment.

`m`	number of Sobol evaluation points for initial SOFT optimization. The default is 500. Ignored when `ofat_design` is supplied.
`N`	size of the Sobol candidate pool for the sequential augmentation. The default is 10000.
`backend`	one of `"auto"`, `"cpu"`, `"cuda"`, or `"metal"`. Invalid, unavailable, or failed GPU backends silently use CPU.
`ofat_design`	`NULL`, or a nonempty numeric matrix with `p` columns, `l * (p + 1)` runs for an even integer `l`, and values in `[0, 1]`. When supplied, `p` and `l` are inferred from its dimensions, and this matrix is augmented directly instead of generating an initial SOFT design. Its range does not affect the Sobol candidate bounds.

Details

When `ofat_design = NULL`, the SOFT component contains $n_{\text{soft}} = l(p+1)$ runs, and the final design contains $l(p+1) + n_{\text{aug}}$ runs. Alternatively, a user-supplied initial design can be provided through `ofat_design`, in which case the supplied design is held fixed and augmented directly.

The augmented points are selected sequentially from a Sobol candidate pool by maximizing the Q criterion. The augmented points themselves are not constrained to follow the OFAT structure. CPU and optional GPU computation are supported. See Yu and Joseph (2026) for further details.

The initial SOFT design and the sequential augmentation use different kernel scales for optimizing the Q criterion. For the initial SOFT optimization, the kernel parameters are fixed internally at $\alpha = 1$ and $\beta = (2l)^{-2}$. During sequential augmentation, they are fixed at $\alpha = 1$ and $\beta = (2l + n_{\text{aug}})^{-2}$.

Regardless of whether `ofat_design` is supplied, candidate bounds for the augmentation are fixed at $[0.25/(2l + n_{\text{aug}}), 1 - 0.25/(2l + n_{\text{aug}})]$.

Value

A list with the following components:

`design`	The design matrix containing the generated or supplied initial design followed by the sequentially augmented points.
`backend_used`	The computational backend actually used.
`device`	The CPU or GPU device used for computation.
`fallback_reason`	The reason for falling back to the CPU backend, or `NULL` if no fallback occurred.

Author(s)

Wei-Yang Yu and V. Roshan Joseph

References

Yu, W.-Y. and Joseph, V. R. (2026). Space-Filling One-Factor-At-A-Time Designs. doi:10.48550/arXiv.2606.02533

See Also

`soft`, `Q`, `qmc_generate`

Examples

```
result <- a_soft(p = 2, l = 4, n_aug = 8)
result$design

existing <- mofat(p = 2, l = 4, method = "projection")
extended <- a_soft(n_aug = 8, ofat_design = existing)
extended$design
```

---

measure *Screening Measures*

Description

This function can be used for computing screening measures.

Usage

```
measure(design, y)
```

Arguments

design	design matrix, which should have the Sobol' design structure
y	response vector

Details

The measure function computes the screening measures such as the total Sobol' indices (Sobol' 1993) and μ^* measure of Campolongo et al. (2007). The design matrix should have the Sobol' design structure. Please see Xiao et al. (2022) and Yu and Joseph (2026) for further details.

Value

mustar	μ^* measure
t	Total Sobol' index

Author(s)

Wei-Yang Yu, Difan Song, Qian Xiao, and V. Roshan Joseph

References

Sobol', I. M. (1993). On Sensitivity Estimation for Nonlinear Mathematical Models. Mathematical Modeling and Computational Experiments, 1, 407–414.

Campolongo, F., Cariboni, J., and Saltelli, A. (2007). An Effective Screening Design for Sensitivity Analysis of Large Models. Environmental Modelling and Software, 22, 1509–1518.

Xiao, Q., Joseph, V. R., and Ray, D. M. (2022). Maximum One-Factor-At-A-Time Designs for Screening in Computer Experiments. Technometrics, 65, 261–270.

Yu, W.-Y. and Joseph, V. R. (2026). Space-Filling One-Factor-At-A-Time Designs. doi:10.48550/arXiv.2606.02533

See Also

mofat, mofatQQ, soft

Examples

```
#Friedman function
fun <- function (X)
{
 Y <- 10*sin(pi*X[1]*X[2]) + 20*(X[3] - 0.5)^2 + 10*X[4] + 5*X[5]
 return(Y)
}
design = mofat(p=10, l=3)
y = apply(design, 1, fun)

#Screening measures
measure(design, y)
```

---

mofat *MOFAT*

Description

This function can be used for generating MOFAT designs.

Usage

```
mofat(p, l, method = "best", nstart = 1, init = "MaxPro")
```

Arguments

p	number of factors
l	number of base runs
method	choose among "uniform", "projection", and "best"
nstart	number of random starts used when l > 3
init	initialization method for the base matrix when l > 3; choose among "rLHD", "MaxPro", and "MmLHD"

Details

The mofat function generates the MOFAT design for a given number of factors ($p \geq 2$) and number of base runs ($l \geq 3$). The total number of runs in the MOFAT design will be $l(p+1)$. A MOFAT design can be viewed as an optimized version of Morris screening design (Morris 1991) by exploiting its connections with the Monte Carlo-based design of Sobol' (1993). Please see Xiao et al. (2022) for details.

Three choices for the method are given: "uniform", "projection", and "best". Option "uniform" gives l equally-spaced levels for the entire design, which are also balanced. "projection" option adjusts the levels of the two base matrices A and B such that there are $2l$ or $2l-1$ levels in the design depending on l is even or odd. Option "best" (default) chooses the best among the first two options using maximin distance criterion.

Value

`design` MOFAT design

Author(s)

Qian Xiao and V. Roshan Joseph

References

Morris, M. D. (1991), "Factorial sampling plans for preliminary computational experiments," Technometrics, 33, 161–174.

Sobol', I. M. (1993), "On sensitivity estimation for nonlinear mathematical models," Mathematical Modeling and Computational Experiments, 1, 407–414.

Xiao, Q., Joseph, V. R., and Ray, D. M. (2022). "Maximum One-Factor-At-A-Time Designs for Screening in Computer Experiments". Technometrics, 65, 261–270.

Examples

```
#MOFAT with three base runs
mofat(p=10, l=3, method="uniform")
mofat(p=10, l=3, method="projection")

#MOFAT with five base runs
mofat(p=10,l=5)
dim(mofat(p=125,l=5))
```

---

mofat_backend_info *Inspect MOFAT Computational Backends*

Description

Report whether the CPU, Metal, and CUDA backends were compiled and whether they are usable in the current R process. CUDA probing includes helper ABI validation and a small runtime self-test. Results are cached for the R session unless `refresh = TRUE` is requested.

Usage

```
mofat_backend_info(refresh = FALSE)
```

Arguments

`refresh` logical scalar. If `TRUE`, discard the cached CUDA probe and repeat helper loading, device discovery, and the self-test.

Value

A named list with `cpu`, `metal`, and `cuda` components. Every component contains `compiled`, `available`, `device`, and `reason`. CUDA additionally reports helper, toolkit, runtime, driver, and compute-capability details when available.

Examples

```
info <- mofat_backend_info()
vapply(info, function(x) isTRUE(x$available), logical(1))
info$cuda$reason
```

`mofatQQ` *MOFAT Designs for Quantitative and Qualitative Factors*

Description

This function generates MOFAT-based designs for continuous, discrete numeric, and nominal factors.

Usage

```
mofatQQ(
  l,
  p_continuous = 0,
  b_continuous = NULL,
  p_disnum = 0,
  l_disnum = NULL,
  p_nom = 0,
  l_nom = NULL,
  nstart = 1,
  ...
)
```

Arguments

`l`	number of base runs
`p_continuous`	number of continuous factors
`b_continuous`	optional list of length `p_continuous`; each element should be a length-two numeric vector giving the lower and upper bounds used to rescale a continuous factor
`p_disnum`	number of discrete numeric factors
`l_disnum`	specification of the discrete numeric levels. Supply either a numeric vector giving the number of levels for each factor, or a list whose jth element contains the numeric levels for factor j
`p_nom`	number of nominal factors
`l_nom`	specification of the nominal levels. Supply either a numeric vector giving the number of levels for each nominal factor, or a list whose element lengths define the number of nominal levels. The returned design always encodes nominal factors as integer level labels
`nstart`	number of candidate mixed-factor designs to generate
`...`	additional arguments passed to `mofat`, such as `method` or `init`

Details

The `mofatQQ` function first generates a base MOFAT design with `mofat`, then transforms the columns according to factor type. The total number of factors must be at least two. Continuous factors are kept on the unit scale or rescaled to user-supplied bounds. Discrete numeric factors are mapped either to integer levels or to user-supplied numeric levels. Nominal factors are encoded by integer level labels, and the nominal columns are placed to the right of the numeric columns in the returned design. Each discrete numeric and nominal factor must have at least two levels.

When `nstart > 1`, the function generates multiple candidate mixed designs and selects the one with the smallest projected space-filling criterion. In that criterion, continuous and discrete numeric distances are measured on a rescaled numeric scale, while nominal differences are treated as binary mismatches.

Value

A design matrix with columns ordered as continuous, discrete numeric, and nominal factors. Nominal factors are returned as integer level labels.

Author(s)

Difan Song and V. Roshan Joseph

References

Xiao, Q., Joseph, V. R., and Ray, D. M. (2022). Maximum One-Factor-At-A-Time Designs for Screening in Computer Experiments. Technometrics, 65, 261–270.

Song, D., and Joseph, V. R. (2026). Efficient Screening Designs for Expensive Black-Box Models with Qualitative and Quantitative Factors. doi:10.48550/arXiv.2609.02087

See Also

`mofat`, `measure`

Examples

```
# Mixed continuous and nominal factors
D <- mofatQQ(
  l = 5,
  p_continuous = 2,
  b_continuous = list(c(-1, 1), c(0, 10)),
  p_nom = 1,
  l_nom = 3,
  nstart = 2,
  init = "rLHD"
)
dim(D)
```

`Q` *Q Criterion*

Description

Evaluate the Q criterion for a given design. This criterion measures the design's space-filling performance and serves as the objective function used to construct SOFT designs. See Yu and Joseph (2026) for further details.

Usage

```
Q(alpha, beta, design, qmc, backend = "auto")
```

Arguments

`alpha`	positive MIM kernel exponent.
`beta`	positive MIM kernel scale.
`design`	the design matrix.
`qmc`	the evaluation set.
`backend`	one of `"auto"`, `"cpu"`, `"cuda"`, or `"metal"`. Invalid, unavailable, or failed GPU backends silently use CPU.

Details

The Q criterion for a design $\mathbf{D} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^T$ is defined as

$$\min_{\mathbf{x} \in \mathbf{E}} \mathbf{r}(\mathbf{x})^T \mathbf{R}^{-1} \mathbf{r}(\mathbf{x}),$$

where $\mathbf{E}$ is the evaluation set provided by `qmc`. Here, $\mathbf{r}(\mathbf{x}) = [R(\mathbf{x}_1, \mathbf{x}), \ldots, R(\mathbf{x}_n, \mathbf{x})]^T$ and $\mathbf{R}_{i,j} = R(\mathbf{x}_i, \mathbf{x}_j)$, where

$$R(\mathbf{x}, \mathbf{x}') = \prod_{j=1}^{p} \left\{ 1 + \frac{(x_j - x'_j)^2}{\beta} \right\}^{-\alpha}.$$

Larger values of Q indicate better space-filling performance.

Value

A list with the following components:

`Q`	the Q criterion of the design.
`normalized_Q`	the normalized criterion defined as $Q^{1/p}$.
`backend_used`	The computational backend actually used.

Author(s)

Wei-Yang Yu and V. Roshan Joseph

References

Yu, W.-Y. and Joseph, V. R. (2026). Space-Filling One-Factor-At-A-Time Designs. doi:10.48550/arXiv.2606.02533

See Also

qmc_generate

Examples

```
design <- soft(p = 2, l = 4)$design
qmc <- qmc_generate(n = 1000, p = 2)
Q(1, 1/(2*4)^2, design, qmc)
```

---

qmc_generate *Evaluation Set for the Q Criterion*

Description

Generate Sobol points and augment them with corner points to evaluate the Q criterion. Duplicate points are removed from the resulting evaluation set. See Yu and Joseph (2026) for further details.

Usage

```
qmc_generate(n, p, threshold = 10)
```

Arguments

n	number of Sobol points.
p	number of factors.
threshold	dimension cutoff for corner augmentation. When p <= threshold, all 2^p binary corners are added. Otherwise, representative corners derived from the Sobol points are added.

Value

A matrix with p columns containing the unique Sobol and corner evaluation points. The number of rows may differ from n because corner points are added and duplicate rows are removed.

Author(s)

Wei-Yang Yu and V. Roshan Joseph

References

Yu, W.-Y. and Joseph, V. R. (2026). Space-Filling One-Factor-At-A-Time Designs. doi:10.48550/arXiv.2606.02533

See Also

Q

Examples

```
qmc <- qmc_generate(n = 1000, p = 2)
```

`soft` *SOFT Designs*

Description

Generate a SOFT design under either the standard or strict OFAT structure. For p factors and l base runs, the resulting design contains $l(p+1)$ runs. The design is constructed by maximizing the Q criterion under either the standard or strict OFAT structure. Computation can use an available GPU backend, with CPU execution used when GPU acceleration is unavailable. See Yu and Joseph (2026) for further details.

Usage

```
soft(p, l, structure = "standard", m = 500L, backend = "auto")
```

Arguments

`p`	number of factors; an integer of at least 2.
`l`	even number of base runs; an integer of at least 2.
`structure`	indicating the OFAT structure. Choose from `"standard"` or `"strict"`. The default is `"standard"`.
`m`	number of Sobol evaluation points generated before vertex augmentation. The default is 500.
`backend`	one of `"auto"`, `"cpu"`, `"cuda"`, or `"metal"`. Invalid, unavailable, or failed GPU backends silently use CPU.

Details

When optimizing Q, we set $\alpha = 1$ and $\beta = (2 * l)^{-2}$ internally.

Value

A list with the following components:

`design`	The SOFT design matrix.
`Q`	The Q criterion of the design.
`normalized_Q`	The normalized criterion defined as $Q^{1/p}$.
`qmc_size`	The number of unique evaluation points after adding corner points and removing duplicates. This value may differ from the input `m`.

backend_used	The computational backend actually used.
device	The CPU or GPU device used for computation.
fallback_reason	The reason for falling back to the CPU backend, or NULL if no fallback occurred.

Author(s)

Wei-Yang Yu and V. Roshan Joseph

References

Yu, W.-Y. and Joseph, V. R. (2026). Space-Filling One-Factor-At-A-Time Designs. doi:10.48550/arXiv.2606.02533

See Also

Q, qmc_generate

Examples

```
result <- soft(p = 2, l = 4)
result$design
```

Index

a_soft, 3

measure, 5, *9*
MOFAT *(*MOFAT-package*)*, 2
mofat, 6, *6*, *8*, *9*
MOFAT-package, 2
mofat_backend_info, 7
mofatQQ, *6*, 8

Q, *4*, 10, *12*, *13*
qmc_generate, *4*, 11, *11*, *13*

soft, *4*, *6*, 12