Package: SKBD (via r-universe)

June 8, 2026

Title Shared Keyboard Designs for Phase I Clinical Trials

Version 0.1.1.9000

Maintainer Jiangyan Zhao <zhaojy2017@126.com>

Description Implements the shared keyboard design (SKBD) for model-assisted phase I dose-finding, including decision-boundary construction, operating-characteristic simulation, and extensions for dose insertion and time-to-event settings. The package also provides an interactive Shiny interface for trial-planning workflows. For more details, see Zhao, Shi, and Xu (2026) <doi:10.48550/arXiv.2605.25043>.

License GPL (>= 3)

URL https://github.com/Jiangyan-Zhao/SKBD

BugReports https://github.com/Jiangyan-Zhao/SKBD/issues

Encoding UTF-8

NeedsCompilation no

Roxygen list(markdown = TRUE)

Depends R (>= 3.5.0)

Imports shiny

Suggests DT, testthat (>= 3.0.0)

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

Config/pak/sysreqs cmake make libuv1-dev zlib1g-dev

Repository https://jiangyan-zhao.r-universe.dev

Date/Publication 2026-06-08 03:11:25 UTC

RemoteUrl https://github.com/jiangyan-zhao/skbd

RemoteRef HEAD

RemoteSha c144512f41c9bbac65a9bcef5ad42e4fc85b88a7

Contents

SKBD-package *Shared Keyboard Designs for Phase I Dose-Finding Trials*

Description

The **SKBD** package provides tools for implementing the Shared Keyboard Design (SKBD) for model-assisted phase I dose-finding trials. SKBD extends the Keyboard design by incorporating kernel-weighted information sharing across dose levels while preserving the transparent key-based dose-escalation and de-escalation structure.

The package supports decision-boundary construction, operating-characteristic simulation, adaptive dose insertion, and time-to-event extensions for late-onset toxicity outcomes. It also provides utilities for generating random monotone dose-toxicity scenarios and an interactive Shiny interface for trial-planning workflows.

Main Functions

Core functionality is organized as follows:

`get_boundary_SKBD` Generate pre-tabulated dose-escalation, de-escalation, and overdose elimination boundaries for the Shared Keyboard Design, conditional on accumulated trial data across dose levels.

`get_OC_SKBD` Simulate operating characteristics for the standard SKBD under fixed dose-toxicity scenarios, including selection accuracy, patient allocation, overdose risk, and monotonicity diagnostics.

`get_OC_TITE_SKBD` Simulate operating characteristics for the time-to-event extension of SKBD, which accommodates delayed toxicity outcomes through weighted partial follow-up information.

`get_OC_Insert_SKBD` Simulate operating characteristics for the dose-insertion extension of SKBD, where the working dose grid can be adaptively refined during the trial.

`PUA` Generate random monotone dose-toxicity scenarios using a pseudo-uniform algorithm for simulation studies.

`run_SKBD_shiny` Launch an interactive Shiny application for generating SKBD decision tables and running operating-characteristic simulations.

References

Zhao J, Shi X, Xu J (2026). Shared Keyboard: An improved Bayesian design for phase I clinical trials via Beta kernel process. *ArXiv*. https://arxiv.org/abs/2605.25043

Yan F, Mandrekar SJ, Yuan Y (2017). Keyboard: A Novel Bayesian Toxicity Probability Interval Design for Phase I Clinical Trials. *Clinical Cancer Research*, 23(15), 3994–4003.

Lin R, Yuan Y (2020). Time-to-event model-assisted designs for dose-finding trials with delayed toxicity. *Biostatistics*, 21(4), 807–824.

Chu Y, Pan H, Yuan Y (2016). Adaptive dose modification for phase I clinical trials. *Statistics in Medicine*, 35(20), 3497–3508.

Clertant M, O'Quigley J (2017). Semiparametric dose finding methods. *Journal of the Royal Statistical Society: Series B*, 79(5), 1487–1508.

`get_boundary_SKBD` *Pre-tabulated Decision Boundaries for the Shared Keyboard Design*

Description

Generate the pre-tabulated dose escalation/de-escalation (and overdose elimination) decision tables for the Shared Keyboard Design (SKBD) at a **current dose** `d`, given the accumulated data across all doses. The output is a keyboard-like boundary table (similar to the original Keyboard design), but the decisions are based on *borrowed information* through a kernel-weighted Beta posterior at the current dose.

Usage

```
get_boundary_SKBD(
  target_prob,
  d,
  y,
  n,
  n_cohort = 10,
  cohort_size = 3,
  k_left = 0.2,
  k_right = 0.8,
  ref_gap = NULL,
  shared = TRUE,
  dose_set = 1:length(y),
  n_earlystop = 1000,
  margin_left = 0.05,
  margin_right = 0.05,
  cutoff_elimin = 0.95,
  extra_safe = FALSE,
  offset = 0.05,
  table_type = c("baseline", "continue"),
  show_past = TRUE,
  start_from = c("current", "next")
)
```

Arguments

target_prob	Target toxicity (DLT) probability ϕ used to define the target key.
d	Integer index of the current dose level (between 1 and length(y)).
y	Integer vector of length J giving the number of observed DLTs at each dose.
n	Integer vector of length J giving the number of treated patients at each dose.
n_cohort	Total number of cohorts in the trial (used to define the maximal table size).
cohort_size	Number of patients per cohort (used for cohort-aligned boundary_tab output).
k_left	Numeric scalar in $[0, 1]$. Left-side neighbor borrowing strength passed to kernel_fun().
k_right	Numeric scalar in $[0, 1]$. Right-side neighbor borrowing strength passed to kernel_fun() (set k_left == k_right for symmetric borrowing).
ref_gap	Optional positive scalar. Reference spacing passed to kernel_fun(). If NULL, kernel defaults to the minimum adjacent spacing in dose_set.
shared	Logical; if TRUE, use kernel-weighted borrowing across doses. If FALSE, use only current-dose data (identity kernel; no borrowing).
dose_set	Numeric vector of dose labels (default 1:length(y)). Used only to compute kernel distances.
n_earlystop	Maximum number of patients to display in output tables (columns are truncated at this value).
margin_left	Left margin of the target key (lower bound is target_prob - margin_left).
margin_right	Right margin of the target key (upper bound is target_prob + margin_right).
cutoff_elimin	Overdose elimination cutoff; eliminate if $\Pr(p_d > \phi \mid \text{data}) >$ this value.
extra_safe	Logical; if TRUE and d==1, return an additional conservative stopping boundary (similar to the BOIN extra-safe rule).
offset	Nonnegative offset in (0, 0.5); used only when extra_safe=TRUE to tighten the stopping rule.
table_type	Character; either "baseline" or "continue" (see Details).
show_past	Logical; if FALSE and table_type="continue", hide columns before the current treated count n[d] (or n[d]+1 depending on start_from).
start_from	Character; only relevant when show_past=FALSE and table_type="continue". "current" starts from column n[d]; "next" starts from column n[d]+1.

Details

The SKBD uses a kernel function to borrow information across dose levels. At the current dose d, define kernel weights $w_s(d)$ for each dose s. Let y_s and n_s be the observed number of DLTs and the number treated at dose s. The borrowed (effective) Beta posterior parameters are

$$\alpha_{\text{post}} = \alpha_0 + \sum_s w_s(d)\, y_s, \qquad \beta_{\text{post}} = \beta_0 + \sum_s w_s(d)\, (n_s - y_s),$$

where $(\alpha_0, \beta_0) = (1, 1)$ by default (non-informative prior).

Given $(\alpha_{\text{post}}, \beta_{\text{post}})$, the strongest keyboard interval ("strongest key") is the interval with the largest posterior probability mass under Beta$(\alpha_{\text{post}}, \beta_{\text{post}})$. The action is then determined by comparing the strongest key to the target key:

- `"E"`: escalate if the strongest key is to the left of the target key;
- `"S"`: stay if the strongest key is the target key;
- `"D"`: de-escalate if the strongest key is to the right of the target key.

For patient safety, an overdose control rule is applied: if at least 3 patients have been treated at the current dose and $\Pr(p_d > \phi \mid \text{data}) >$ cutoff_elimin, the current dose is eliminated and labeled as `"DU"` in the decision table.

Two table modes. This function supports two ways to construct a keyboard-like table:

- `table_type = "baseline"`: ignore the current dose's existing history (`n[d]`, `y[d]`) when tabulating; only other doses contribute as fixed borrowed information. This is useful for creating a "baseline" conditional table given other-dose information.
- `table_type = "continue"`: condition on the current dose's existing history (`n[d]`, `y[d]`) and tabulate decisions *from now on*. Only $n \geq n[d]$ columns are relevant, and (optionally) past columns can be hidden via `show_past = FALSE`.

Value

A list with components:

- `boundary_tab`: a 4-row boundary table (cohort-aligned columns) containing escalation, de-escalation, and elimination boundaries.
- `full_boundary_tab`: the full 4-row boundary table (possibly truncated/hiding past columns).
- `decision_table`: the underlying decision table with entries `"E"`, `"S"`, `"D"`, `"DU"`.
- `weight`: the normalized kernel weights used for borrowing at dose `d`.
- `meta`: a list recording `table_type`, `show_past`, `start_from`, and the starting column used.

If `extra_safe=TRUE` and `d==1`, additional elements `cutoff` and `stop_boundary` are returned.

References

Zhao J, Shi X, Xu J (2026). Shared Keyboard: An improved Bayesian design for phase I clinical trials via Beta kernel process. *ArXiv*. https://arxiv.org/abs/2605.25043

Examples

```
## Example data across 5 doses:
y <- c(0, 1, 2, 2, 0)
n <- c(3, 6, 9, 3, 0)
d <- 3
target_prob <- 0.3

## 1) Baseline conditional table:
##    Ignore (n[d], y[d]) at the current dose when tabulating
out_baseline <- get_boundary_SKBD(target_prob = target_prob,
                                  d = d, y = y, n = n,
                                  table_type = "baseline")
out_baseline$full_boundary_tab
```

```
## 2) Continue table:
##    Condition on (n[d], y[d]) and tabulate decisions from now on
out_continue <- get_boundary_SKBD(target_prob = target_prob,
                                  d = d, y = y, n = n,
                                  table_type = "continue")
out_continue$full_boundary_tab

## 3) Continue table (display only future columns):
out_future <- get_boundary_SKBD(target_prob = target_prob,
                                d = d, y = y, n = n,
                                table_type = "continue",
                                show_past = FALSE,
                                start_from = "next")
out_future$full_boundary_tab
```

`get_OC_Insert_SKBD` *Operating Characteristics for the Dose-Insertion Shared Keyboard Design*

Description

Simulate phase I dose-finding trials under the dose-insertion Shared Keyboard design (Insert-SKBD), where the working dose grid can be adaptively refined by inserting new dose levels when posterior evidence suggests that the target toxicity level is not well covered by the current prespecified grid.

Usage

```
get_OC_Insert_SKBD(
  target_prob,
  tox_prob,
  dose_set,
  n_cohort = 10,
  cohort_size = 3,
  C1 = 0.6,
  C2 = 0.6,
  start_dose = 1,
  margin_left = 0.05,
  margin_right = 0.05,
  n_earlystop = 1000,
  cutoff_elimin = 0.95,
  extra_safe = FALSE,
  offset = 0.05,
  k_left = 0.2,
  k_right = 0.8,
  ref_gap = NULL,
  light_return = TRUE,
  n_trial = 1000,
```

```
  seed = 6
)
```

Arguments

target_prob	Scalar in $(0, 1)$. The target toxicity rate ϕ.
tox_prob	Numeric vector in $[0, 1]$. True DLT probabilities at prespecified doses. Must have the same length as dose_set.
dose_set	Numeric vector of prespecified doses on the original (clinical) scale. Must be strictly increasing with no duplicates.
n_cohort	Positive integer. Number of cohorts per simulated trial.
cohort_size	Positive integer. Number of patients per cohort.
C1, C2	Scalars in $(0, 1)$. Posterior-evidence thresholds for triggering insertion. Larger values make insertion more conservative.
start_dose	Positive integer. Starting dose index (in 1:length(dose_set)).
margin_left, margin_right	Nonnegative scalars. Define the target key $(\phi-\epsilon_1, \phi+\epsilon_2)$ with key_L = target_prob - margin_left and key_U = target_prob + margin_right.
n_earlystop	Positive integer. Maximum number of patients allowed at a single dose before stopping accrual at that dose (operational cap).
cutoff_elimin	Scalar in $(0, 1)$. Overdose elimination cutoff: a dose is eliminated if $Pr(\pi(d) > \phi \mid \mathcal{D})$ exceeds cutoff_elimin (or cutoff_elimin - offset when extra_safe = TRUE).
extra_safe	Logical. If TRUE, use a more conservative elimination cutoff cutoff_elimin - offset.
offset	Scalar in $[0, 0.5)$. Safety offset used when extra_safe = TRUE.
k_left, k_right	Scalars in $[0, 1]$ controlling borrowing strength to the left and right neighbors in the SKBD pseudo-posterior update. Typically k_right > k_left for safety.
ref_gap	Optional positive scalar. Reference gap used for kernel scaling. Note: in the current implementation, ref_gap is recomputed internally as min(diff(dose_set_work)) after each insertion/update.
light_return	Logical. If TRUE, return only summaries; if FALSE, also return per-trial details in trial_detail.
n_trial	Positive integer. Number of simulated trials.
seed	Integer. RNG seed for reproducibility.

Details

The prespecified dose set dose_set is internally standardized to $[0, 1]$ for kernel construction and numerical stability. At each interim, Insert-SKBD:

1. updates Beta pseudo-posteriors across all current dose levels via kernel borrowing;
2. checks whether dose insertion is warranted near the current dose based on posterior-evidence thresholds C1 and C2;

3. if insertion occurs, augments the working grid and continues cohort-wise treating;
4. applies overdose elimination and a keyboard-style move rule for escalation/de-escalation.

The final selected dose is the admissible dose whose isotonic-smoothed posterior mean toxicity is closest to `target_prob`. If the trial stops early for safety, no dose is selected.

Value

A list with components:

`sel_pct_prespec` Numeric vector. Selection percentage (%) at each prespecified dose.

`pts_pct_prespec` Numeric vector. Patient allocation percentage (%) at each prespecified dose.

`insertion` A list summarizing inserted-dose behavior: `sel_pct`, `pts_pct`, `dose_mean`, `dose_sd`, `trial_pct`, `cohort_mean`, `n_median`.

`simdata` A data frame storing per-trial, per-dose counts: `Simulation`, `Dose`, `N` (treated), `X` (DLTs), and `Selection`.

`sel_dose_idx` Integer vector of selected dose indices on the *working* grid for each trial; `-1` indicates early stop / no selection.

`sel_dose` Numeric vector of selected dose values on the original scale (NA if none).

`n_insertions` Integer vector. Number of insertions performed in each trial.

`insert_at_cohort` Integer vector. Cohort index when the first insertion occurs (`0` if no insertion).

`trial_detail` (Only if `light_return = FALSE`) A list of length `n_trial` with per-trial working grids, counts, elimination flags, insertion info, and early-stop indicator.

References

Zhao J, Shi X, Xu J (2026). Shared Keyboard: An improved Bayesian design for phase I clinical trials via Beta kernel process. *ArXiv*. https://arxiv.org/abs/2605.25043

Chu, Y., H. Pan, and Y. Yuan (2016). Adaptive dose modification for phase I clinical trials. *Statistics in Medicine 35*(20), 3497–3508.

Examples

```
tox_prob <- c(0.14, 0.45, 0.63, 0.74, 0.80)
dose_set <- c(5, 15, 25, 35, 45)
out <- get_OC_Insert_SKBD(
  target_prob = 0.30,
  tox_prob = tox_prob,
  dose_set = dose_set,
  n_trial = 100
)
out$insertion$trial_pct
```

get_OC_SKBD *Operating Characteristics for the Shared Keyboard Design*

Description

Simulate Phase I dose-finding trials under the shared keyboard design (SKBD) and summarize operating characteristics (e.g., PCS, PCA, overdosing risk). SKBD follows the keyboard decision rules (strongest key vs. target key) but constructs posterior toxicity at the current dose using a kernel-weighted Beta update (Beta Kernel Process-style borrowing) when `shared=TRUE`.

Usage

```
get_OC_SKBD(
  target_prob,
  tox_prob,
  n_cohort,
  cohort_size,
  dose_set = 1:length(tox_prob),
  start_dose = 1,
  margin_left = 0.05,
  margin_right = 0.05,
  n_earlystop = 1000,
  cutoff_elimin = 0.95,
  extra_safe = FALSE,
  offset = 0.05,
  k_left = 0.2,
  k_right = 0.8,
  ref_gap = NULL,
  shared = TRUE,
  light_return = TRUE,
  n_trial = 1000,
  seed = 6
)
```

Arguments

`target_prob`	A scalar in $(0, 1)$ giving the target DLT probability ϕ.
`tox_prob`	A numeric vector of length `n_dose` giving the true DLT probabilities at each prespecified dose level (used for simulation).
`n_cohort`	Integer; number of cohorts per simulated trial.
`cohort_size`	Integer; number of patients per cohort.
`dose_set`	Numeric vector of length `length(tox_prob)` giving dose locations (must be ordered increasingly). Used for dose standardization and kernel construction when `shared=TRUE`.
`start_dose`	Integer in `1:n_dose`; starting dose index for each trial.

margin_left	Nonnegative scalar; left tolerance ε_1 for the target key $(\phi - \varepsilon_1, \phi + \varepsilon_2)$.
margin_right	Nonnegative scalar; right tolerance ε_2 for the target key $(\phi - \varepsilon_1, \phi + \varepsilon_2)$.
n_earlystop	Integer; if the number treated at the current dose reaches n_earlystop, the trial loop stops (useful for preventing excessive sampling at one dose in simulations).
cutoff_elimin	Scalar in $(0, 1)$; overdose-control cutoff. A dose is eliminated if $Pr(\pi_j > \phi \mid \mathcal{D})$ exceeds this cutoff and n_j >= 3.
extra_safe	Logical; if TRUE, uses a more conservative elimination threshold cutoff_elimin - offset.
offset	Nonnegative scalar; safety offset used only when extra_safe=TRUE.
k_left	Numeric scalar in $[0, 1]$. Left-side neighbor borrowing strength passed to kernel_fun().
k_right	Numeric scalar in $[0, 1]$. Right-side neighbor borrowing strength passed to kernel_fun() (set k_left == k_right for symmetric borrowing).
ref_gap	Optional positive scalar. Reference spacing passed to kernel_fun(). If NULL, kernel defaults to the minimum adjacent spacing in dose_set.
shared	Logical; if TRUE, use kernel-weighted borrowing across doses to construct the posterior at the current dose. If FALSE, reduces to local updating (keyboard-style) via an identity kernel.
light_return	Logical; if TRUE, do not store/return individual-level dose/DLT paths to reduce memory usage. If FALSE, returns dose_Paths and DLT_Paths.
n_trial	Integer; number of simulated trials.
seed	Integer; random seed for reproducibility.

Details

In each simulated trial, cohorts are treated sequentially. After each cohort, the posterior toxicity probability at the current dose is updated:

- If shared=FALSE, the update uses only the data at the current dose.
- If shared=TRUE, the update uses normalized kernel weights over the set of doses with observed data, borrowing more from nearby doses (and optionally asymmetrically from higher vs. lower doses).

The design identifies the strongest key (toxicity interval with largest posterior probability) and applies the standard keyboard escalation/de-escalation rule relative to the target key. An overdose-control rule eliminates overly toxic doses and all higher doses. At the end of the trial, the MTD is selected from the admissible doses by applying isotonic regression, if needed, to the posterior mean toxicity estimates and choosing the dose whose adjusted estimate is closest to target_prob. When shared = TRUE, the final toxicity estimates are obtained from a weak-prior Beta posterior with symmetric kernel borrowing; when shared = FALSE, they reduce to within-dose weak-prior Beta estimates.

Value

A list with components:

- PCS: percent correct selection (in %).

- PCA: percent correct allocation (in %).
- above_MTD: percent treated above the MTD (in %).
- ROD60, ROD80: risk of overdosing (in %), defined as the percent of trials with >60% or >80% patients treated above the MTD.
- dose_select: length-n_trial vector of selected MTD indices; -1 indicates no MTD.
- select_percent: selection percentages for each dose and -1.
- n_patient_mean: mean number of patients per trial.
- n_DLT: mean number of DLTs per trial.
- monotonic_percent: among trials with an admissible MTD decision, the percent whose estimated toxicity profile is already monotone nondecreasing before isotonic regression.
- Y, N: n_trial x n_dose matrices of DLT counts and treated counts.
- dose_Paths, DLT_Paths: only returned when light_return=FALSE; n_trial x (n_cohort*cohort_size) matrices recording individual-level assignments and outcomes.

References

Zhao J, Shi X, Xu J (2026). Shared Keyboard: An improved Bayesian design for phase I clinical trials via Beta kernel process. *ArXiv*. https://arxiv.org/abs/2605.25043

Yan, F., Mandrekar, S. J., and Yuan, Y. (2017). Keyboard: A Novel Bayesian Toxicity Probability Interval Design for Phase I Clinical Trials. *Clinical Cancer Research 23*(15), 3994–4003.

Examples

```
res <- get_OC_SKBD(
  target_prob = 0.30,
  tox_prob = c(0.05, 0.12, 0.30, 0.45, 0.60),
  n_cohort = 10, cohort_size = 3
)
str(res)
```

---

get_OC_TITE_SKBD *Operating Characteristics for the TITE-SKBD Design*

Description

Simulate phase I dose-finding trials under the time-to-event shared keyboard design (TITE-SKBD), which accommodates late-onset toxicities and pending outcomes via weighted (partial) follow-up.

Usage

```
get_OC_TITE_SKBD(
  target_prob,
  tox_prob,
  n_cohort,
  cohort_size,
  k_left = 0.2,
  k_right = 0.8,
  ref_gap = NULL,
  shared = TRUE,
  dose_set = 1:length(tox_prob),
  n_earlystop = 1000,
  start_dose = 1,
  margin_left = 0.05,
  margin_right = 0.05,
  cutoff_elimin = 0.95,
  extra_safe = FALSE,
  offset = 0.05,
  tau = 3,
  accrual = 2,
  alpha = 0.5,
  piecewise = FALSE,
  prior_p = rep(1/3, 3),
  dist_DLT = c("weibull", "loglogistic", "uniform"),
  dist_enter = c("exp", "uniform"),
  light_return = TRUE,
  n_trial = 1000,
  seed = 6
)
```

Arguments

target_prob	Scalar in $(0, 1)$. Target toxicity rate ϕ.
tox_prob	Numeric vector in $[0, 1]$. True DLT probabilities at each dose level.
n_cohort	Positive integer. Number of cohorts per simulated trial.
cohort_size	Positive integer. Number of patients per cohort.
k_left, k_right	Scalars in $[0, 1]$ controlling borrowing strength to the left and right neighbors when shared = TRUE. Typically k_right > k_left for safety.
ref_gap	Optional positive scalar. Reference spacing passed to kernel_fun(). If NULL, kernel defaults to the minimum adjacent spacing in dose_set.
shared	Logical. If TRUE, use kernel borrowing (TITE-SKBD). If FALSE, use within-dose updates (keyboard-style).
dose_set	Numeric vector of dose values. Defaults to 1:length(tox_prob). Internally standardized to $[0, 1]$ for kernel construction.
n_earlystop	Positive integer. Operational cap on the number of treated patients at a single dose; if reached, accrual at that dose stops.

`start_dose` Positive integer. Starting dose index in `1:length(tox_prob)`.

`margin_left`, `margin_right` Nonnegative scalars. Define the target key $(\phi - \epsilon_1, \phi + \epsilon_2)$.

`cutoff_elimin` Scalar in $(0, 1)$. Overdose elimination cutoff for $Pr(\pi(d) > \phi \mid \mathcal{D})$.

`extra_safe` Logical. If `TRUE`, use a more conservative elimination cutoff `cutoff_elimin - offset`.

`offset` Scalar in $[0, 0.5)$. Safety offset used when `extra_safe = TRUE`.

`tau` Positive scalar. Length of the DLT assessment window (maximum follow-up time).

`accrual` Positive scalar. Patient accrual rate parameter (used as `rate` for exponential, or to set the range for uniform inter-arrival times).

`alpha` Positive scalar. Shape parameter used in `gen_tite()` for generating DLT times (interpretation depends on `dist_DLT`).

`piecewise` Logical. If `TRUE`, use a piecewise weighting function for pending outcomes over thirds of `tau`, with probabilities `prior_p`.

`prior_p` Numeric vector of length 3. Prior probabilities for the piecewise weighting segments; will be normalized to sum to 1 if needed.

`dist_DLT` Character. Distribution for time-to-DLT generation in `gen_tite()`: one of `"weibull"`, `"loglogistic"`, `"uniform"`.

`dist_enter` Character. Enrollment-time distribution: `"exp"` or `"uniform"`.

`light_return` Logical. If `TRUE`, do not store full patient-level paths for each trial. If `FALSE`, return `dose_Paths` and `DLT_Paths`.

`n_trial` Positive integer. Number of simulated trials.

`seed` Integer. RNG seed for reproducibility.

Details

This function implements a cohort-wise trial simulation with time-to-event toxicity outcomes. Patients accrue according to `dist_enter` at rate `accrual`, and each patient has a DLT time generated under `dist_DLT` over a maximum assessment window `tau`.

At each interim decision time, pending patients contribute fractional information through weights $\omega_i \in [0, 1]$ proportional to follow-up time, i.e., $\omega_i = u_i/\tau$ for the uniform hazard setting, with an optional piecewise extension controlled by `piecewise` and `prior_p`. The design borrows information across dose levels using a kernel-weighted Beta pseudo-posterior when `shared = TRUE`; otherwise it reduces to a keyboard-style Beta-binomial update using only within-dose data.

A safety suspension rule is enforced: dose escalation is not allowed until at least two patients at the current dose have completed the DLT assessment. Dose elimination is triggered when $Pr(\pi(d) > \phi \mid \mathcal{D})$ exceeds `cutoff_elimin` (or `cutoff_elimin - offset` when `extra_safe = TRUE`) and at least 3 patients at that dose have completed assessment.

The final MTD is selected from admissible doses (treated and not eliminated) as the dose whose (isotonic-adjusted, if needed) posterior mean toxicity is closest to `target_prob`.

Value

A list containing operating characteristics and trial-level summaries:

`PCS` Percentage of correct selection (%).

`PCA` Percentage of correct allocation (%).

`above_MTD` Percentage of patients treated above the MTD region (%).

`ROD60`, `ROD80` Risk of overdosing (%): proportion of trials where >60% (or >80%) of patients were treated above the target key.

`dose_select` Selected MTD index for each trial; `-1` indicates no selection (early stop).

`select_percent` Selection percentage by dose (including `-1` for no selection).

`n_patient_mean` Average number of patients per trial.

`n_DLT` Average number of DLTs per trial.

`duration_mean` Average trial duration (in the same time unit as `tau`).

`monotonic_percent` Percentage of trials where estimated toxicity means are monotone before isotonic adjustment.

`Y`, `N` Matrices (`n_trial` by `n_dose`) of DLT counts and treated counts.

`dose_Paths`, `DLT_Paths` (Only if `light_return = FALSE`) Patient-level dose assignment paths and DLT indicators for each trial.

References

Zhao J, Shi X, Xu J (2026). Shared Keyboard: An improved Bayesian design for phase I clinical trials via Beta kernel process. *ArXiv*. https://arxiv.org/abs/2605.25043

Lin, R. and Y. Yuan (2020). Time-to-event model-assisted designs for dose-finding trials with delayed toxicity. *Biostatistics 21*(4), 807–824.

Examples

```
tox_prob <- c(0.05, 0.12, 0.20, 0.35, 0.50)
out <- get_OC_TITE_SKBD(
  target_prob = 0.20,
  tox_prob = tox_prob,
  n_cohort = 10,
  cohort_size = 3,
  tau = 3,
  accrual = 2,
  dist_DLT = "weibull",
  dist_enter = "exp",
  n_trial = 200,
  seed = 1
)
out$PCS
```

PUA *Generate Random Monotone Toxicity Scenarios by the Pseudo-Uniform Algorithm*

Description

Generate random monotone dose-toxicity scenarios using a pseudo-uniform algorithm (PUA) for simulation studies in phase I dose-finding.

Usage

```
PUA(
  dose_set,
  target_prob,
  n_scenarios = 1000,
  margin_left = 0.05,
  margin_right = 0.05,
  seed = 6
)
```

Arguments

`dose_set`	Numeric vector of prespecified dose levels. Only its length is used by the current implementation.
`target_prob`	Scalar in $(0, 1)$ giving the target toxicity probability ϕ.
`n_scenarios`	Positive integer. Number of random toxicity scenarios to generate.
`margin_left`	Nonnegative scalar. Left margin ϵ_1 defining the target key $(\phi - \epsilon_1, \phi + \epsilon_2)$.
`margin_right`	Nonnegative scalar. Right margin ϵ_2 defining the target key $(\phi - \epsilon_1, \phi + \epsilon_2)$.
`seed`	Integer. Random seed for reproducibility.

Details

This function generates `n_scenarios` monotone toxicity curves over a prespecified dose set. For each scenario, it first randomly selects a dose level to serve as the target-dose location, then samples an upper bound for the toxicity range, and finally draws a sorted vector of toxicity probabilities from a uniform distribution on that range.

A generated scenario is accepted only if:

- the selected dose is the closest dose to the target toxicity rate `target_prob`, and
- the toxicity probability at that dose lies within the target key $(\phi - \epsilon_1, \phi + \epsilon_2)$, and
- the adjacent gaps around that dose, when applicable, are both greater than 0.05 and smaller than 0.3.

The resulting scenarios therefore have a unique and reasonably well-separated target dose, making them suitable for benchmarking dose-finding designs under random monotone settings.

Value

A numeric matrix with `n_scenarios` rows and `length(dose_set)` columns. Each row is an accepted monotone toxicity scenario, with entries giving the true toxicity probabilities at the corresponding dose levels.

References

Zhao J, Shi X, Xu J (2026). Shared Keyboard: An improved Bayesian design for phase I clinical trials via Beta kernel process. *ArXiv*. https://arxiv.org/abs/2605.25043

Clertant, M. and J. O'Quigley (2017). Semiparametric dose finding methods. *Journal of the Royal Statistical Society Series B: Statistical Methodology 79*(5), 1487–1508.

Zhou, Y., J. J. Lee, S. Wang, S. Bailey, and Y. Yuan (2021). Incorporating historical information to improve phase I clinical trials. *Pharmaceutical Statistics 20*(6), 1017–1034.

Examples

```
set.seed(1)
scen <- PUA(
  dose_set = 1:5,
  target_prob = 0.30,
  n_scenarios = 10
)
scen
```

`run_SKBD_shiny` *Launch an Interactive Shiny App for SKBD*

Description

`run_SKBD_shiny()` launches a Shiny interface for exploring key functions in the `SKBD` package, including decision-boundary generation and operating characteristic simulation.

Usage

```
run_SKBD_shiny(
  launch.browser = getOption("shiny.launch.browser", interactive()),
  ...
)
```

Arguments

`launch.browser`	Logical or function; controls whether and how the app is opened after launch. The default is `getOption("shiny.launch.browser", interactive())`, which follows the standard behavior of `shiny::runApp()`. In RStudio, this typically opens the app in the RStudio Viewer or Shiny window. Set `launch.browser = TRUE` to open the app in the system default browser, or `launch.browser = FALSE` to start the app without opening a browser.
`...`	Additional arguments passed to `shiny::runApp()`, such as `port`, `host`, or `display.mode`.

Details

The app currently provides two tabs:

- **Trial Setting:** interactively generate dose-escalation and de-escalation boundary tables.
- **Simulation:** run batch simulations under user-specified scenarios and summarize the operating characteristics.

By default, the app follows the standard behavior of `shiny::runApp()`. In RStudio, this typically opens the app in the RStudio Viewer or Shiny window; in other R sessions, it may open in the system default browser.

Value

Invisibly returns the value from `shiny::runApp()`.

Examples

```
if (interactive()) {
  run_SKBD_shiny()

  # Open in the system default browser
  run_SKBD_shiny(launch.browser = TRUE)

  # Start the app without opening a browser
  run_SKBD_shiny(launch.browser = FALSE)
}
```

Index