Package 'drasR'

November 6, 2026

Type Package

Title Drift-Aware Risk-Guaranteed Acceptance Sampling

Version 0.1.0

Description Implements drift-aware risk-guaranteed acceptance sampling methods for adaptive lot disposition. The package provides tools for classical acceptance sampling, producer-risk and consumer-risk assessment, exponentially weighted moving average drift monitoring, adaptive tightening policies, decision fragility analysis, and safety-cost Pareto frontier construction. Classical binomial acceptance-plan design is described in the NIST/SEMATECH e-Handbook of Statistical Methods <https://www.itl.nist.gov/div898/handbook/pmc/section2/pmc232.htm>. Exponentially weighted moving averages follow Roberts (1959) <doi:10.1080/00401706.1959.10489860>. The methods are intended for simulation studies, supplier quality monitoring, and audit-oriented inspection decision support.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.1.0)

Imports stats, graphics, utils

Suggests testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.1.1

URL https://github.com/zerish12/drasR

BugReports https://github.com/zerish12/drasR/issues

RoxygenNote 7.3.3

LazyData true

NeedsCompilation no

Author Muhammad Zahir Khan [aut, cre]

Maintainer Muhammad Zahir Khan <zahirstat007@gmail.com>

Repository CRAN

Date/Publication 2026-11-06 12:30:07 UTC

Contents

apply_fixed_plan . . . 2
consumer_risk . . . 3
decision_fragility . . . 4
dras_ewma . . . 5
dras_monitor . . . 6
evaluate_sampling . . . 7
example_lot_data . . . 8
find_plan . . . 8
pareto_frontier . . . 9
plot_dras_frontier . . . 10
prepare_lot_data . . . 11
prob_accept . . . 12
producer_risk . . . 12
run_dras_experiment . . . 13
select_dras_policy . . . 14
simulate_defect_process . . . 15
simulate_lots . . . 16
summarise_dras_monitor . . . 16
tune_dras_policy . . . 17
write_dras_report . . . 18

Index **20**

`apply_fixed_plan` *Apply a Fixed Acceptance Sampling Plan*

Description

Apply a Fixed Acceptance Sampling Plan

Usage

```
apply_fixed_plan(lots, n, c, aql = 0.01)
```

Arguments

`lots`	Data frame with column `p_true`.
`n`	Sample size.
`c`	Acceptance number.
`aql`	Acceptable quality level.

Value

Data frame with lot decisions.

Examples

```
plan <- find_plan(0.01, 0.05)
p <- simulate_defect_process(20)
lots <- simulate_lots(p)
apply_fixed_plan(lots, n = plan$n, c = plan$c)
```

`consumer_risk` *Consumer Risk*

Description

Computes consumer risk, defined as the probability of accepting a lot at the limiting quality level.

Usage

```
consumer_risk(n, c, ltpd)
```

Arguments

`n`	Sample size.
`c`	Acceptance number.
`ltpd`	Limiting quality level.

Value

Consumer risk.

Examples

```
consumer_risk(n = 132, c = 3, ltpd = 0.05)
```

`decision_fragility` *Acceptance Decision Fragility*

Description

Computes whether an accept-favouring decision changes under plausible uncertainty around an estimated defect probability.

Usage

```
decision_fragility(
  n,
  c,
  p_hat,
  uncertainty_width = 0.01,
  threshold = 0.5,
  grid_points = 101
)
```

Arguments

`n`	Sample size.
`c`	Acceptance number.
`p_hat`	Estimated defect probability.
`uncertainty_width`	Half-width of uncertainty interval.
`threshold`	Acceptance probability threshold.
`grid_points`	Number of grid points in uncertainty interval.

Value

Integer value: 1 if fragile, 0 if stable.

Examples

```
decision_fragility(n = 132, c = 3, p_hat = 0.03)
```

dras_ewma *Apply EWMA Drift-Aware Acceptance Sampling*

Description

Applies a drift-aware acceptance sampling policy using an EWMA statistic to trigger tightened inspection.

Usage

```
dras_ewma(
  lots,
  n0,
  c0,
  lambda = 0.2,
  warning_threshold = 0.03,
  tighten_factor = 1.5,
  aql = 0.01
)
```

Arguments

lots	Data frame with column p_true.
n0	Baseline sample size.
c0	Baseline acceptance number.
lambda	EWMA smoothing parameter.
warning_threshold	Drift warning threshold.
tighten_factor	Multiplicative factor applied to n0 under tightened inspection.
aql	Acceptable quality level.

Value

Data frame with adaptive lot decisions.

Examples

```
plan <- find_plan(0.01, 0.05)
p <- simulate_defect_process(50, "sudden_deterioration")
lots <- simulate_lots(p)
dras_ewma(lots, n0 = plan$n, c0 = plan$c)
```

`dras_monitor` *Monitor Real Lot Inspection Data Using DRAS*

Description

Applies an EWMA-based DRAS monitoring rule to real lot-level inspection data prepared by `prepare_lot_data()`. Unlike simulation functions, this function does not require the true defect probability.

Usage

```
dras_monitor(
  data,
  baseline_sample_size,
  acceptance_number,
  lambda = 0.2,
  warning_threshold = 0.03,
  tighten_factor = 1.5,
  initial_rate = 0.01
)
```

Arguments

`data`	A data frame returned by `prepare_lot_data()`.
`baseline_sample_size`	Baseline sample size under normal inspection.
`acceptance_number`	Acceptance number used for lot disposition.
`lambda`	EWMA smoothing parameter.
`warning_threshold`	EWMA warning threshold.
`tighten_factor`	Multiplicative tightening factor used when the EWMA statistic exceeds the warning threshold.
`initial_rate`	Initial EWMA value. Defaults to 0.01.

Value

A data frame with EWMA statistic, inspection mode, and recommended next sample size.

Examples

```
raw_data <- data.frame(
  lot_id = paste0("L", 1:6),
  sample_size = rep(100, 6),
  defects_found = c(1, 0, 1, 4, 5, 6)
)
```

```
prepared <- prepare_lot_data(
  raw_data,
  lot_id = "lot_id",
  sample_size = "sample_size",
  defects = "defects_found"
)

dras_monitor(prepared, baseline_sample_size = 132, acceptance_number = 3)
```

`evaluate_sampling` *Evaluate Lot-Disposition Results*

Description

Evaluates false acceptance, false rejection, inspection burden, and tightened-inspection frequency.

Usage

```
evaluate_sampling(results, method_name = "method")
```

Arguments

`results`	Data frame returned by `apply_fixed_plan()` or `dras_ewma()`.
`method_name`	Name of method.

Value

One-row data frame of performance metrics.

Examples

```
plan <- find_plan(0.01, 0.05)
p <- simulate_defect_process(20)
lots <- simulate_lots(p)
res <- apply_fixed_plan(lots, plan$n, plan$c)
evaluate_sampling(res, "Fixed ASP")
```

example_lot_data *Example Lot Inspection Data*

Description

A simulated lot-level inspection dataset for demonstrating drift-aware risk-guaranteed acceptance sampling. The data represent 200 production lots under a sudden deterioration scenario, where the true defect probability increases halfway through the sequence.

Usage

```
example_lot_data
```

Format

A data frame with 200 rows and 4 variables:

lot Lot index.

p_true True lot-level defect probability used in simulation.

lot_size Number of units in the lot.

true_defects Simulated number of true defective units in the lot.

Details

This dataset is intended for examples, tests, and vignettes. It is not a real industrial dataset. Users can apply the same functions to their own lot-level inspection data containing defect rates or defect counts.

Examples

```
data(example_lot_data)
head(example_lot_data)
```

find_plan *Find a Single Acceptance Sampling Plan*

Description

Finds the smallest sample size and acceptance number satisfying producer-risk and consumer-risk constraints.

Usage

```
find_plan(aql, ltpd, alpha = 0.05, beta = 0.1, max_n = 500)
```

Arguments

aql	Acceptable quality level.
ltpd	Limiting quality level.
alpha	Maximum producer risk.
beta	Maximum consumer risk.
max_n	Maximum sample size to search.

Value

A data frame with sample size, acceptance number, producer risk, and consumer risk.

Examples

```
find_plan(aql = 0.01, ltpd = 0.05, alpha = 0.05, beta = 0.10)
```

---

pareto_frontier *Compute a Safety-Cost Pareto Frontier*

Description

Identifies Pareto-efficient policies where lower inspection burden and lower false acceptance are preferred.

Usage

```
pareto_frontier(
  x,
  cost_col = "mean_avg_sample_number",
  safety_col = "mean_false_accepts"
)
```

Arguments

x	Data frame containing policy results.
cost_col	Column name for inspection burden.
safety_col	Column name for false acceptance burden.

Value

Data frame of Pareto-efficient rows.

Examples

```
df <- data.frame(cost = c(100, 120, 150), risk = c(5, 3, 2))
pareto_frontier(df, cost_col = "cost", safety_col = "risk")
```

`plot_dras_frontier` *Plot the DRAS Safety-Cost Frontier*

Description

Plots the safety-cost trade-off from a DRAS tuning table. The x-axis represents inspection burden and the y-axis represents false-acceptance burden.

Usage

```
plot_dras_frontier(
  tuning_results,
  selected_policies = NULL,
  cost_col = "mean_avg_sample_number",
  safety_col = "mean_false_accepts",
  main = "DRAS Safety-Cost Frontier"
)
```

Arguments

`tuning_results`	Data frame returned by `tune_dras_policy()`.
`selected_policies`	Optional data frame returned by `select_dras_policy()`.
`cost_col`	Column name for inspection burden.
`safety_col`	Column name for false-acceptance burden.
`main`	Plot title.

Value

Invisibly returns `tuning_results`.

Examples

```
tuned <- tune_dras_policy(
  n_reps = 2,
  n_lots = 30,
  scenario = "sudden_deterioration",
  lambda_values = c(0.20, 0.40),
  warning_values = c(0.03),
  tighten_values = c(1.50, 2.00)
)
selected <- select_dras_policy(tuned)
plot_dras_frontier(tuned, selected)
```

prepare_lot_data *Prepare Real Lot-Level Inspection Data*

Description

Converts real industrial or pharmaceutical inspection data into a standard format suitable for DRAS analysis.

Usage

```
prepare_lot_data(
  data,
  lot_id,
  sample_size,
  defects,
  lot_size = NULL,
  date = NULL,
  supplier = NULL
)
```

Arguments

data	A data frame containing lot-level inspection records.
lot_id	Name of the lot identifier column.
sample_size	Name of the inspected sample-size column.
defects	Name of the observed defect-count column.
lot_size	Optional name of the lot-size column.
date	Optional name of the date column.
supplier	Optional name of the supplier identifier column.

Value

A data frame in DRAS-ready format.

Examples

```
raw_data <- data.frame(
  lot_id = paste0("L", 1:5),
  date = as.Date("2026-01-01") + 0:4,
  supplier = c("A", "A", "A", "B", "B"),
  lot_size = rep(1000, 5),
  sample_size = rep(100, 5),
  defects_found = c(1, 0, 2, 5, 4)
)

prepare_lot_data(
```

```
  raw_data,
  lot_id = "lot_id",
  sample_size = "sample_size",
  defects = "defects_found",
  lot_size = "lot_size",
  date = "date",
  supplier = "supplier"
)
```

`prob_accept` *Probability of Accepting a Lot*

Description

Computes the probability of accepting a lot under a binomial acceptance sampling plan.

Usage

```
prob_accept(n, c, p)
```

Arguments

`n`	Sample size.
`c`	Acceptance number. The lot is accepted if observed defects are less than or equal to `c`.
`p`	Defect probability.

Value

Probability of lot acceptance.

Examples

```
prob_accept(n = 132, c = 3, p = 0.01)
```

`producer_risk` *Producer Risk*

Description

Computes producer risk, defined as the probability of rejecting a lot at the acceptable quality level.

Usage

```
producer_risk(n, c, aql)
```

Arguments

n	Sample size.
c	Acceptance number.
aql	Acceptable quality level.

Value

Producer risk.

Examples

```
producer_risk(n = 132, c = 3, aql = 0.01)
```

---

run_dras_experiment *Run a DRAS Monte Carlo Experiment*

Description

Runs repeated simulations comparing fixed acceptance sampling with EWMA-based drift-aware risk-guaranteed acceptance sampling.

Usage

```
run_dras_experiment(
  n_reps = 100,
  n_lots = 200,
  scenario = c("stable", "sudden_deterioration", "gradual_drift"),
  aql = 0.01,
  ltpd = 0.05,
  alpha = 0.05,
  beta = 0.1,
  lambda = 0.4,
  warning_threshold = 0.02,
  tighten_factor = 2,
  seed = NULL
)
```

Arguments

n_reps	Number of Monte Carlo replications.
n_lots	Number of lots per replication.
scenario	Defect-process scenario.
aql	Acceptable quality level.
ltpd	Limiting quality level.
alpha	Maximum producer risk.

beta	Maximum consumer risk.
lambda	EWMA smoothing parameter.
warning_threshold	EWMA warning threshold.
tighten_factor	Multiplicative tightening factor.
seed	Optional random seed.

Value

A data frame of summary results by method.

Examples

```
run_dras_experiment(n_reps = 10, scenario = "sudden_deterioration")
```

---

select_dras_policy *Select Low-Cost, Balanced, and High-Safety DRAS Policies*

Description

Selects three representative policies from a DRAS tuning table: a low-cost policy, a balanced policy, and a high-safety policy.

Usage

```
select_dras_policy(
  tuning_results,
  cost_col = "mean_avg_sample_number",
  safety_col = "mean_false_accepts"
)
```

Arguments

tuning_results	Data frame returned by tune_dras_policy().
cost_col	Column name for inspection burden.
safety_col	Column name for false-acceptance burden.

Value

A data frame containing three selected DRAS policies.

Examples

```
tuned <- tune_dras_policy(
  n_reps = 2,
  n_lots = 30,
  scenario = "sudden_deterioration",
  lambda_values = c(0.20, 0.40),
  warning_values = c(0.03),
  tighten_values = c(1.50, 2.00)
)
select_dras_policy(tuned)
```

---

simulate_defect_process

Simulate a Defect Process

Description

Simulates lot-level defect probabilities under stable, sudden deterioration, or gradual drift scenarios.

Usage

```
simulate_defect_process(
  n_lots = 200,
  scenario = c("stable", "sudden_deterioration", "gradual_drift"),
  p0 = 0.01,
  p1 = 0.06
)
```

Arguments

n_lots	Number of lots.
scenario	One of "stable", "sudden_deterioration", or "gradual_drift".
p0	Initial defect probability.
p1	Deteriorated defect probability.

Value

Numeric vector of true defect probabilities.

Examples

```
simulate_defect_process(10, scenario = "gradual_drift")
```

simulate_lots *Simulate Lot Inspection Data*

Description

Simulate Lot Inspection Data

Usage

```
simulate_lots(p_true, lot_size = 1000)
```

Arguments

p_true	Numeric vector of true defect probabilities.
lot_size	Lot size.

Value

Data frame of simulated lots.

Examples

```
p <- simulate_defect_process(20)
simulate_lots(p)
```

summarise_dras_monitor *Summarise DRAS Monitoring Results*

Description

Summarises the output from `dras_monitor()` to provide an audit-friendly overview of lot decisions, warnings, inspection modes, and defect rates.

Usage

```
summarise_dras_monitor(monitored_data, by_supplier = FALSE)
```

Arguments

monitored_data	A data frame returned by `dras_monitor()`.
by_supplier	Logical. If TRUE and supplier information is available, returns summaries by supplier.

Value

A data frame containing DRAS monitoring summary statistics.

Examples

```
raw_data <- data.frame(
  lot_id = paste0("L", 1:6),
  supplier = c("A", "A", "A", "B", "B", "B"),
  sample_size = rep(100, 6),
  defects_found = c(1, 0, 1, 4, 5, 6)
)

prepared <- prepare_lot_data(
  raw_data,
  lot_id = "lot_id",
  sample_size = "sample_size",
  defects = "defects_found",
  supplier = "supplier"
)

monitored <- dras_monitor(
  prepared,
  baseline_sample_size = 132,
  acceptance_number = 3
)

summarise_dras_monitor(monitored)
```

---

tune_dras_policy *Tune DRAS Policies*

Description

Searches over EWMA smoothing parameters, warning thresholds, and tightening factors to identify safety-cost trade-offs for DRAS.

Usage

```
tune_dras_policy(
  n_reps = 100,
  n_lots = 200,
  scenario = c("stable", "sudden_deterioration", "gradual_drift"),
  lambda_values = c(0.05, 0.1, 0.2, 0.3, 0.4),
  warning_values = c(0.02, 0.03, 0.04),
  tighten_values = c(1.25, 1.5, 2),
  aql = 0.01,
  ltpd = 0.05,
  alpha = 0.05,
```

```
  beta = 0.1,
  seed = NULL
)
```

Arguments

n_reps	Number of Monte Carlo replications per policy.
n_lots	Number of lots per replication.
scenario	Defect-process scenario.
lambda_values	Candidate EWMA smoothing values.
warning_values	Candidate warning thresholds.
tighten_values	Candidate tightening factors.
aql	Acceptable quality level.
ltpd	Limiting quality level.
alpha	Maximum producer risk.
beta	Maximum consumer risk.
seed	Optional random seed.

Value

A data frame of policy performance results.

Examples

```
tune_dras_policy(
  n_reps = 2,
  n_lots = 30,
  scenario = "sudden_deterioration",
  lambda_values = c(0.20, 0.40),
  warning_values = c(0.03),
  tighten_values = c(1.50)
)
```

---

write_dras_report *Write a DRAS Audit Report*

Description

Writes DRAS monitoring outputs and audit summaries to CSV files. This function is intended for industrial and quality-assurance workflows where lot-level monitoring decisions must be saved for review.

Usage

```
write_dras_report(monitored_data, output_dir, include_supplier_summary = TRUE)
```

Arguments

monitored_data	A data frame returned by dras_monitor().
output_dir	Required, non-empty directory path supplied by the user. No default is provided.
include_supplier_summary	Logical. If TRUE and supplier information is available, writes a supplier-level summary.

Value

Invisibly returns a character vector of written file paths.

Examples

```
raw_data <- data.frame(
  lot_id = paste0("L", 1:6),
  supplier = c("A", "A", "A", "B", "B", "B"),
  sample_size = rep(100, 6),
  defects_found = c(1, 0, 1, 4, 5, 6)
)

prepared <- prepare_lot_data(
  raw_data,
  lot_id = "lot_id",
  sample_size = "sample_size",
  defects = "defects_found",
  supplier = "supplier"
)

monitored <- dras_monitor(
  prepared,
  baseline_sample_size = 132,
  acceptance_number = 3
)

temp_dir <- tempfile("dras_report_")
write_dras_report(monitored, output_dir = temp_dir)
```

Index

∗ **datasets**
example_lot_data, 8

apply_fixed_plan, 2

consumer_risk, 3

decision_fragility, 4
dras_ewma, 5
dras_monitor, 6

evaluate_sampling, 7
example_lot_data, 8

find_plan, 8

pareto_frontier, 9
plot_dras_frontier, 10
prepare_lot_data, 11
prob_accept, 12
producer_risk, 12

run_dras_experiment, 13

select_dras_policy, 14
simulate_defect_process, 15
simulate_lots, 16
summarise_dras_monitor, 16

tune_dras_policy, 17

write_dras_report, 18