Package: KWELA (via r-universe)

October 14, 2026

Type Package

Title Hierarchical Adaptive 'RT-QuIC' Classification for Complex Matrices

Version 1.0.0

Description Extends 'RT-QuIC' (Real-Time Quaking-Induced Conversion) statistical analysis to complex environmental matrices through hierarchical adaptive classification. 'KWELA' is named after a deity of the Fore people of Papua New Guinea, among whom Kuru, a notable human prion disease, was identified. Implements a 6-layer architecture: hard gate biological constraints, per-well adaptive scoring, separation-aware combination, Youden-optimized cutoffs, replicate consensus, and matrix instability detection. Features dual-mode operation (diagnostic/research), auto-profile selection (Standard/Sensitive/Matrix-Robust), RAF integration for artifact detection, matrix-aware baseline correction, and multiple consensus rules. Methods include energy distance (Szekely and Rizzo (2013) <doi:10.1016/j.jspi.2013.03.018>), CRPS (Gneiting and Raftery (2007) <doi:10.1198/016214506000001437>), SSMD (Zhang (2007) <doi:10.1016/j.ygeno.2007.01.005>), and Jensen-Shannon divergence (Lin (1991) <doi:10.1109/18.61115>). This package implements methodology currently under peer review; please contact the author before publication using this approach. Development followed an iterative human-machine collaboration where all algorithmic design, statistical methodologies, and biological validation logic were conceptualized, tested, and iteratively refined by Richard A. Feiss through repeated cycles of running experimental data, evaluating analytical outputs, and selecting among candidate algorithms and approaches. AI systems ('Anthropic Claude' and 'OpenAI GPT') served as coding assistants and analytical sounding boards under continuous human direction. The selection of statistical methods, evaluation of biological plausibility, and all final methodology decisions were made by the human author. AI systems

did not independently originate algorithms, statistical approaches, or scientific methodologies.

License MIT + file LICENSE

URL https://github.com/RFeissIV/KWELA

BugReports https://github.com/RFeissIV/KWELA/issues

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 4.0.0)

Imports stats, graphics

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Richard A. Feiss IV [aut, cre] (ORCID: <https://orcid.org/0009-0008-0409-6042>)

Maintainer Richard A. Feiss IV <feiss026@umn.edu>

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

Date/Publication 2026-02-28 21:10:02 UTC

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

RemoteRef HEAD

RemoteSha 3837d16b92e0c494d7bce607adf5bc3c2ec3b182

Contents

KWELA-package 3
cohens_d 4
compute_instability_flags 5
compute_stochastic_profile 6
crps_empirical 7
cv 8
energy_distance 10
entropy 11
kwela_analyze 12
kwela_plate_normalize 14
kwela_summarize 15
robust_scale 17

Index **18**

KWELA-package *KWELA: Hierarchical Adaptive 'RT-QuIC' Classification*

Description

Extends 'RT-QuIC' (Real-Time Quaking-Induced Conversion) statistical analysis to complex environmental matrices through hierarchical adaptive classification. 'KWELA' is named after a deity of the Fore people of Papua New Guinea, among whom Kuru, a notable human prion disease, was identified.

Value

No return value. This is a package documentation page.

6-Layer Architecture

Layer 1: Hard Gate Biological constraint filter

Layer 2: Per-Well Scoring Profile-dependent adaptive transforms

Layer 3: Adaptive Combination Separation-aware score combiner

Layer 4: Adaptive Cutoff Youden-optimized threshold

Layer 5: Replicate Consensus Treatment-level classification

Layer 6: Instability Detection Matrix interference override

Dual-Mode Operation

diagnostic Default mode. Deterministic classification from TTT/MP/RAF evidence only. No stochastic rescue or score blending.

research Full adaptive architecture with stochastic rescue and distributional scoring.

Main Functions

`kwela_analyze` Main analysis function

`kwela_summarize` Treatment-level summary

`kwela_diagnostics` Diagnostic information

`compute_instability_flags` Matrix instability detection

Plate/Batch Functions

`kwela_plate_normalize` Multi-plate normalization

`kwela_bootstrap_summary` Bootstrap confidence intervals

cohens_d *Separation Quality Assessment*

Description

cohens_d computes Cohen's d effect size between two distributions.

assess_separation evaluates PC/NC separation quality and recommends a classification profile. Cohen's d thresholds are heuristic, calibrated against three RT-QuIC datasets.

Usage

```
cohens_d(x, y)

assess_separation(pc_mp, nc_mp, pc_ttt, nc_ttt)
```

Arguments

x	First distribution (numeric vector)
y	Second distribution (numeric vector)
pc_mp	Positive control MP values
nc_mp	Negative control MP values
pc_ttt	Positive control TTT values
nc_ttt	Negative control TTT values

Value

cohens_d returns a single numeric value (positive = x > y), or NA_real_ if insufficient data.

assess_separation returns a list with:

d_mp Cohen's d for MP

d_ttt Cohen's d for TTT

d_combined Median of absolute d values

regime Separation regime: "poor_separation", "moderate_separation", or "strong_separation"

recommended_profile Recommended profile: "matrix_robust", "sensitive", or "standard"

Examples

```
set.seed(42)
pc_mp <- rnorm(8, 100, 10)
nc_mp <- rnorm(8, 20, 5)
pc_ttt <- rnorm(8, 8, 1)
nc_ttt <- rnorm(8, 72, 5)

cohens_d(pc_mp, nc_mp)
assess_separation(pc_mp, nc_mp, pc_ttt, nc_ttt)
```

compute_instability_flags *Compute Instability Flags for a Treatment*

Description

Evaluates 6 deterministic metrics to detect matrix interference that compromises classification reliability. A treatment is flagged as unstable when it behaves unlike both positive and negative controls.

Usage

```
compute_instability_flags(
  trt_mp,
  trt_ttt,
  pc_mp,
  nc_mp,
  pc_ttt,
  nc_ttt,
  trt_raf = NULL,
  trt_mp_raf = NULL,
  pc_raf_mp_ratios = NULL,
  crossing_threshold = NA_real_,
  strictness = "moderate",
  has_raf = FALSE
)
```

Arguments

trt_mp	Finite MP values for treatment wells
trt_ttt	TTT values for treatment wells (may contain NA/Inf)
pc_mp, nc_mp	Positive/negative control MP values (finite)
pc_ttt, nc_ttt	Positive/negative control TTT values
trt_raf	RAF values for treatment wells (NULL if unavailable)
trt_mp_raf	MP values corresponding to RAF wells (NULL if unavailable)
pc_raf_mp_ratios	RAF/MP ratios from positive controls (NULL if unavailable)
crossing_threshold	TTT crossing threshold for this treatment's group
strictness	"moderate" (2+ flags), "strict" (1+), or "lenient" (3+)
has_raf	Whether RAF data is available

Details

The six metrics evaluated are:

1. Fano factor deviation from positive controls
2. Crossing variability (ambiguous threshold crossing rate)
3. Wasserstein distance from BOTH controls (normalized by NC spread)
4. Energy distance from BOTH controls (normalized by NC spread)
5. TTT dispersion ratio vs positive controls
6. RAF-MP ratio inconsistency vs positive controls

All metrics are deterministic (no random number generation).

Value

List with components: `unstable` (logical), `reasons` (character vector), `n_flags` (integer), `min_flags_required` (integer), `metrics` (named list).

Examples

```
set.seed(42)
flags <- compute_instability_flags(
  trt_mp = rnorm(8, 50, 20),
  trt_ttt = rnorm(8, 40, 15),
  pc_mp = rnorm(8, 100, 10),
  nc_mp = rnorm(8, 20, 5),
  pc_ttt = rnorm(8, 8, 1),
  nc_ttt = rnorm(8, 72, 5),
  crossing_threshold = 40,
  strictness = "moderate"
)
```

---

compute_stochastic_profile

Compute Stochastic Profile

Description

Computes comprehensive stochastic metrics for a sample compared to controls.

Usage

```
compute_stochastic_profile(sample_values, pc_values, nc_values)
```

Arguments

`sample_values`	Numeric vector of sample measurements
`pc_values`	Numeric vector of positive control measurements
`nc_values`	Numeric vector of negative control measurements

Value

List containing comprehensive stochastic metrics:

mean, var, cv, fano Basic sample statistics

cv_ratio_pc, cv_ratio_nc CV ratios vs controls

fano_ratio_pc Fano factor ratio vs PC

stochastic_index Normalized position between NC and PC variance

ssmd_vs_nc, ssmd_vs_pc SSMD vs controls

snr Signal-to-noise ratio vs NC

js_vs_pc, js_vs_nc, js_ratio Jensen-Shannon metrics

energy_vs_pc, energy_vs_nc, energy_ratio Energy distance metrics

wasserstein_vs_pc, wasserstein_vs_nc, wasserstein_ratio Wasserstein metrics

log_dist_pc, log_dist_nc Log-space Euclidean distances

Examples

```
set.seed(42)
pc <- rnorm(20, 100, 10)
nc <- rnorm(20, 20, 5)
sample <- rnorm(20, 80, 15)
profile <- compute_stochastic_profile(sample, pc, nc)
```

---

crps_empirical *Proper Scoring Rules*

Description

Proper scoring rules for evaluating probabilistic forecasts.

Usage

```
crps_empirical(forecast_samples, observation)

dawid_sebastiani(observation, predicted_mean, predicted_var)

log_predictive_score(observation, predicted_mean, predicted_sd)

interval_score(observation, lower, upper, alpha = 0.1)
```

Arguments

forecast_samples	Numeric vector of samples from forecast distribution
observation	Single observed value
predicted_mean	Predicted mean
predicted_var	Predicted variance
predicted_sd	Predicted standard deviation
lower	Lower bound of prediction interval
upper	Upper bound of prediction interval
alpha	Nominal miscoverage rate (default 0.1 for 90% interval)

Value

All functions return a single numeric value or NA_real_ if inputs are invalid.

crps_empirical returns a non-negative score (lower is better).

dawid_sebastiani returns the DS score (calibration metric).

log_predictive_score returns negative log density.

interval_score returns width plus penalties for miscoverage.

References

Gneiting T, Raftery AE (2007). Strictly proper scoring rules, prediction, and estimation. Journal of the American Statistical Association 102(477):359-378.

Examples

```
set.seed(42)
forecast <- rnorm(50, 5, 1)

crps_empirical(forecast, 5.5)
dawid_sebastiani(5.5, 5, 1)
log_predictive_score(5.5, 5, 1)
interval_score(5.5, 3, 7, alpha = 0.1)
```

---

cv *Stochasticity Metrics*

Description

Functions for quantifying stochastic variability in RT-QuIC data.

Usage

```
cv(x)

fano_factor(x)

ssmd(treatment, control)

normalized_stochastic_index(treatment_var, pc_var, nc_var)

snr_vs_control(treatment, control)
```

Arguments

x	Numeric vector
treatment	Numeric vector of treatment values
control	Numeric vector of control values
treatment_var	Treatment variance
pc_var	Positive control variance
nc_var	Negative control variance

Value

All functions return a single numeric value or NA_real_ if insufficient data.

cv returns the coefficient of variation (MAD-based).

fano_factor returns the variance-to-mean ratio.

ssmd returns the strictly standardized mean difference.

normalized_stochastic_index returns a value in [0, 1].

snr_vs_control returns the signal-to-noise ratio.

References

Zhang XHD (2007). A pair of new statistical parameters for quality control in RNA interference high-throughput screening assays. Genomics 89(4):552-61.

Examples

```
set.seed(42)
x <- rnorm(30, 10, 2)
y <- rnorm(30, 5, 2)

cv(x)
fano_factor(x + 10)
ssmd(x, y)
snr_vs_control(x, y)
normalized_stochastic_index(var(x), 4, 1)
```

energy_distance *Distance Metrics*

Description

Geometric distance metrics for comparing distributions.

energy_distance uses deterministic quantile subsampling for large datasets (no RNG dependency).

Usage

```
energy_distance(x, y, max_n = NULL)

wasserstein_1d(x, y)

log_euclidean_distance(x, y)

mahalanobis_distance(sample_params, reference_mean, reference_cov)
```

Arguments

x	First distribution (numeric vector)
y	Second distribution (numeric vector)
max_n	Maximum sample size before subsampling (NULL = no limit)
sample_params	Named numeric vector of sample parameters
reference_mean	Named numeric vector of reference means
reference_cov	Covariance matrix of reference distribution

Value

All functions return a single non-negative numeric value or NA_real_ if insufficient data or computation fails.

References

Szekely GJ, Rizzo ML (2013). Energy statistics: A class of statistics based on distances. Journal of Statistical Planning and Inference 143(8):1249-72.

Examples

```
set.seed(42)
x <- rnorm(30, 0, 1)
y <- rnorm(30, 2, 1)

energy_distance(x, y)
wasserstein_1d(x, y)
log_euclidean_distance(exp(x), exp(y))
mahalanobis_distance(c(1, 2), c(0, 0), diag(2))
```

`entropy` *Information-Theoretic Metrics*

Description

`entropy` computes Shannon entropy from discretized distribution.

`jensen_shannon` computes symmetric Jensen-Shannon divergence with Laplace smoothing.

Usage

```
entropy(x, n_bins = 10)

jensen_shannon(p, q, n_bins = 15)
```

Arguments

`x`	Numeric vector
`n_bins`	Number of histogram bins
`p`	First distribution (numeric vector)
`q`	Second distribution (numeric vector)

Value

`entropy` returns entropy in nats, or `NA_real_` if fewer than 3 values.

`jensen_shannon` returns JSD bounded in [0, ln(2)], or `NA_real_` if either distribution has fewer than 5 values.

References

Lin J (1991). Divergence measures based on the Shannon entropy. IEEE Transactions on Information Theory 37(1):145-151.

Examples

```
entropy(rnorm(100))

set.seed(42)
p <- rnorm(50, 0, 1)
q <- rnorm(50, 2, 1)
jensen_shannon(p, q)
```

`kwela_analyze` *KWELA Main Analysis Function*

Description

Implements hierarchical adaptive classification for 'RT-QuIC' data:

Layer 1: Hard Gate Biological constraint filter

Layer 2: Per-Well Scoring Profile-dependent transforms

Layer 3: Adaptive Combination Separation-aware combiner

Layer 4: Adaptive Cutoff Youden-optimized threshold

Layer 5: Replicate Consensus Treatment-level classification

Layer 6: Instability Detection Matrix interference override

Usage

```
kwela_analyze(
  data,
  pc_pattern = "\\bPositive\\s*Control\\b|^POS\\b|\\bPC\\b",
 nc_pattern = "\\bNegative\\s*Control\\b|^NEG\\b|\\bNC\\b|\\bTDB\\b|\\bBlank\\b",
  spiked_pattern = "\\+\\s*Pos|Pos\\s*\\d*%|spiked|\\+\\s*CWD",
  profile = c("auto", "standard", "sensitive", "matrix_robust"),
  consensus = c("majority", "strict", "flexible", "threshold"),
  consensus_threshold = 0.5,
  matrix_groups = NULL,
  use_raf = TRUE,
  mode = c("diagnostic", "research"),
  instability_check = TRUE,
  instability_strictness = c("moderate", "strict", "lenient"),
  verbose = TRUE
)
```

Arguments

`data`	Data frame with Treatment, TTT, MP columns (RAF optional)
`pc_pattern`	Regex for positive controls
`nc_pattern`	Regex for negative controls
`spiked_pattern`	Regex for spiked samples
`profile`	Classification profile: "auto", "standard", "sensitive", or "matrix_robust". "auto" selects based on separation quality.
`consensus`	Replicate consensus rule: "majority", "strict", "flexible", or "threshold"
`consensus_threshold`	For "threshold" consensus: minimum mean well score to classify treatment as positive (default 0.5)

`matrix_groups` Optional: column name for matrix grouping (enables per-group baseline correction). NULL = global baselines.

`use_raf` Logical: include RAF in scoring (default TRUE if present)

`mode` "diagnostic" (deterministic, no stochastic rescue) or "research" (full adaptive architecture). Default: "diagnostic".

`instability_check` Logical: run instability detection (default TRUE)

`instability_strictness` "moderate" (2+ flags), "strict" (1+), or "lenient" (3+). Controls override sensitivity.

`verbose` Print progress

Value

Data frame with per-well results including:

Type Well type: "PC", "NC", or "Sample"

is_spiked Logical indicating if sample is spiked

crossed_threshold Logical for TTT threshold crossing

has_signal Logical for signal above NC threshold

hard_gate_pass Logical for Layer 1 gate passage

artifact_flag Logical for suspected artifacts

stoch_rescue Logical for stochastic rescue (research mode only)

ttt_score TTT component score (0-1)

signal_score Signal component score (0-1)

raf_score RAF component score (0-1, if available)

stoch_score Stochastic component score (0-1)

well_score Combined well score (0-1)

well_positive Logical for well-level positive classification

seed_score Alias for well_score

classification Well classification: "POSITIVE", "NEGATIVE", "INCONCLUSIVE", "ARTIFACT_SUSPECT", or "INVALID"

confidence Confidence level: "VERY_HIGH", "HIGH", "MODERATE", "LOW", or "CONTROL"

trt_classification Treatment-level consensus classification (may be "INCONCLUSIVE_MATRIX_EFFECT" if instability detected)

trt_confidence Treatment-level confidence

trt_positive_rate Fraction of positive wells in treatment

trt_mean_score Mean well score for treatment

matrix_instability Logical for Layer 6 instability override

Plus attributes: pc_stats, nc_stats, separation, profile, consensus, optimal_cutoff, youden_j, thresholds, trt_summary, control_check, score_semantics, version, mode, instability_check, instability_strictness, instability_summary, group_controls, group_profiles, group_separation.

Examples

```
set.seed(42)
df <- data.frame(
  Treatment = c(rep("Positive Control", 8), rep("Negative Control", 8),
                rep("Sample_A", 8)),
  TTT = c(rnorm(8, 8, 1), rnorm(8, 72, 5), rnorm(8, 12, 3)),
  MP = c(rnorm(8, 100, 10), rnorm(8, 20, 5), rnorm(8, 85, 15))
)
result <- kwela_analyze(df)
summary <- kwela_summarize(result)
```

`kwela_plate_normalize` *Plate Normalization and Bootstrap*

Description

`kwela_plate_normalize` normalizes TTT and MP values within each plate using negative control baselines.

`kwela_bootstrap_summary` computes bootstrap confidence intervals for treatment scores. Set seed externally for reproducibility.

Usage

```
kwela_plate_normalize(
  data,
  plate_col = "Plate",
  pc_pattern = "\\bPositive\\s*Control\\b|^POS\\b|\\bPC\\b",
 nc_pattern = "\\bNegative\\s*Control\\b|^NEG\\b|\\bNC\\b|\\bTDB\\b|\\bBlank\\b",
  method = c("zscore", "median_ratio")
)

kwela_bootstrap_summary(result, B = 1000, conf = 0.95)
```

Arguments

`data`	Data frame with Treatment, TTT, MP columns
`plate_col`	Name of plate identifier column
`pc_pattern`	Regex for positive controls
`nc_pattern`	Regex for negative controls
`method`	Normalization method: "zscore" or "median_ratio"
`result`	Output from `kwela_analyze`
`B`	Number of bootstrap replicates (default 1000)
`conf`	Confidence level (default 0.95)

Value

`kwela_plate_normalize` returns a data frame with added columns:

plate_id Plate identifier

TTT_norm Normalized TTT values

MP_norm Normalized MP values

`kwela_bootstrap_summary` returns a data frame with:

Treatment Treatment name

n_wells Number of wells

mean_score Mean well score

score_lo Lower CI bound for mean score

score_hi Upper CI bound for mean score

positive_rate Observed positive rate

pr_lo Lower CI bound for positive rate

pr_hi Upper CI bound for positive rate

Examples

```
set.seed(42)
df <- data.frame(
  Treatment = c(rep("Positive Control", 4), rep("Negative Control", 4),
                rep("Sample_A", 4)),
  TTT = c(rnorm(4, 8, 1), rnorm(4, 72, 5), rnorm(4, 12, 3)),
  MP = c(rnorm(4, 100, 10), rnorm(4, 20, 5), rnorm(4, 85, 15)),
  Plate = rep("Plate1", 12)
)
normalized <- kwela_plate_normalize(df)

# Bootstrap example
result <- kwela_analyze(df, verbose = FALSE)
set.seed(123)
boot <- kwela_bootstrap_summary(result, B = 500)
```

---

kwela_summarize *KWELA Summary and Diagnostics*

Description

`kwela_summarize` extracts treatment-level consensus results.

`kwela_diagnostics` returns detailed diagnostic information.

Usage

```
kwela_summarize(result)

kwela_diagnostics(result)
```

Arguments

`result`	Output from `kwela_analyze`

Value

`kwela_summarize` returns a data frame with treatment-level results:

Treatment Treatment name

n_wells Number of wells

n_positive Number of positive wells

positive_rate Fraction positive

mean_score Mean well score

classification Treatment classification

confidence Confidence level

consensus_rule Consensus rule used

is_spiked Whether treatment is spiked

`kwela_diagnostics` returns a list with:

version KWELA version string

mode Analysis mode ("diagnostic" or "research")

profile Profile used

consensus Consensus rule used

optimal_cutoff Youden-optimized cutoff

youden_j Youden J statistic

separation Separation quality metrics

well_level Well-level statistics

treatment_level Treatment-level statistics

instability Instability detection results: check_enabled, strictness, n_inconclusive_matrix, treatments_flagged

n_samples Number of sample wells

n_spiked Number of spiked wells

n_unspiked Number of unspiked wells

Examples

```
set.seed(42)
df <- data.frame(
  Treatment = c(rep("Positive Control", 8), rep("Negative Control", 8),
                rep("Sample_A", 8)),
  TTT = c(rnorm(8, 8, 1), rnorm(8, 72, 5), rnorm(8, 12, 3)),
  MP = c(rnorm(8, 100, 10), rnorm(8, 20, 5), rnorm(8, 85, 15))
)
result <- kwela_analyze(df, verbose = FALSE)
summary <- kwela_summarize(result)
diag <- kwela_diagnostics(result)
```

`robust_scale` *Utility Functions*

Description

`robust_scale` provides MAD-based scale estimation.

`safe_sd` provides a guaranteed-positive standard deviation.

Usage

```
robust_scale(x)

safe_sd(x, na.rm = TRUE)
```

Arguments

`x`	Numeric vector
`na.rm`	Logical; should NA values be removed?

Value

`robust_scale` returns a robust scale estimate using MAD with fallback to IQR/1.349 and SD. Returns `NA_real_` if fewer than 2 values.

`safe_sd` returns standard deviation, guaranteed to be positive (returns machine epsilon if zero or NA).

Examples

```
robust_scale(rnorm(50))
robust_scale(c(1, 2, 3, 100))

safe_sd(c(5, 5, 5))
```

Index

assess_separation *(*cohens_d*)*, 4

cohens_d, 4
compute_instability_flags, *3*, 5
compute_stochastic_profile, 6
crps_empirical, 7
cv, 8

dawid_sebastiani *(*crps_empirical*)*, 7

energy_distance, 10
entropy, 11

fano_factor *(*cv*)*, 8

interval_score *(*crps_empirical*)*, 7

jensen_shannon *(*entropy*)*, 11

KWELA *(*KWELA-package*)*, 3
KWELA-package, 3
kwela_analyze, *3*, 12
kwela_bootstrap_summary, *3*
kwela_bootstrap_summary
        *(*kwela_plate_normalize*)*, 14
kwela_diagnostics, *3*
kwela_diagnostics *(*kwela_summarize*)*, 15
kwela_plate_normalize, *3*, 14
kwela_summarize, *3*, 15

log_euclidean_distance
        *(*energy_distance*)*, 10
log_predictive_score *(*crps_empirical*)*, 7

mahalanobis_distance *(*energy_distance*)*,
        10

normalized_stochastic_index *(*cv*)*, 8

robust_scale, 17

safe_sd *(*robust_scale*)*, 17

snr_vs_control *(*cv*)*, 8
ssmd *(*cv*)*, 8

wasserstein_1d *(*energy_distance*)*, 10