Package: SNPAssayR (via r-universe)

October 22, 2026

Title SNP Genotyping Assay Design and Visualization

Version 1.0.1

Description Designs cleaved amplified polymorphic sequence (CAPS), derived CAPS (dCAPS), conventional amplification refractory mutation system (ARMS) and tetra-primer ARMS assays for single-locus, biallelic single nucleotide polymorphisms. Evaluates sequence-compatible primer binding, reconstructs ideal polymerase chain reaction products and computes theoretical complete restriction digestion and conditional genotype profiles. Transparent preference ranking, scientific visualizations and reproducible exports retain assumptions, missing specificity evidence and bounded search limitations. An optional offline 'shiny' interface uses the independently available analysis functions. Predictions do not establish experimental amplification, digestion efficiency or genome-wide specificity. Implements perfect-match nearest-neighbor thermodynamics from SantaLucia (1998) <doi:10.1073/pnas.95.4.1460>, single-substitution site-creation dCAPS based on Neff et al. (1998) <doi:10.1046/j.1365-313X.1998.00124.x>, and conditional ARMS designs based on Newton et al. (1989) <doi:10.1093/nar/17.7.2503> and Ye et al. (2001) <doi:10.1093/nar/29.17.e88>.

License GPL-3

Copyright ICAR-Indian Institute of Rapeseed-Mustard Research (ICAR-IIRMR)

Encoding UTF-8

Depends R (>= 4.1.0)

Suggests testthat (>= 3.2.1), shiny (>= 1.8.0), bslib (>= 0.6.1)

Config/testthat/edition 3

RoxygenNote 7.3.1

NeedsCompilation no

Author Prashant Yadav [aut, cre] (Contribution: Conceptualization; methodology; scientific review), Vinod Kumar [aut] (Contribution: Software; package design; implementation; validation; testing), Vijay Veer Singh [aut] (Contribution: Conceptualization; supervision), ICAR-Indian Institute of Rapeseed-Mustard Research (ICAR-IIRMR) [cph]

Maintainer Prashant Yadav <prashantnduat@gmail.com>

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

Date/Publication 2026-10-22 19:20:49 UTC

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

RemoteRef HEAD

RemoteSha 227cd3cf3f660959ae29227403f03272c5058ae1

Contents

assay_bundle . . . 3
assay_preferences . . . 4
assay_scene . . . 5
assay_tables . . . 5
band_scene . . . 6
compare_profiles . . . 7
design_arms . . . 7
design_caps . . . 9
design_dcaps . . . 11
design_tetra_arms . . . 13
digest_fragments . . . 14
draw_scene . . . 15
evaluate_primer . . . 15
export_assay . . . 16
export_scene . . . 17
find_binding_sites . . . 18
find_hairpins . . . 18
fragment_profiles . . . 19
group_bands . . . 20
primer_interactions . . . 20
rank_assays . . . 21
read_locus . . . 22
reconstruct_amplicon . . . 22
replay_assay . . . 23
restriction_enzymes . . . 24
restriction_sites . . . 24
reverse_complement . . . 25
run_app . . . 26
screen_primer_pair . . . 26
simulate_digest . . . 27
snpassay_app . . . 28

validate_snp 28

Index **30**

`assay_bundle` *Construct a reproducible scientific export bundle*

Description

Scientific content is deterministic. Wall-clock/environment metadata is kept separately and does not enter ranking. All source result records are retained. No seed is required: the analytical workflows and ranking are deterministic.

Usage

```
assay_bundle(
  result,
  preferences = assay_preferences(),
  selected = if (length(result$candidates)) 1L else NULL
)
```

Arguments

`result`	Supported assay result from the installed package version.
`preferences`	User choices from `assay_preferences()`.
`selected`	Original candidate index for linked figures; NULL for no selection.

Value

S3 snpassay_bundle with scientific content and separate execution record.

Examples

```
# Artificial coordinate fixture; three-base oligos are not laboratory designs.
s <- validate_snp(read_locus("ACGTTGAATTCAAAAAAGTC", "fixture"), 6, "G", "A")
x <- design_caps(s, restriction_enzymes()[1, , drop = FALSE],
  primer_lengths = 3, product_min = 20, product_max = 20,
  gc_range = c(0, 1), tm_range = c(-250, 100),
  max_tm_difference = 100, screening_mismatches = 0)
bundle <- assay_bundle(x)
names(bundle$scientific)
```

assay_preferences — *Set explicit laboratory preferences for candidate comparison*

Description

Preferences are user choices, not validated biological thresholds. Ranking compares candidates within one method and locus; scores across methods are not comparable. Smaller weighted penalties are preferred, not more likely to amplify. No probability, efficiency or experimental specificity is modeled.

Usage

```
assay_preferences(
 weights = c(quality = 1, target = 1, copies = 1, discrimination = 1, resolution = 1,
    evidence = 1, preferences = 1),
  preferred_tm = 60,
  preferred_gc = 0.5,
  preferred_product = 200,
  available_enzymes = NULL,
  tolerance_bp = 0L,
  min_size_bp = 1L
)
```

Arguments

`weights` Named nonnegative weights for quality, target, copies, discrimination, resolution, evidence and preferences. At least one positive.

`preferred_tm` Preferred perfect-match modeled primer Tm in degrees C.

`preferred_gc` Preferred primer GC fraction.

`preferred_product` Preferred PCR product size in bp.

`available_enzymes` Character vector of enzyme names available in the lab; NULL means no inventory preference, not evidence that enzymes are available.

`tolerance_bp`, `min_size_bp` Arithmetic size comparison/display preferences.

Value

A validated simple list of user preferences and fixed scale constants.

Examples

```
assay_preferences(weights=c(quality=2,target=1,copies=2,discrimination=1,
  resolution=1,evidence=1,preferences=1))
```

assay_scene — *Extract auditable coordinate features from an assay result*

Description

Extract auditable coordinate features from an assay result

Usage

```
assay_scene(result, candidate = 1L)
```

Arguments

result	CAPS, dCAPS, ARMS or tetra-primer design result.
candidate	One-based retained candidate index.

Value

S3 snpassay_scene. All coordinates are on the original locus plus axis; cleavage boundaries are explicitly distinguished from inclusive base intervals.

Examples

```
# Artificial coordinate fixture; three-base oligos are not laboratory designs.
s <- validate_snp(read_locus("ACGTTGAATTCAAAAAAGTC", "fixture"), 6, "G", "A")
x <- design_caps(s, restriction_enzymes()[1, , drop = FALSE],
  primer_lengths = 3, product_min = 20, product_max = 20,
  gc_range = c(0, 1), tm_range = c(-250, 100),
  max_tm_difference = 100, screening_mismatches = 0)
assay_scene(x)$features
```

assay_tables — *Extract exact scientific tables for all retained assay candidates*

Description

Preserves original candidate order and indices, 5-prime to 3-prime primers, plus-axis coordinates, complete fragment multiplicity and empty conditional reaction lanes. Profiles are conditional; missing bands are not genotype calls.

Usage

```
assay_tables(result)
```

Arguments

result	A supported assay design result.

Value

Named list of data frames: primers, coordinate features, fragments, restriction sites, conditional profiles and supplied-copy screening records. The full nested result retains all detailed pair/template binding information.

Examples

```
# Artificial coordinate fixture; three-base oligos are not laboratory designs.
s <- validate_snp(read_locus("ACGTTGAATTCAAAAAAGTC", "fixture"), 6, "G", "A")
x <- design_caps(s, restriction_enzymes()[1, , drop = FALSE],
  primer_lengths = 3, product_min = 20, product_max = 20,
  gc_range = c(0, 1), tm_range = c(-250, 100),
  max_tm_difference = 100, screening_mismatches = 0)
assay_tables(x)$primers
```

---

band_scene — *Extract separate theoretical size lanes from validated assay/profile objects*

Description

Extract separate theoretical size lanes from validated assay/profile objects

Usage

```
band_scene(result, candidate = 1L, tolerance_bp = 0L, min_size_bp = 1L)
```

Arguments

`result`	Assay design, or snpassay_profiles from `fragment_profiles()`.
`candidate`	One-based assay candidate index (ignored for profiles).
`tolerance_bp, min_size_bp`	Explicit schematic grouping settings from `group_bands()`.

Value

S3 snpassay_scene with full source-linked fragments and schematic bands. Conventional ARMS preserves reaction labels; tetra includes its existing control.

Examples

```
profiles <- fragment_profiles(c(20, 40, 40), 100)
scene <- band_scene(profiles, min_size_bp = 30)
scene$lanes$AB$records
```

compare_profiles — *Compare visible fragment-size sets under a disclosed arithmetic heuristic*

Description

A visible size is diagnostic only when every size in the other visible profile is farther away than tolerance. Exact duplicates are ignored only for this comparison; full fragment multisets are preserved. Distinct profiles may remain indistinguishable under the tolerance or cutoff. No gel detection, migration, intensity, calibrated resolution or genotype-call claim is made.

Usage

```
compare_profiles(first, second, tolerance_bp = 0L, min_size_bp = 1L)
```

Arguments

`first`, `second`	Positive integer size vectors, including empty profiles.
`tolerance_bp`	Nonnegative integer maximum cross-profile size difference considered indistinguishable; equality at the boundary is indistinguishable.
`min_size_bp`	Inclusive schematic visibility cutoff.

Value

S3 snpassay_profile_comparison with visible/full diagnostic size sets, cutoff-masked differences, parameters, warnings and an arithmetic classification.

Examples

```
compare_profiles(c(20,40,40),100,min_size_bp=30)$classification
```

design_arms — *Design conventional ARMS primers for two separate allele-specific reactions*

Description

Both allele-specific primers end at the SNP and use the same opposing conserved common primer. Reverse terminal bases are allele complements. Optional one extra substitution is explicit and identical in both allele-specific oligos outside the terminal SNP. No mismatch-extension probability or guarantee is calculated. Profiles for AA/AB/BB assume ideal discrimination; count-based compatibility screening independently retains possible non-target/copy products.

Usage

```
design_arms(
  snp,
  strands = c("+", "-"),
  primer_lengths = 18:24,
  additional_mismatch_position = NULL,
  additional_mismatch_bases = c("A", "C", "G", "T"),
  product_min = 80L,
  product_max = 500L,
  gc_range = c(0.2, 0.8),
  tm_range = c(45, 75),
  max_tm_difference = 5,
  sodium_molar = 0.05,
  total_strand_molar = 5e-07,
  max_candidates = 20L,
  max_specific_checks = 1000L,
  max_pair_checks = 10000L,
  max_common_primers = 200L,
  max_common_checks = 10000L,
  screening_mismatches = 2L
)
```

Arguments

`snp` Validated biallelic SNP from `validate_snp()`. A is reference, B alternative.

`strands` Designed allele-specific primer orientations, subset of c("+","-").

`primer_lengths` Allowed whole-number lengths 2 to 60; ascending enumeration.

`additional_mismatch_position`
NULL by default (no extra mismatch), or one index 2 to 60 measured from the 3-prime end. Shorter lengths are skipped.

`additional_mismatch_bases`
Allowed replacement oligo bases, a subset of A/C/G/T. Used only when an additional position is specified; the original base is excluded. No choice is described as biochemically optimal.

`product_min`, `product_max`
Inclusive product-size bounds, 1 to 10000 bp.

`gc_range`, `tm_range`
Inclusive GC fraction and perfect-match oligo Tm bounds. No mismatched-target Tm model is applied.

`max_tm_difference`
Maximum absolute Tm difference for either reaction pair.

`sodium_molar`, `total_strand_molar`
Conditions passed to `evaluate_primer()`.

`max_candidates` Maximum returned three-primer sets, 1 to 100.

`max_specific_checks`
Maximum allele-primer variants inspected, 1 to 10000.

`max_pair_checks`
: Maximum specific/common geometries inspected, 1 to 100000.

`max_common_primers`
: Maximum passing common primers per orientation, 1 to 500.

`max_common_checks`
: Maximum common windows inspected per orientation, 1 to 100000.

`screening_mismatches`
: Count-based screening tolerance per site. Must include the terminal non-target mismatch plus any extra substitution; default 2. This deliberately screens risks rather than assuming failed extension.

Value

An S3 `snpassay_arms` list with three-primer candidates, full conditional mature products on both original alleles, intended mismatch records, separate ideal reaction profiles, sequence-compatibility screens, controls warning, parameters, deterministic search coverage and model caveats.

References

Newton et al. (1989), doi:10.1093/nar/17.7.2503.

Examples

```
# Artificial geometry fixture; short oligos are not laboratory recommendations.
s <- validate_snp(read_locus("ACGAAAGTC", "fixture"), 3, "G", "T")
x <- design_arms(s,strands="+",primer_lengths=3,product_min=9,product_max=9,
  gc_range=c(0,1),tm_range=c(-250,100),max_tm_difference=100)
x$candidates[[1]]$ideal_reaction_profiles
```

`design_caps` *Design bounded natural CAPS candidates with conserved flanking primers*

Description

Deterministically searches flanks nearest the SNP first, with lengths ascending. Primers never cover the SNP. Each intended allele product is reconstructed from its full outer binding coordinates and completely digested. Candidates require different natural sites AND different exact fragment-size band sets. All internal sites are retained, including additional nondiagnostic sites. Search limits are explicit; a limited search cannot establish absence. These are computational candidates, not experimentally validated assays.

Usage

```
design_caps(
  snp,
  enzymes = restriction_enzymes(),
  primer_lengths = 18:24,
  product_min = 80L,
  product_max = 500L,
  gc_range = c(0.2, 0.8),
  tm_range = c(45, 75),
  max_tm_difference = 5,
  sodium_molar = 0.05,
  total_strand_molar = 5e-07,
  max_candidates = 20L,
  max_pair_checks = 10000L,
  max_primer_candidates = 200L,
  max_primer_checks = 10000L,
  screening_mismatches = 1L,
  min_band_difference = 0L
)
```

Arguments

`snp` A validated biallelic SNP from `validate_snp()`.

`enzymes` Supported enzyme-record data frame, default the curated table.

`primer_lengths` Allowed whole-number lengths, 2 to 60 bp.

`product_min`, `product_max` Inclusive product-size bounds, 1 to 10000 bp.

`gc_range` Inclusive GC-fraction bounds, two values in 0..1.

`tm_range` Inclusive modeled Tm bounds in Celsius, two finite values.

`max_tm_difference` Maximum absolute modeled primer Tm difference.

`sodium_molar`, `total_strand_molar` Conditions passed to `evaluate_primer()`.

`max_candidates` Maximum returned candidates, 1 to 100.

`max_pair_checks` Maximum flanking primer pairs inspected, 1 to 100000.

`max_primer_candidates` Maximum quality-passing primers per orientation, 1 to 500; reaching the limit is conservatively reported as truncation.

`max_primer_checks` Maximum windows inspected per orientation, 1 to 100000.

`screening_mismatches` Supplied-copy screening mismatch tolerance per primer; 0 to the shortest primer length. No extension model is implied.

`min_band_difference`
: Minimum nearest opposing-band distance in bp for at least one exclusive band. Zero means exact band-set difference only. This is a user-selected size heuristic, not validated gel resolution.

Value

An S3 `snpassay_caps` list with candidate summary, detailed primer evaluations, reconstructed products, complete digests, supplied-copy risk screening, parameters, rejected-pair counts, search coverage and caveats.

Examples

```
# Artificial geometry fixture; short primers are not laboratory recommendations.
x <- validate_snp(read_locus("ACGTTGAATTCAAAAAAGTC", "fixture"), 6, "G", "A")
design_caps(x, restriction_enzymes()[1, ], primer_lengths=3,
  product_min=20, product_max=20, gc_range=c(0,1), tm_range=c(-250,100),
  max_tm_difference=100, screening_mismatches=0)$summary
```

`design_dcaps` *Design single-substitution derived CAPS candidates on either primer strand*

Description

Creates a new SNP-spanning restriction site in one reconstructed allele using exactly one intentional primer substitution outside the SNP. Requires complete digest band-set difference; a local motif alone never certifies a candidate. Site-removal-only strategies, multiple substitutions, tails and overlapping primers are excluded. Search caps retain uncertainty, not proof of absence.

Usage

```
design_dcaps(
  snp,
  enzymes = restriction_enzymes(),
  primer_lengths = 18:24,
  primer_gaps = 0:3,
  min_mismatch_from_3prime = 2L,
  product_min = 80L,
  product_max = 500L,
  gc_range = c(0.2, 0.8),
  tm_range = c(45, 75),
  max_tm_difference = 5,
  sodium_molar = 0.05,
  total_strand_molar = 5e-07,
  max_candidates = 20L,
  max_design_checks = 10000L,
  max_pair_checks = 10000L,
```

```
  max_common_primers = 200L,
  max_common_checks = 10000L,
  screening_mismatches = 1L,
  min_band_difference = 0L
)
```

Arguments

`snp` Validated biallelic SNP from `validate_snp()`.

`enzymes` Supported enzyme table, default `restriction_enzymes()`.

`primer_lengths` Allowed primer lengths, whole numbers 2 to 60.

`primer_gaps` Numbers of unprimed bases between the designed primer's 3-prime endpoint and SNP, whole numbers 0 to 10; default 0 to 3.

`min_mismatch_from_3prime`
Minimum intentional mismatch index counted from the designed primer's 3-prime end; default 2 excludes terminal changes.

`product_min`, `product_max`
Inclusive size bounds, 1 to 10000 bp.

`gc_range`, `tm_range`
Inclusive GC fraction and perfect-match Tm bounds. Mismatched primer-template Tm is NOT modeled.

`max_tm_difference`
Maximum absolute perfect-match primer Tm difference.

`sodium_molar`, `total_strand_molar`
Conditions for `evaluate_primer()`.

`max_candidates` Maximum returned candidates, 1 to 100.

`max_design_checks`
Maximum single-base variants inspected, 1 to 100000.

`max_pair_checks`
Maximum designed/common primer pairs inspected, 1 to 100000.

`max_common_primers`
Maximum passing common primers per orientation, 1 to 500.

`max_common_checks`
Maximum common windows inspected per orientation, 1 to 100000.

`screening_mismatches`
Supplied-copy tolerance per primer, at least 1 to include the intentional mismatch, at most the shortest primer length.

`min_band_difference`
Size-only nearest opposing-band threshold, 0 to 10000 bp; not measured electrophoretic resolution.

Value

An S3 `snpassay_dcaps` list containing detailed candidates and summary, both mature products, intentional edits, full digests, original-template specificity risk screens, quality warnings, parameters and search coverage.

References

Neff et al. (1998), doi:10.1046/j.1365-313X.1998.00124.x; Neff et al. (2002), doi:10.1016/S0168-9525(02)02820-2.

Examples

```
# Artificial geometry fixture; short primers are not laboratory recommendations.
s <- validate_snp(read_locus("ACGGACTTCAAAGTC", "fixture"), 9, "C", "G")
design_dcaps(s,restriction_enzymes()[1,],primer_lengths=c(3,8),
  primer_gaps=0,product_min=15,product_max=15,gc_range=c(0,1),
  tm_range=c(-250,100),max_tm_difference=100)$summary
```

---

design_tetra_arms *Design bounded tetra-primer ARMS candidates*

Description

Combines independently designed forward and reverse ARMS candidates. Outer primers must lie strictly outside the opposing inner primer intervals. Both assignments of alleles to inner orientations are considered. All six unordered primer pairs are screened on original alleles and supplied comparison copies. AA/AB/BB profiles assume successful control amplification and ideal allele discrimination. Size separation is a user-selected arithmetic filter, not a prediction of gel resolution. No experimental amplification is inferred. Shared product_min and product_max bounds apply to all three intended products, including the outer control. All six risk screens independently cover products from 1 to 10000 bp, including products outside the design range.

Usage

```
design_tetra_arms(
  snp,
  min_size_difference = 20L,
  max_candidates = 20L,
  pool_size = 20L,
  max_combinations = 1000L,
  ...
)
```

Arguments

`snp`	Validated biallelic SNP.
`min_size_difference`	Minimum pairwise intended product size difference, in bases. A transparent length filter, not an electrophoretic model.
`max_candidates`	Maximum retained tetra candidates (1 to 100).
`pool_size`	Maximum conventional candidates per inner orientation (1 to 100).
`max_combinations`	Maximum examined orientation/allele combinations.

...	Shared named design constraints passed to design_arms(), excluding snp, strands and max_candidates. Its explicit search limits remain operative.

Value

S3 snpassay_tetra object with four primers, three intended products, all six interaction/compatibility assessments and conditional genotype profiles.

Examples

```
# Artificial coordinate fixture; short oligos are not laboratory designs.
s <- validate_snp(read_locus("ACGTCAGTACGATCGTACGTTAGCGATCGA", "fixture"),
  14, "C", "A")
x <- design_tetra_arms(s, primer_lengths = 3,
  product_min = 12, product_max = 25, gc_range = c(0, 1),
  tm_range = c(-250, 150), max_tm_difference = 200,
  min_size_difference = 1, pool_size = 2, max_candidates = 1)
x$candidates[[1]]$intended_products
```

---

digest_fragments *Split a linear strand axis at complete-digest cut boundaries*

Description

Split a linear strand axis at complete-digest cut boundaries

Usage

```
digest_fragments(length_bp, cut_boundaries = integer())
```

Arguments

length_bp	Whole-number linear product length, 1 to 10000.
cut_boundaries	Numeric vector of whole-number boundaries 0 to length_bp. Duplicates are deduplicated; terminal boundaries do not make zero fragments.

Value

An S3 snpassay_fragments list with positive fragment lengths, inclusive base intervals, internal cuts, terminal boundaries and input cuts.

Examples

```
digest_fragments(100, c(20,60))$fragments
```

draw_scene — *Draw a source-linked scientific scene using base R graphics*

Description

Draw a source-linked scientific scene using base R graphics

Usage

```
draw_scene(scene)
```

Arguments

scene	Scene from assay_scene() or band_scene().

Value

Invisibly the input scene. Uses an accessible color palette plus text, symbols and arrow orientation; band strokes have uniform width.

Examples

```
scene <- band_scene(fragment_profiles(c(20, 40, 40), 100))
draw_scene(scene)
```

evaluate_primer — *Evaluate an unambiguous primer and its perfect-match duplex*

Description

Uses SantaLucia (1998), Table 2 and equations 3 and 8, with a contribution from each terminal base pair, automatic self-complementarity detection, and sodium correction to entropy. The model assumes an unphosphorylated, perfectly matched DNA duplex, two-state melting and salt-independent enthalpy. No magnesium, dNTP or mismatch correction is applied. This is not a PCR annealing-temperature prescription. Sequence interaction scores are geometric screens, not free energies or amplification probabilities.

Usage

```
evaluate_primer(primer, sodium_molar = 0.05, total_strand_molar = 5e-07)
```

Arguments

`primer` Uppercase A/C/G/T primer, 2 to 60 bases. These are software bounds, not a claim that all such lengths are experimentally validated.

`sodium_molar` Sodium concentration in mol/L, 0.01 to 1. Calibration data cover 0.01 to 0.3 M with a 1 M parameter baseline.

`total_strand_molar` Total strand concentration in mol/L, positive and at most 0.001 (software bound). For non-self-complementary duplexes this is the sum of two equimolar strands; the logarithm uses total/4. For self-complementary strands it uses the total directly.

Value

An S3 `snpassay_primer` list with length, GC fraction, Tm (Celsius), decomposed enthalpy/entropy terms, parameters, sequence scores and caveats.

References

SantaLucia J Jr (1998). doi:10.1073/pnas.95.4.1460.

Examples

```
evaluate_primer("ACGTTGCAACGTTGCAACGT")$tm_celsius
```

`export_assay` *Export a bundle as exact tables, full RDS, methods and selected figures*

Description

Writes a new directory transactionally; refuses overwrite. No external zip command is required. RDS retains scientific nested data and replay request; CSVs are presentation tables, not substitutes for the complete RDS object.

Usage

```
export_assay(bundle, directory, figure_formats = "pdf")
```

Arguments

`bundle` Object from `assay_bundle()`.

`directory` New output directory; its parent must exist.

`figure_formats` Subset of pdf, svg, png, or character() for tables only.

Value

Invisibly the exported absolute directory. All files get an MD5 manifest for accidental-change checking, not cryptographic authenticity.

Examples

```
# Artificial coordinate fixture; three-base oligos are not laboratory designs.
s <- validate_snp(read_locus("ACGTTGAATTCAAAAAAGTC", "fixture"), 6, "G", "A")
x <- design_caps(s, restriction_enzymes()[1, , drop = FALSE],
  primer_lengths = 3, product_min = 20, product_max = 20,
  gc_range = c(0, 1), tm_range = c(-250, 100),
  max_tm_difference = 100, screening_mismatches = 0)
bundle <- assay_bundle(x)
destination <- tempfile("snpassay-export-")
export_assay(bundle, destination, figure_formats = character())
list.files(destination)
unlink(destination, recursive = TRUE)
```

`export_scene` *Export a scene using supported local R devices*

Description

Export a scene using supported local R devices

Usage

```
export_scene(scene, file, width = 10, height = 7, res = 300L)
```

Arguments

`scene`	Validated source-linked scene.
`file`	Destination ending in .svg, .png or .pdf. Existing files are refused.
`width, height`	Positive dimensions in inches, 3 to 30.
`res`	PNG dots per inch, 72 to 1200; ignored for vector formats.

Value

Invisibly the absolute output path. Unsupported SVG/PNG devices produce an informative error; no compulsory external binary or online renderer.

Examples

```
scene <- band_scene(fragment_profiles(c(20, 40, 40), 100))
path <- tempfile(fileext = ".pdf")
export_scene(scene, path)
file.exists(path)
unlink(path)
```

find_binding_sites *Find ungapped primer-compatible sites on both template strands*

Description

All coordinates refer to the supplied plus-reference sequence, one-based inclusive. A plus site has its 3-prime coordinate at end; a minus site at start. Mismatch position 1 from the 3-prime end is terminal. Ambiguous windows are reported separately and never treated as definite matches. Full-length ungapped sites only; no tails, indels or extension model.

Usage

```
find_binding_sites(primer, template, max_mismatches = 0L)
```

Arguments

`primer`	Uppercase canonical DNA, 1 to 60 bases, written 5-prime to 3-prime.
`template`	Normalized uppercase DNA, at most 10000 bases; IUPAC allowed.
`max_mismatches`	Whole-number mismatch tolerance from zero to primer length.

Value

An S3 `snpassay_binding` list containing typed sites, skipped ambiguous windows, exact inputs, tolerance and screening limitations.

Examples

```
find_binding_sites("ACG", "ACGAAACGT")$sites
```

find_hairpins *Enumerate sequence-compatible hairpin stems*

Description

For each outer base pair, reports the longest contiguous inward complementary stem that retains at least min_loop unpaired bases. Nested stems may appear separately. No bulges, mismatched stems or thermodynamic stability are modeled.

Usage

```
find_hairpins(primer, min_loop = 3L, min_stem = 2L)
```

Arguments

`primer`	Uppercase canonical DNA, 1 to 60 bases.
`min_loop`	Minimum unpaired loop bases, integer 3 to 60; a geometric rule.
`min_stem`	Minimum reported contiguous stem pairs, integer 1 to 30.

Value

A list with a typed stem table and explicit geometry parameters.

Examples

```
find_hairpins("GCGAAACGC")$stems
```

`fragment_profiles` *Construct conditional biallelic fragment mixtures*

Description

Retains every input fragment, including repeats. AB concatenates one theoretical product profile from each allele; counts do not imply genomic dosage, DNA mass, band intensity or amplification efficiency. For digests, pass complete fragment length vectors from `simulate_digest()`. For PCR products, profiles must already encode the stated ideal allele-discrimination assumptions. Separate conventional ARMS reactions must remain separate and must not be concatenated into one lane.

Usage

```
fragment_profiles(reference_fragments, alternative_fragments)
```

Arguments

`reference_fragments`
: Positive integer fragment sizes, retaining repeats.

`alternative_fragments`
: Positive integer fragment sizes, retaining repeats.

Value

S3 snpassay_profiles with full AA/AB/BB size lists and allele-labelled source records. Assumes an isolated biallelic locus and complete theoretical inputs.

Examples

```
fragment_profiles(c(20,40,40),100)$profiles
```

group_bands — *Group theoretical sizes without discarding fragment multiplicity*

Description

Groups ascending sizes against each group's smallest member, so the group's full span cannot exceed tolerance. No transitive chaining or mobility model. Exact equal sizes comigrate in this size-only schematic; nonzero tolerance is an explicit arithmetic grouping preference, never measured gel resolution.

Usage

```
group_bands(sizes, tolerance_bp = 0L, min_size_bp = 1L)
```

Arguments

`sizes`	Positive integer sizes; empty profiles are allowed.
`tolerance_bp`	Nonnegative integer maximum group span in bases.
`min_size_bp`	Minimum retained size, inclusive; smaller fragments remain in full records but are hidden in the schematic band view.

Value

S3 snpassay_bands containing full fragment records, displayed groups, hidden records and explicit parameters. Multiplicity is not intensity.

Examples

```
group_bands(c(20,40,40),min_size_bp=30)$bands
```

primer_interactions — *Screen ungapped antiparallel primer complementarity*

Description

Aligns primer A with the reverse complement of B at every overlapping offset. Offset zero aligns their first displayed bases; positive offsets shift the reverse-complemented B right. Reports contiguous complementary runs, including runs ending at A's 3-prime base and starting at reverse-complemented B's first base (B's 3-prime base). Bulges and internal gaps are not modeled.

Usage

```
primer_interactions(primer_a, primer_b)
```

Arguments

`primer_a`, `primer_b`
: Uppercase canonical DNA, 1 to 60 bases each.

Value

A list with every offset score and maxima; sequence scores only.

Examples

```
primer_interactions("AAAA", "TTTT")$longest_run
```

---

rank_assays — *Rank retained assays with fully disclosed preference penalties*

Description

Filters malformed candidate geometry, primer evaluations, terminal SNPs and digest records; exclusions retain original indices and reasons. Repeated IDs are rejected. Score is the sum of weight times seven disclosed penalties; ascending original index breaks ties. It is a within-result preference index, not an assay success probability, allele specificity guarantee or cross-method comparison. Zero weights ignore a criterion but never erase its evidence.

Usage

```
rank_assays(result, preferences = assay_preferences())
```

Arguments

`result`	Supported assay design object.
`preferences`	Preferences from `assay_preferences()`.

Value

S3 snpassay_ranking with sorted components, contributions, exclusions, weights, formula and limitations; original result is never reordered.

Examples

```
# Artificial coordinate fixture; three-base oligos are not laboratory designs.
s <- validate_snp(read_locus("ACGTTGAATTCAAAAAAGTC", "fixture"), 6, "G", "A")
x <- design_caps(s, restriction_enzymes()[1, , drop = FALSE],
  primer_lengths = 3, product_min = 20, product_max = 20,
  gc_range = c(0, 1), tm_range = c(-250, 100),
  max_tm_difference = 100, screening_mismatches = 0)
ranking <- rank_assays(x)
ranking$components
```

read_locus — *Import a single locus from DNA or FASTA text*

Description

Whitespace removal and uppercasing are explicitly recorded. IUPAC ambiguity is retained and reported; gaps and RNA bases are rejected. FASTA identifiers are the first whitespace-delimited header tokens and must be unique. All records are validated; multiple records require explicit selection. No file paths are opened implicitly: join readLines(path) with newline characters. Comparison sequences are unaligned and confer no specificity evidence. Limits are 10000 bases per sequence, 50 FASTA records and 50 comparisons.

Usage

```
read_locus(sequence, locus_id, record = NULL, comparison_sequences = NULL)
```

Arguments

`sequence`	One DNA string or multiline FASTA text.
`locus_id`	Nonempty locus identifier, preserved verbatim.
`record`	FASTA record identifier; required for multi-record input.
`comparison_sequences`	Optional named character vector of DNA strings.

Value

A simple S3 `snpassay_locus` list with normalized sequence, normalization provenance, ambiguity positions, comparisons and metadata.

Examples

```
locus <- read_locus("acgt", "artificial_example")
locus$normalization
```

reconstruct_amplicon — *Reconstruct both ideal mature products with at most one primer substitution*

Description

Primers are full-length, untailed and nonoverlapping, with the SNP strictly between them. The forward sequence replaces the product's left endpoint; the reverse complement of the reverse primer replaces its right endpoint. Rejects SNP coverage, ambiguous product bases and more than one substitution in total. These are conditional mature-product sequences, not observed PCR.

Usage

```
reconstruct_amplicon(
  snp,
  forward_primer,
  reverse_primer,
  forward_start,
  reverse_start
)
```

Arguments

`snp` Validated SNP from `validate_snp()`.

`forward_primer`, `reverse_primer`
Canonical uppercase DNA, 2 to 60 bases, written 5-prime to 3-prime.

`forward_start`, `reverse_start`
One-based left coordinates of the plus-axis binding intervals. The reverse primer's 3-prime base maps to reverse_start.

Value

An S3 `snpassay_amplicon` list with both full reconstructed sequences, primer/template mismatch positions, strand-aware edits and provenance.

Examples

```
# Artificial geometry fixture, not laboratory primer recommendations.
s <- validate_snp(read_locus("ACGGACTTCAAAGTC", "fixture"), 9, "C", "G")
reconstruct_amplicon(s, "ACGGAATT", "GAC", 1, 13)$reference_sequence
```

`replay_assay` *Rerun the saved design request with installed R functions*

Description

Rerun the saved design request with installed R functions

Usage

```
replay_assay(bundle)
```

Arguments

`bundle` Object from `assay_bundle()`, optionally restored with readRDS().

Value

Fresh assay result; no code strings are evaluated and no files/network are opened. Version mismatch fails rather than silently claiming reproducibility.

Examples

```
# Artificial coordinate fixture; three-base oligos are not laboratory designs.
s <- validate_snp(read_locus("ACGTTGAATTCAAAAAAGTC", "fixture"), 6, "G", "A")
x <- design_caps(s, restriction_enzymes()[1, , drop = FALSE],
  primer_lengths = 3, product_min = 20, product_max = 20,
  gc_range = c(0, 1), tm_range = c(-250, 100),
  max_tm_difference = 100, screening_mismatches = 0)
bundle <- assay_bundle(x)
identical(replay_assay(bundle), x)
```

---

restriction_enzymes *Retrieve the curated initial restriction-enzyme facts*

Description

Twenty independently assembled entries support one palindromic recognition motif and one symmetric double-strand cut within it. Offset p means cutting after p bases from the motif's left edge on the displayed plus-reference axis. Both strand offsets use this same axis; the minus strand runs 3-prime to 5-prime on the display. No buffers, activity models or database code are copied.

Usage

```
restriction_enzymes()
```

Value

A data frame with motifs, both cut offsets, source URLs/dates and version.

Examples

```
restriction_enzymes()
```

---

restriction_sites *Find every supported restriction recognition site*

Description

Includes overlapping matches. A palindromic motif is scanned once on the plus axis to avoid counting the same double-strand site twice. Every matched window must be canonical DNA; ambiguous template windows are separately unresolved, even at a degenerate motif position. Cuts are boundary coordinates 0..N.

Usage

```
restriction_sites(sequence, enzyme)
```

Arguments

sequence	Normalized uppercase DNA, at most 10000 bases; IUPAC permitted.
enzyme	Curated name or one mathematically valid enzyme-record data frame.

Value

An S3 snpassay_restriction_sites list with all sites, both strand cuts, overhang metadata, ambiguity and verification scope.

Examples

```
restriction_sites("AGAATTCT", "EcoRI")$sites
```

---

reverse_complement *Reverse-complement validated uppercase DNA*

Description

Supports the DNA IUPAC alphabet with R/Y, K/M, B/V and D/H complement pairs and self-complementary S, W and N. No case or whitespace changes are made: import unnormalized input with read_locus() first.

Usage

```
reverse_complement(sequence)
```

Arguments

sequence	One uppercase DNA string, at most 10000 bases.

Value

An uppercase character string in reverse-complement orientation.

Examples

```
reverse_complement("ACGTRYN")
```

run_app — *Launch the offline SNPAssayR interface*

Description

Launch the offline SNPAssayR interface

Usage

```
run_app(port = NULL, launch.browser = interactive())
```

Arguments

`port`	NULL for an available port, or a whole-number port from 1024 to 65535.
`launch.browser`	Whether to open the local URL in a browser.

Value

The value returned invisibly by shiny::runApp() when the session ends.

Examples

```
if (interactive()) run_app()
```

screen_primer_pair — *Screen primer-pair products on a target and supplied related copies*

Description

Enumerates both assignments of the two primers to inward-facing plus/minus sites. Binding intervals must not overlap. Product length is outer end minus outer start plus one. Reports full-length untailed product geometry only; no product reconstruction, indels, genome search or amplification guarantee. Limits apply separately to each template; reaching max_products is conservatively flagged as truncated even if it could coincide with the last possible product.

Usage

```
screen_primer_pair(
  primer1,
  primer2,
  locus,
  max_mismatches = 0L,
  product_min = 1L,
  product_max = 10000L,
  max_products = 1000L,
  max_pair_checks = 100000L
)
```

Arguments

`primer1`, `primer2`
: Uppercase canonical DNA, 1 to 60 bases each.

`locus`
: A locus from `read_locus()`, including optional comparisons.

`max_mismatches`
: Maximum mismatches per site, no larger than either primer.

`product_min`, `product_max`
: Inclusive product-size bounds in bp, 1 to 10000.

`max_products`
: Maximum returned products per sequence, 1 to 10000.

`max_pair_checks`
: Maximum candidate site pairs examined per sequence, 1 to 1000000. Exhaustion is reported, never silently interpreted as absence.

Value

An S3 `snpassay_pair_screen` list with per-template sites/products, coverage, potential cross-amplification flags and separate specificity states.

Examples

```
x <- read_locus("ACGAAAAAGTC", "artificial", comparison_sequences = c(copy = "ACGAAAAAGTC"))
screen_primer_pair("ACG", "GAC", x)$potential_cross_amplification
```

`simulate_digest` *Simulate theoretical complete digestion of a canonical linear product*

Description

Retains all recognition sites and both strand cuts. Fragment lengths use the displayed plus-strand boundary convention, retaining multiplicity. No actual digestion efficiency, end-flank requirement, methylation or star activity is modeled. Ambiguous product bases are rejected rather than assumed digestible.

Usage

```
simulate_digest(sequence, enzyme)
```

Arguments

`sequence`
: Normalized uppercase canonical DNA, at most 10000 bases.

`enzyme`
: Curated enzyme name or one supported record data frame.

Value

An S3 `snpassay_digest` list with product, enzyme, sites, full fragments, exact size-only band set, verification status and model caveats.

Examples

```
simulate_digest("AGAATTCT", "EcoRI")$fragments
```

---

snpassay_app — *Create the offline SNPAssayR Shiny application*

Description

Shiny and bslib are optional dependencies required only for the interface. Installed local assets are used; no remote service, font or genome is fetched. Analyses delegate to the same four public design APIs as console workflows. Input and design edits invalidate prior results and downloads. Example presets include a synthetic 550 bp teaching locus and short geometry fixtures; none establishes laboratory performance.

Usage

```
snpassay_app()
```

Value

A Shiny application object; `run_app()` starts a local session.

Examples

```
if (interactive() && requireNamespace("shiny", quietly=TRUE) &&
    requireNamespace("bslib", quietly=TRUE)) snpassay_app()
```

---

validate_snp — *Validate one biallelic SNP in a locus*

Description

Validate one biallelic SNP in a locus

Usage

```
validate_snp(locus, position, ref, alt)
```

Arguments

`locus`	A locus returned by `read_locus()`.
`position`	One finite whole-number coordinate, one-based.
`ref`, `alt`	Distinct uppercase canonical DNA bases.

Value

A `snpassay_snp` list containing the locus, SNP and reconstructed reference/alternative sequences. This does not predict amplification.

Examples

```
validate_snp(read_locus("ACGT", "artificial_example"), 2, "C", "T")
```

Index

assay_bundle, 3
assay_bundle(), *16*, *23*
assay_preferences, 4
assay_preferences(), *3*, *21*
assay_scene, 5
assay_scene(), *15*
assay_tables, 5

band_scene, 6
band_scene(), *15*

compare_profiles, 7

design_arms, 7
design_arms(), *14*
design_caps, 9
design_dcaps, 11
design_tetra_arms, 13
digest_fragments, 14
draw_scene, 15

evaluate_primer, 15
evaluate_primer(), *8*, *10*, *12*
export_assay, 16
export_scene, 17

find_binding_sites, 18
find_hairpins, 18
fragment_profiles, 19
fragment_profiles(), *6*

group_bands, 20
group_bands(), *6*

primer_interactions, 20

rank_assays, 21
read_locus, 22
read_locus(), *25*, *27*, *28*
reconstruct_amplicon, 22
replay_assay, 23
restriction_enzymes, 24
restriction_enzymes(), *12*
restriction_sites, 24
reverse_complement, 25
run_app, 26
run_app(), *28*

screen_primer_pair, 26
simulate_digest, 27
simulate_digest(), *19*
snpassay_app, 28

validate_snp, 28
validate_snp(), *8*, *10*, *12*, *23*