Package: Rsassy (via r-universe)

May 31, 2026

Title 'R' Bindings to the 'sassy' Approximate String Matcher

Version 0.2.1-0.1.0.9000

Description Fast approximate string matching for short patterns in longer texts using the 'sassy' Rust crate. 'sassy' implements SIMD-accelerated fuzzy search over ASCII, DNA, and IUPAC alphabets, with support for reverse-complement search, overhang alignments, CIGAR strings, and batched searches. See Beeloo and Groot Koerkamp (2025) <doi:10.1101/2025.07.22.666207> and Beeloo and Groot Koerkamp (2026) <doi:10.64898/2026.03.10.710811>.

License GPL (>= 2)

Copyright See inst/AUTHORS and inst/LICENCE.note for bundled Rust crate authorship and licensing details.

Encoding UTF-8

SystemRequirements Cargo (Rust's package manager), rustc >= 1.91.0, xz

Suggests knitr, rmarkdown, tinytest

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

URL https://github.com/sounkou-bioinfo/Rsassy, https://sounkou-bioinfo.github.io/Rsassy/

BugReports https://github.com/sounkou-bioinfo/Rsassy/issues

Config/Needs/website pkgdown

Config/pak/sysreqs xz-utils libclang-dev

Repository https://sounkou-bioinfo.r-universe.dev

Date/Publication 2026-05-31 04:20:03 UTC

RemoteUrl https://github.com/sounkou-bioinfo/Rsassy

RemoteRef HEAD

RemoteSha 01a9181095ba98d9dc9824c18f477a58d7ea3817

Contents

`print.sassy_features` *Print Rsassy feature information*

Description

Print Rsassy feature information

Usage

```
## S3 method for class 'sassy_features'
print(x, ...)
```

Arguments

`x`	A `sassy_features` object returned by `sassy_features()`.
`...`	Ignored; accepted for compatibility with `print()`.

Value

`x`, invisibly.

print.sassy_matches *Print sassy match data frames*

Description

Print sassy match data frames

Usage

```
## S3 method for class 'sassy_matches'
print(x, ..., color = getOption("Rsassy.coloring", FALSE))
```

Arguments

x	A sassy_matches data frame.
...	Ignored; accepted for compatibility with print().
color	If TRUE, color match_region by CIGAR operation with ANSI escape sequences: green matches, orange substitutions, blue inserted text, and red gaps for pattern bases absent from the text. Defaults to getOption("Rsassy.coloring", FALSE).

Value

x, invisibly.

sassy_as_sam *Format matches in SAM-compatible text direction*

Description

Rsassy normally follows the upstream sassy TSV convention: reverse-strand match_region values are reverse-complemented and CIGAR strings are oriented in the input pattern direction. sassy_as_sam() converts reverse-strand rows to the text direction used by SAM and by upstream sassy --sam output.

Usage

```
sassy_as_sam(x, alphabet = "dna")
```

Arguments

x	A sassy_matches data frame.
alphabet	Alphabet profile used for the search. One of "dna" or "iupac" when x includes match_region.

Value

A copy of `x` with reverse-strand `cigar` values reversed and, when present, reverse-strand `match_region` values reverse-complemented back to text direction.

Examples

```
sassy_as_sam(
  sassy_search(list("ACGA"), list("TTTCGTTT"), 0, alphabet = "dna", match_region = TRUE),
  alphabet = "dna"
)
```

`sassy_crispr` *Search CRISPR guide targets*

Description

`sassy_crispr()` is an R-level equivalent of the upstream `sassy crispr` workflow for in-memory sequences. Guides include the PAM at the end. By default, the PAM must match exactly under IUPAC matching, while the rest of the guide may have up to `k` edits.

Usage

```
sassy_crispr(
  guide,
  text,
  k,
  pam_length = 3L,
  allow_pam_edits = FALSE,
  max_n_frac = 0.2,
  rc = TRUE,
  threads = 1L,
  pattern_id = NULL,
  text_id = NULL
)
```

Arguments

`guide`	List of guide sequences including the PAM suffix. Each element must be a raw vector or non-missing character scalar.
`text`	List of text sequences to search. Each element must be a raw vector or non-missing character scalar.
`k`	Maximum edit distance for the searched guide sequence. With `allow_pam_edits = FALSE`, the exact-PAM filter means this is effectively the edit threshold outside the PAM.
`pam_length`	Length of the PAM suffix.

allow_pam_edits
: If TRUE, do not require an exact PAM match.

max_n_frac
: Maximum allowed fraction of N bases in match_region.

rc
: If TRUE, search reverse-complement targets as well.

threads
: Number of worker threads to request.

pattern_id
: Optional guide/pattern identifiers. If supplied, must be a character vector with one entry per guide and adds/replaces a pattern_id column. Names on guide are not inspected.

text_id
: Optional text identifiers. If supplied, must be a character vector with one entry per text and adds/replaces a text_id column. Names on text are not inspected.

Value

A data frame with CLI-style columns: guide, cost, strand, start, end, match_region, and cigar. If pattern_id or text_id are supplied, mapped identifier columns are included.

Examples

```
sassy_crispr(list("ACGTNGG"), list("TTTACGTAGGTTT"), k = 0, rc = FALSE, text_id = "chr1")
```

---

sassy_fastx_iter *Create a chunked FASTA/FASTQ iterator*

Description

sassy_fastx_iter() opens a FASTA or FASTQ file and returns an iterator that yields record-count-bounded batches. Parsing is performed by the vendored Rust needletail parser. Sequence and quality data in each batch are exposed as read-only raw ALTREP slices over immutable native batch buffers; they are not eagerly materialized as R strings.

Usage

```
sassy_fastx_iter(path, batch_records = 100000L, include_qual = TRUE)
```

Arguments

path
: Path to a FASTA/FASTQ file. Gzip-compressed input is supported by the vendored needletail gzip backend.

batch_records
: Maximum number of records returned by each sassy_fastx_next() call.

include_qual
: If TRUE, FASTQ qualities are included as batch$qual. If FALSE, or for FASTA input, batch$qual is NULL.

Value

An external pointer with class sassy_fastx_iter.

Examples

```
fq <- tempfile(fileext = ".fastq")
writeLines(c("@r1", "ACGT", "+", "!!!!"), fq, useBytes = TRUE)
it <- sassy_fastx_iter(fq, batch_records = 1)
batch <- sassy_fastx_next(it)
rawToChar(batch$seq[[1]])
```

---

sassy_fastx_next *Get the next FASTA/FASTQ batch*

Description

Get the next FASTA/FASTQ batch

Usage

```
sassy_fastx_next(iter)
```

Arguments

iter	An iterator created by `sassy_fastx_iter()`.

Value

`NULL` at end of file, otherwise a `sassy_fastx_batch` list with `id`, `seq`, and `qual` elements. `id` is an ALTREP character vector, while `seq` and `qual` are ALTREP lists whose elements are raw ALTREP vectors.

Examples

```
fq <- tempfile(fileext = ".fastq")
writeLines(c("@r1", "ACGT", "+", "!!!!"), fq, useBytes = TRUE)
it <- sassy_fastx_iter(fq, batch_records = 1)
batch <- sassy_fastx_next(it)
length(batch$id)
```

sassy_features *Report Rsassy build and CPU feature information*

Description

Returns diagnostic information about the selected Rsassy backend. Calling this initializes the native backend if it has not already been loaded. `rsassy_selected_backend` reports the runtime-selected backend. `rsassy_installed_backends` is a character vector of backend libraries found in the package installation, and `rsassy_supported_backends` is the subset supported by the current CPU/runtime. With `"auto"` selection, Rsassy chooses the best supported installed backend: AVX-512 before AVX2 on x86_64, NEON on arm64, WebAssembly SIMD128 on wasm, and scalar otherwise. The `selected_*` fields describe the loaded Rust backend. The `cpu_*` fields are detected by the C shim.

Usage

```
sassy_features()
```

Value

A `sassy_features` list of build, selected-backend, and CPU/runtime feature values.

Examples

```
sassy_features()
```

sassy_search *Search approximate matches with 'sassy'*

Description

Convenience wrapper that creates a searcher, searches, and returns a `sassy_matches` data frame. Coordinates are 0-based and half-open.

Usage

```
sassy_search(
  pattern,
  text,
  k,
  alphabet = "dna",
  rc = TRUE,
  alpha = NULL,
  all = FALSE,
  threads = 1L,
  strategy = "pairwise",
```

```
  pattern_id = NULL,
  text_id = NULL,
  match_region = FALSE,
  sam = FALSE
)
```

Arguments

pattern	List of raw vectors or non-missing character scalars.
text	List of raw vectors or non-missing character scalars.
k	Maximum edit distance.
alphabet	Alphabet profile. One of "dna", "iupac", or "ascii".
rc	If TRUE, search reverse-complement strand as well where supported.
alpha	Optional IUPAC overhang cost in [0, 1]. Use NULL to disable.
all	If FALSE, return the usual local-minimum matches. If TRUE, return every end position with score <= k; this can include overlapping and nested candidate alignments and requires strategy = "pairwise".
threads	Number of worker threads to request for bulk searches.
strategy	Search strategy. "pairwise" searches each pattern/text pair independently and is the general default. "batch_texts" uses one text per SIMD lane. "batch_patterns" and "encoded_patterns" (alias "v2") use Sassy's multi-pattern encoding, which in sassy 0.2.1 is implemented for alphabet = "iupac" and equal byte-length patterns.
pattern_id	Optional pattern identifiers. If supplied, must be a non-missing character vector with one entry per pattern and adds/replaces a pattern_id column. Names on pattern are not inspected.
text_id	Optional text identifiers. If supplied, must be a non-missing character vector with one entry per text and adds/replaces a text_id column. Names on text are not inspected.
match_region	If TRUE, include a match_region column. Reverse-strand regions are reverse-complemented so the region and CIGAR are in the input pattern direction.
sam	If TRUE, format reverse-strand match_region and cigar in the text direction used by SAM and by the upstream sassy --sam output.

Value

A data frame with 0-based indices and coordinates: pattern_idx, text_idx, text_start, text_end, pattern_start, pattern_end, cost, strand, and cigar. If pattern_id or text_id are supplied, mapped identifier columns are included. If requested, also includes match_region. Rows are ordered by input text, then text start/end coordinate, then pattern index.

Examples

```
sassy_search(list("ACGT"), list("TTACGTAA"), 0, alphabet = "dna", rc = FALSE)
```

sassy_searcher *Create a reusable 'sassy' searcher*

Description

A searcher stores the selected alphabet profile and reverse-complement behavior. Reuse a searcher when searching many patterns or texts with the same settings.

Usage

```
sassy_searcher(alphabet = "dna", rc = TRUE, alpha = NULL)
```

Arguments

alphabet	Alphabet profile. One of "dna", "iupac", or "ascii".
rc	If TRUE, search reverse-complement strand as well where supported.
alpha	Optional IUPAC overhang cost in [0, 1]. Use NULL to disable.

Value

An external pointer with class sassy_searcher.

Examples

```
searcher <- sassy_searcher("dna", rc = FALSE)
sassy_searcher_search(searcher, list("ACGT"), list("TTACGTAA"), 0)
```

sassy_searcher_search *Search with a reusable 'sassy' searcher*

Description

pattern and text must be lists of sequences. Each element must be a raw vector or a non-missing character scalar. Every pattern is searched against every text and the returned pattern_idx and text_idx columns identify the 0-based input indices. Use threads > 1 for larger batches.

Usage

```
sassy_searcher_search(
  searcher,
  pattern,
  text,
  k,
  all = FALSE,
  threads = 1L,
```

```
  strategy = "pairwise",
  pattern_id = NULL,
  text_id = NULL,
  match_region = FALSE,
  sam = FALSE
)
```

Arguments

searcher	A searcher created by sassy_searcher().
pattern	List of raw vectors or non-missing character scalars.
text	List of raw vectors or non-missing character scalars.
k	Maximum edit distance.
all	If FALSE, return the usual local-minimum matches. If TRUE, return every end position with score <= k; this can include overlapping and nested candidate alignments and requires strategy = "pairwise".
threads	Number of worker threads to request for bulk searches.
strategy	Search strategy. "pairwise" searches each pattern/text pair independently and is the general default. "batch_texts" uses one text per SIMD lane. "batch_patterns" and "encoded_patterns" (alias "v2") use Sassy's multi-pattern encoding, which in sassy 0.2.1 is implemented for alphabet = "iupac" and equal byte-length patterns.
pattern_id	Optional pattern identifiers. If supplied, must be a non-missing character vector with one entry per pattern and adds/replaces a pattern_id column. Names on pattern are not inspected.
text_id	Optional text identifiers. If supplied, must be a non-missing character vector with one entry per text and adds/replaces a text_id column. Names on text are not inspected.
match_region	If TRUE, include a match_region column. Reverse-strand regions are reverse-complemented so the region and CIGAR are in the input pattern direction.
sam	If TRUE, format reverse-strand match_region and cigar in the text direction used by SAM and by the upstream sassy --sam output.

Value

A data frame with 0-based indices and coordinates: pattern_idx, text_idx, text_start, text_end, pattern_start, pattern_end, cost, strand, and cigar. If pattern_id or text_id are supplied, mapped identifier columns are included. If requested, also includes match_region. Rows are ordered by input text, then text start/end coordinate, then pattern index.

`sassy_set_backend` *Select the Rsassy native backend*

Description

Select a backend for the current R process. Backend loading is intentionally one-shot: the selected shared library is fixed for the lifetime of the R process. This must be called before the first native Rsassy operation, including `sassy_features()`, `sassy_searcher()`, or `sassy_search()`. Rsassy does not unload and replace backend DLLs because that is not reliable across R platforms. Use this for benchmarking installed backends against each other in separate fresh R processes.

Usage

```
sassy_set_backend(
  backend = c("auto", "scalar", "avx2", "avx512", "neon", "wasm_simd128")
)
```

Arguments

`backend`	One of `"auto"`, `"scalar"`, `"avx2"`, `"avx512"`, `"neon"`, or `"wasm_simd128"`.

Value

The requested backend name, invisibly. `"auto"` means runtime dispatch will choose the best installed backend supported by the current CPU/runtime when the backend is first loaded.

Index