Package: Rminibwa (via r-universe)

June 23, 2026

Title Native R Bindings and SIMD Dispatch for 'minibwa'

Version 0.0.0.9000

Description Provides native R bindings to the 'minibwa' genomic read aligner with raw-vector query input, external-pointer alignment batches, ALTREP column views, and an installed C API for downstream packages. The package builds staged Single Instruction Multiple Data ('SIMD') KSW backends and selects portable scalar, SSE4, or AVX2 code at runtime without global instruction-set compiler flags.

License GPL (>= 2)

Depends R (>= 4.0.0)

SystemRequirements GNU make, zlib

LinkingTo RsimdDispatch (>= 0.1.2.9001)

Suggests knitr, rmarkdown, roxygen2, Rtinycc, tinytest

VignetteBuilder knitr

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

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

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

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

Config/pak/sysreqs make zlib1g-dev

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

Date/Publication 2026-06-23 08:17:18 UTC

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

RemoteRef HEAD

RemoteSha 2e418fb29315b322eaa32316a1dede05db3022d6

Contents

`mb_align_n` *Inspect native minibwa alignment batches*

Description

`mb_align_col()` returns ALTREP views over native alignment columns. These views do not copy columns unless R forces materialization. C consumers should prefer the installed `Rminibwa.h` API.

Usage

```
mb_align_n(x)

mb_align_col(x, name)

mb_align_cigar_words(x)
```

Arguments

`x`	Native alignment batch returned by `mb_map()`.
`name`	Column name.

Value

`mb_align_n()` returns the number of records. `mb_align_col()` returns an ALTREP integer or real vector. `mb_align_cigar_words()` returns a raw debug copy of packed minibwa CIGAR words.

mb_index_build *Build a minibwa index with the native library*

Description

Builds index files at `prefix` using the vendored minibwa C library. The package is GPL, so the native build includes minibwa's GPL-compatible low-memory indexing path; `low_memory = FALSE` uses the default `libsais` path.

Usage

```
mb_index_build(fasta, prefix, meth = FALSE, threads = 1L, low_memory = FALSE)
```

Arguments

`fasta`	Reference FASTA path. Gzip-compressed input is supported by minibwa/zlib.
`prefix`	Output prefix for `.l2b`, `.mbw`, and optionally `.meth.mbw`.
`meth`	Build the directional methylation index as well.
`threads`	Number of index-build threads requested by minibwa.
`low_memory`	Use minibwa's low-memory BWT builder.

Value

`prefix`, invisibly.

mb_index_contigs *Return minibwa index contigs*

Description

Names are returned as raw byte vectors to avoid forced string materialization.

Usage

```
mb_index_contigs(index)
```

Arguments

`index`	Index handle returned by `mb_index_load()`.

Value

A list with raw-vector `name` entries and numeric `length` values.

mb_index_load *Load a native minibwa index*

Description

Load a native minibwa index

Usage

```
mb_index_load(prefix, meth = FALSE)
```

Arguments

prefix	Index prefix.
meth	Load the methylation BWT index.

Value

An external pointer handle to a native minibwa index.

mb_map *Map raw sequence bytes with native minibwa*

Description

`mb_map()` is pipe-friendly: the sequence bytes are the first argument. The result is a native alignment batch external pointer. Use `mb_align_col()` for ALTREP column views or pass the batch to downstream C consumers.

Usage

```
mb_map(x, index, opt = mb_opts(), name = NULL, meth = c("none", "c2t", "g2a"))
```

Arguments

x	Raw vector of query sequence bytes.
index	Index handle returned by `mb_index_load()`.
opt	Options list from `mb_opts()`.
name	Optional read name as raw bytes. Character names are accepted for convenience but raw bytes are preferred.
meth	Methylation code: `"none"`, `"c2t"`, or `"g2a"`.

Value

A native alignment batch external pointer.

Examples

```
# See vignettes for full index-backed examples.
```

`mb_opts` *Native minibwa options*

Description

Returns a plain list consumed by native mapping functions. Only the selected options are exposed initially; all other fields use minibwa defaults for the chosen preset.

Usage

```
mb_opts(
  preset = c("adap", "sr", "lr"),
  paired = NULL,
  methylation = NULL,
  out_n = NULL,
  min_seed_len = NULL,
  threads = NULL,
  match_score = NULL,
  mismatch_penalty = NULL,
  gap_open = NULL,
  gap_extend = NULL
)
```

Arguments

`preset`	One of `"adap"`, `"sr"`, or `"lr"`.
`paired`, `methylation`	Optional logical overrides for minibwa flags.
`out_n`, `min_seed_len`, `threads`, `match_score`, `mismatch_penalty`, `gap_open`, `gap_extend`	Optional integer overrides for common minibwa fields.

Value

A plain list.

minibwa_available *Test whether the minibwa CLI is available*

Description

Test whether the minibwa CLI is available

Usage

```
minibwa_available(path = NULL)
```

Arguments

path	Optional executable name or path. When NULL, use Rminibwa's packaged minibwa executable.

Value

TRUE when an executable can be resolved, otherwise FALSE.

minibwa_cli *Run the minibwa CLI*

Description

Low-level helper for invoking arbitrary minibwa commands from R. Prefer minibwa_index() and minibwa_map() for common workflows.

Usage

```
minibwa_cli(args = character(), stdout = TRUE, path = minibwa_path())
```

Arguments

args	Character vector of command-line arguments passed to minibwa.
stdout	Passed to system2(). Use TRUE to capture standard output, FALSE to inherit it, or a file path to redirect it.
path	Executable name or path.

Value

If stdout = TRUE, a character vector of captured output; otherwise an invisible process status.

`minibwa_index` *Build a minibwa index through the CLI*

Description

Build a minibwa index through the CLI

Usage

```
minibwa_index(
  reference,
  prefix = NULL,
  threads = NULL,
  meth = FALSE,
  low_memory = FALSE,
  extra_args = character(),
  path = minibwa_path()
)
```

Arguments

`reference`	Path to a FASTA reference.
`prefix`	Optional output prefix. When `NULL`, minibwa uses `reference` as the prefix.
`threads`	Optional number of worker threads.
`meth`	If `TRUE`, build a directional bisulfite sequencing index via `--meth`.
`low_memory`	If `TRUE`, pass `-l` to minibwa index.
`extra_args`	Additional CLI arguments inserted before positional arguments.
`path`	Executable name or path.

Value

Invisibly, the expected index prefix.

`minibwa_map` *Map reads through the minibwa CLI*

Description

Map reads through the minibwa CLI

Usage

```
minibwa_map(
  index,
  reads,
  output = NULL,
  format = c("sam", "paf"),
  threads = NULL,
  meth = FALSE,
  hic = FALSE,
  extra_args = character(),
  path = minibwa_path()
)
```

Arguments

index	Index prefix produced by `minibwa_index()` or the minibwa CLI.
reads	One or more FASTA/FASTQ input paths. The common cases are one single-end file or two paired-end files.
output	Optional output file. When `NULL`, standard output is captured and returned as a character vector.
format	Output format: `"sam"` by default, or `"paf"` to pass `-f`.
threads	Optional number of worker threads.
meth	If `TRUE`, pass `--meth`.
hic	If `TRUE`, pass `--hic`.
extra_args	Additional CLI arguments inserted before positional arguments.
path	Executable name or path.

Value

If `output = NULL`, captured alignment records as a character vector; otherwise the output path, invisibly.

---

minibwa_path *Locate the packaged minibwa executable*

Description

`minibwa_path()` resolves the command-line executable used by the CLI-backed helpers. By default it returns the executable built from the vendored minibwa sources and installed with Rminibwa. Supplying `path` is mainly for developer comparisons against another minibwa build.

Usage

```
minibwa_path(path = NULL, must_work = TRUE)
```

Arguments

path	Optional executable name or path. When NULL, use Rminibwa's packaged minibwa executable.
must_work	If TRUE, throw an error when the executable cannot be found.

Value

A scalar character path, or NA_character_ when must_work = FALSE and no executable is found.

---

minibwa_upstream_info *Report the pinned upstream minibwa source*

Description

Rminibwa pins a specific upstream minibwa commit for native-library work. minibwa_upstream_info() reports the installed copy of that provenance metadata.

Usage

```
minibwa_upstream_info()
```

Value

A named list with fields such as Component, Version, Repository, Commit, and PatchDirectory.

Examples

```
minibwa_upstream_info()[c("Version", "Commit")]
```

---

minibwa_version *Report the minibwa CLI version*

Description

Report the minibwa CLI version

Usage

```
minibwa_version(path = minibwa_path())
```

Arguments

path	Optional executable name or path. When NULL, use Rminibwa's packaged minibwa executable.

Value

A scalar character version string.

simd_backend *Report the selected SIMD backend*

Description

Report the selected SIMD backend

Usage

```
simd_backend()
```

Value

A character scalar.

simd_info *Report SIMD dispatch diagnostics*

Description

Report SIMD dispatch diagnostics

Usage

```
simd_info()
```

Value

A named list with dispatch mode, requested backend, selected backend, compiled backends, CPU-supported backends, available backends, and target.

simd_set_backend *Select the runtime SIMD backend*

Description

Rminibwa compiles staged KSW backend objects and switches among them at runtime. Use `"auto"` for the best available backend, or force one of `"scalar"`, `"sse4"`, or `"avx2"` for diagnostics and benchmarks. The `"scalar"` backend is the portable SIMDe baseline with native intrinsics disabled; compilers may still lower portable vector types efficiently.

Usage

```
simd_set_backend(backend = "auto")
```

Arguments

backend	Character scalar backend name.

Value

The selected backend, invisibly.

Index