

Package: genoaligner (via r-universe)

October 11, 2026

Type Package

Title GPU-Portable Pairwise and Codon-Aware Multiple Sequence Alignment

Version 2.0.0.9000

Description Sequence alignment from one portable C++17 core: edit-distance (Levenshtein / WFA-equivalent) and Smith-Waterman local pairwise alignment with score and CIGAR reconstruction, plus multiple sequence alignment in DNA, protein and codon-aware modes (MACSE-class: whole-codon indels that preserve the reading frame, optional frameshift refinement). The core builds and runs anywhere (it is the CPU backend, so it needs no GPU toolchain); the wavefront/GPU backend (ROCm/CUDA) computes the same results with acceleration and is tracked in the sibling C++ repository. A batch API aligns many pairs in one call and reports how many were resolved, so silent under-serving is impossible. Designed to slot into data.frame/tibble pipelines.

URL <https://github.com/alroble/genoaligner-r>,
<https://alroble.github.io/genoaligner-r/>

BugReports <https://github.com/alroble/genoaligner-r/issues>

License MIT + file LICENSE

Encoding UTF-8

Language en-US

Depends R (>= 4.1.0)

Imports Rcpp

LinkingTo Rcpp

Suggests testthat (>= 3.0.0), knitr, rmarkdown, spelling, ape

VignetteBuilder knitr

SystemRequirements C++17

NeedsCompilation yes

Config/testthat/edition 3
Roxygen list(markdown = TRUE)
Config/roxygen2/version 8.0.0
LazyData true
Repository https://alrobls.r-universe.dev
Date/Publication 2026-09-28 13:25:40 UTC
RemoteUrl https://github.com/alrobls/genoaligner-r
RemoteRef HEAD
RemoteSha d7a0767e8605f0537deb2b15a683bc5ae49d22e3

Contents

genoaligner-package	2
align_edit	3
align_sw	4
as_dnabin	5
lepus_cytb	5
msa_align	6
msa_codon	7
Index	10

genoaligner-package	<i>genoaligner: GPU-portable pairwise sequence alignment</i>
---------------------	--

Description

Pairwise sequence alignment from one portable C++17 core: edit distance (Levenshtein / WFA-equivalent) and Smith-Waterman local alignment, both with score and CIGAR reconstruction. The core builds and runs anywhere (it is the CPU backend, so it needs no GPU toolchain); the sibling C++ library tracks the GPU (ROCm/CUDA) backend that computes the same results with wavefront acceleration.

Details

The two entry points are `align` (edit distance bounded by `smax`) and `align_sw` (local Smith-Waterman with affine gaps). Both are vectorised over pairs, so a two-column pipeline (query, reference) maps straight to a `data.frame` of scores and CIGARs.

Author(s)

Maintainer: Angel Robles-Fernandez <a.l.robles.fernandez@gmail.com>
Authors:
• Angel Robles-Fernandez <a.l.robles.fernandez@gmail.com>

See Also

Useful links:

- <https://github.com/alrobls/genoaligner-r>
- <https://alrobls.github.io/genoaligner-r/>
- Report bugs at <https://github.com/alrobls/genoaligner-r/issues>

align_edit

Align pairs by edit distance

Description

Computes the Levenshtein edit distance (substitution = 1, insertion = 1, deletion = 1) between each pair of sequences, with an optional CIGAR reconstruction. This is the same distance the WFA formulation computes; wavefront/GPU speed is a backend concern, the result is identical.

Usage

```
align_edit(pattern, text, smax = 64, with_cigar = TRUE)
```

Arguments

pattern	Character vector; query sequences (recycled to match text).
text	Character vector; reference sequences (recycled to match pattern). NA entries produce NA rows.
smax	Numeric; maximum edit distance searched, in $[0, 511]$. Pairs whose true distance exceeds it are unresolved by design.
with_cigar	Logical; if FALSE only scores are computed (cheaper, no traceback) and cigar is NA.

Details

A pair is *resolved* only when its true distance is at most *smax*; otherwise score is NA and the pair is reported as *resolved* = FALSE. The count of resolved pairs is deliberately visible: a caller that only reads scores can silently miss that most input was abandoned when *smax* is set too tightly.

Value

A data.frame with one row per pair and columns: score (edit distance, NA if unresolved), resolved (logical), cigar (over {M, X, I, D}, NA if unresolved or !with_cigar), rescore_ok and wellformed_ok (validation flags computed in the library).

CIGAR convention (identical for align_edit and align_sw): M/X consume a text and a pattern base; I consumes an extra *text* base; D consumes an extra *pattern* base.

Examples

```
align_edit("ACGTACGT", "ACGTTCGT", smax = 8)
align_edit(c("AAAACCC", "ACGT"), c("AAAATCC", "ACCT"), smax = 8)
```

align_sw	<i>Smith-Waterman local alignment</i>
----------	---------------------------------------

Description

Local alignment with affine gap penalties {match, mismatch, gap_open, gap_extend}. Penalties are positive and subtracted. A pair with score == 0 has no positive-scoring local alignment (disjoint sequences); all four coordinates are -1 and cigar is empty.

Usage

```
align_sw(text, pattern, scoring = c(2L, -3L, 5L, 2L), with_cigar = TRUE)
```

Arguments

text	Character vector; row-axis sequences (recycled).
pattern	Character vector; column-axis sequences (recycled).
scoring	Numeric vector of length 4, named or positional: (match, mismatch, gap_open, gap_extend). Recycled if a single vector is given. The defaults are an example, not a recommendation. Degenerate schemes are refused, matching the C++ API: match <= 0, mismatch >= match, gap_open <= 0, gap_extend <= 0, and gap_extend > gap_open when with_cigar (the emitted CIGAR could not reproduce its own score under that regime; score-only calls still accept it).
with_cigar	Logical; if FALSE, only score and end coordinates are computed (start coordinates and cigar are NA/-1).

Value

A data.frame with one row per pair and columns: score, start_i, start_j, end_i, end_j (0-based coordinates of the aligned span, -1 when score is 0/unknown), cigar (over the span), rescore_ok, wellformed_ok and too_large (the pair's matrix exceeded the supported size limit).

Examples

```
align_sw("TTTACGTGTT", "ACGTGT", scoring = c(2, -3, 5, 2))
align_sw(c("ACGTACGT", "GGGG"), c("ACGTTCGT", "ACGT"), c(2, -3, 5, 2))
```

as_dnabin	<i>Coerce a genoaligner_msa alignment to ape::DNAbin</i>
-----------	--

Description

Requires the **ape** package (in Suggests; used here only).

Usage

```
as_dnabin(x)
```

Arguments

x A genoaligner_msa object.

Value

An ape::DNAbin matrix.

Examples

```
if (requireNamespace("ape", quietly = TRUE)) {
  as_dnabin(msa_align(c("ACGTACGT", "ACGTCGT")))
}
```

lepus_cytb	<i>Cytochrome b sequences from four leporid species</i>
------------	---

Description

Real mitochondrial cytochrome b (cytb) records downloaded from NCBI nuccore for three hare species (*Lepus granatensis*, *L. timidus*, *L. europaeus*) — all flagged as population-genetic candidates in the EcoSeek genominer database (<https://genominer.ecoseek.org>) — plus the European rabbit (*Oryctolagus cuniculus*) as an outgroup.

Usage

```
lepus_cytb
```

Format

A data.frame with 37 rows and 4 columns:

accession NCBI nuccore accession (character).

species Species epithet: granatensis, timidus, europaeus or cuniculus (character).

title Full NCBI FASTA header (character).

sequence Nucleotide sequence as downloaded, untrimmed (character, 400–1200 bp; MN098958.1 contains one IUPAC ambiguity code, W).

Details

Sequences are shipped exactly as retrieved: mixed lengths, untrimmed, and including two mislabelled records (XM_017337704.3 and XM_051846530.2, predicted nuclear mRNAs that a "cytochrome b" title search returns but that are not the mitochondrial gene). The lepus-cytb vignette uses them to show that a near-zero local alignment score flags a mislabelled record.

Source

NCBI nuccore, downloaded 2026-09-24; see data-raw/lepus_cytb.R in the source repository for the exact eutils queries and snapshot FASTAs.

Examples

```
data(lepus_cytb)
table(lepus_cytb$species)
align_edit(lepus_cytb$sequence[1:3], lepus_cytb$sequence[2], smax = 200)
```

msa_align	<i>Multiple sequence alignment (DNA or protein)</i>
-----------	---

Description

Progressive MSA: fragment-corrected k-mer distances, a deterministic neighbor-joining guide tree, post-order profile-vs-profile semiglobal alignment (free end gaps, position-specific gap penalties). The pipeline is deterministic: identical input always produces identical output.

Usage

```
msa_align(
  seqs,
  names = NULL,
  mode = c("dna", "protein"),
  guide = "nj",
  iter_refine = 0L,
  fft_band = 0L,
  fft_lags = 4L,
  fft_min_rel = 0.1
)
```

Arguments

seqs	Character vector; unaligned sequences (no "-"). IUPAC nucleotide letters in dna mode; amino-acid letters in protein mode.
names	Optional character vector of sequence names, same length as seqs; defaults to names(seqs) or generated ids. Used for row names of the aligned matrix and FASTA output.
mode	"dna" (default) or "protein".

guide	Guide-tree construction; only "nj" (deterministic neighbor joining) exists today.
iter_refine	Integer ≥ 0 ; iterative-refinement rounds over the guide-tree bipartitions (MAFFT FFT-NS-i class): every tree edge splits the alignment into two induced sub-profiles, the pair is realigned, and the candidate is kept only if the sum-of-pairs score strictly improves. 0 (default) keeps the plain progressive alignment. Two rounds is a good first value; the pass converges and a further round reports no accepted changes.
fft_band	Integer ≥ 0 ; FFT anchor band half-width in columns. When > 0 (and iter_refine > 0) the engine detects homology anchors by Fourier cross-correlation of the two profile property signals and runs the realignment DP inside a band around them — the literal FFT of MAFFT FFT-NS-i. Weak anchor evidence falls back to the full DP, and the strict-improvement gate applies either way, so this is a speed/robustness knob, not a quality risk. 0 (default) uses the full matrix.
fft_lags	Integer ≥ 1 ; number of anchor diagonals whose bands are unioned (default 4). Only used when fft_band > 0 .
fft_min_rel	Numeric ≥ 0 ; minimum cosine similarity of the best anchor before the band is trusted (default 0.10). Higher values make the engine fall back to the full DP more often.

Value

An object of class `genoaligner_msa` with components `aligned` (character matrix, one row per input sequence, one column per alignment position), `qc` (per-sequence data.frame; all NA outside codon mode) and `params` (echo of the effective request). See [msa_codon](#) for the codon-aware mode.

Examples

```
msa_align(c("ACGTACGTAA", "ACGTTTCGT", "ACGTACG"))
msa_align(c(insulin = "MALWMRLLPLL", insulin2 = "MALWTRLPL"),
          mode = "protein")
```

msa_codon

Codon-aware multiple sequence alignment

Description

Codon-aware MSA of coding sequences: each sequence is tokenized to codons (NCBI genetic-code table `gc`) and aligned over a codon substitution matrix (BLOSUM62 on the translated amino acids plus a per-identical-nucleotide bonus and a stop-codon penalty). With the default `refine = FALSE` every indel is a whole codon, so every aligned row stays a multiple of 3 and the reading frame is preserved by construction.

Usage

```
msa_codon(
  seqs,
  names = NULL,
  gc = 1L,
  refine = FALSE,
  iter_refine = 0L,
  fft_band = 0L,
  fft_lags = 4L,
  fft_min_rel = 0.1
)
```

Arguments

seqs	Character vector; unaligned sequences (no "-"). IUPAC nucleotide letters in dna mode; amino-acid letters in protein mode.
names	Optional character vector of sequence names, same length as seqs; defaults to names(seqs) or generated ids. Used for row names of the aligned matrix and FASTA output.
gc	Integer; NCBI genetic code: 1L (standard, default) or 2L (vertebrate mitochondrial — the right table for CYTB/COI/ND).
refine	Logical; run the stage-2 frameshift refinement pass.
iter_refine	Integer ≥ 0 ; iterative-refinement rounds over the guide-tree bipartitions (MAFFT FFT-NS-i class): every tree edge splits the alignment into two induced sub-profiles, the pair is realigned, and the candidate is kept only if the sum-of-pairs score strictly improves. 0 (default) keeps the plain progressive alignment. Two rounds is a good first value; the pass converges and a further round reports no accepted changes.
fft_band	Integer ≥ 0 ; FFT anchor band half-width in columns. When > 0 (and <code>iter_refine</code> > 0) the engine detects homology anchors by Fourier cross-correlation of the two profile property signals and runs the realignment DP inside a band around them — the literal FFT of MAFFT FFT-NS-i. Weak anchor evidence falls back to the full DP, and the strict-improvement gate applies either way, so this is a speed/robustness knob, not a quality risk. 0 (default) uses the full matrix.
fft_lags	Integer ≥ 1 ; number of anchor diagonals whose bands are unioned (default 4). Only used when <code>fft_band</code> > 0 .
fft_min_rel	Numeric ≥ 0 ; minimum cosine similarity of the best anchor before the band is trusted (default 0.10). Higher values make the engine fall back to the full DP more often.

Details

With `refine = TRUE` a MACSE-style stage-2 pass reintroduces internal 1-2 nt frameshift events as narrow columns, so the output is a nucleotide MSA no longer guaranteed to be a multiple of 3 — that is the intended semantics for sequences that truly carry frameshifts.

Value

A `genoaligner_msa` object. In codon mode `qc` carries measured values per input sequence: `frame` (chosen reading frame, 0-2), `n_stops` (stop codons in the tokenized stream) and `partial` (partial/ambiguous codons and held-out frameshift blocks). A clean coding sequence reports `n_stops = 0`.

Examples

```
msa_codon(c("ATGATAATCACC", "ATGATTATCACCTGA"))  
msa_codon(c("ATGATAATCACC", "ATGATTATCACCTGA"), gc = 2)
```

Index

* **datasets**

lepus_cytb, [5](#)

align, [2](#)

align_edit, [3](#)

align_sw, [2](#), [4](#)

as_dnabin, [5](#)

genoaligner (genoaligner-package), [2](#)

genoaligner-package, [2](#)

lepus_cytb, [5](#)

msa_align, [6](#)

msa_codon, [7](#), [7](#)