Package: RsimdDispatch (via r-universe)

June 1, 2026

Title Runtime 'SIMD' Dispatch Templates for 'C' Code in 'R' Packages

Version 0.1.1

Description Provides templates and a working example for runtime Single Instruction Multiple Data ('SIMD') dispatch in 'C' code used by 'R' packages. Packages can stage scalar and architecture-specific kernel objects during configuration, then select a compiled and CPU-supported implementation at runtime through guarded function pointers. The package also vendors the header-only 'SIMDe' library for downstream packages through the 'LinkingTo' field.

License GPL (>= 2)

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

SystemRequirements GNU make

Suggests bench, knitr, rmarkdown, tinytest

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.3

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

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

NeedsCompilation yes

Author Sounkou Mahamane Toure [aut, cre], Evan Nemerson [cph] (SIMDe copyright holder), SIMDe contributors [ctb] (Contributors to the bundled SIMDe header library)

Maintainer Sounkou Mahamane Toure <sounkoutoure@gmail.com>

Config/pak/sysreqs make

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

Date/Publication 2026-06-01 14:30:02 UTC

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

RemoteRef HEAD

RemoteSha cdb9200069aafaac091915bfe6b2040babab1ba8

Contents

`count_nonzero` *Count non-zero bytes with the selected SIMD backend*

Description

Demonstration kernel for the runtime dispatch template. `count_nonzero()` counts bytes that are not `00` in a raw vector using the currently selected backend. The default backend is `"auto"`, which selects the best compiled backend supported by the current CPU/runtime.

Usage

```
count_nonzero(x)
```

Arguments

x	A raw vector.

Value

A numeric scalar count.

Examples

```
count_nonzero(as.raw(c(0, 1, 0, 2)))
```

simd_backend *Report the currently selected SIMD backend*

Description

Report the currently selected SIMD backend

Usage

```
simd_backend()
```

Value

A character scalar naming the selected backend.

Examples

```
simd_backend()
```

simd_dispatch_template_path *Configure an R package for C runtime SIMD dispatch*

Description

`use_simd_dispatch()` copies the dispatch scaffold into an R package and performs the package-name and C-prefix substitutions needed for a working package. It writes package files, updates `DESCRIPTION`, `.Rbuildignore`, and `.gitignore`, and returns the copied paths invisibly.

Usage

```
simd_dispatch_template_path()

use_simd_dispatch(
  path = ".",
  pkg = NULL,
  prefix = NULL,
  overwrite = FALSE,
  quiet = FALSE
)
```

Arguments

path	Package root where the template should be copied.
pkg	R package name. If NULL, the name is read from DESCRIPTION.
prefix	C symbol prefix used to replace rsd_ in the copied sources. The default is a sanitized lowercase package name.
overwrite	Whether to overwrite existing files.
quiet	Whether to suppress progress messages.

Value

Invisibly returns copied file paths.

Developer utility

This function is intended for package authors. It is not needed at runtime by users of packages that already include generated dispatch code.

Examples

```
simd_dispatch_template_path()
```

---

simd_info *Report runtime SIMD dispatch diagnostics*

Description

Returns the requested backend, selected backend, compiled backends, CPU-supported backends, SIMDe-native backends, target information, and SIMDe provenance compiled into the shared library. Calling this initializes the lazy auto-dispatch selection if it has not already been initialized.

Usage

```
simd_info()
```

Value

A named list of dispatch and CPU feature diagnostics. Backend-set entries are character vectors, not comma-separated strings.

Examples

```
names(simd_info())
```

simd_set_backend *Select the runtime SIMD backend*

Description

Select the backend used by subsequent calls to `count_nonzero()`. `RsimdDispatch` keeps all compiled variants in one shared object and switches guarded function pointers. This makes same-process benchmarking possible.

Usage

```
simd_set_backend(
  backend = c("auto", "scalar", "sse2", "sse41", "avx2", "avx512", "neon")
)
```

Arguments

`backend`	One of `"auto"`, `"scalar"`, `"sse2"`, `"sse41"`, `"avx2"`, `"avx512"`, or `"neon"`.

Value

The selected backend, invisibly. For `"auto"`, this is the backend chosen from the compiled and CPU-supported set.

Examples

```
old <- simd_backend()
simd_set_backend("scalar")
simd_set_backend("auto")
```

simde_info *Report vendored SIMDe provenance*

Description

`simde_info()` reports the version, upstream repository, pinned commit, and commit date for the bundled header-only SIMDe library.

Usage

```
simde_info()
```

Value

A named list of character scalars describing the vendored SIMDe copy.

Examples

```
simde_info()[c("version", "commit")]
```

Index