Package 'ucminfcpp'

May 8, 2026

Type Package

Title 'C++' Reimplementation of the 'ucminf' Unconstrained Nonlinear Optimizer

Version 1.0.0

Description A modern 'C++17/ reimplementation of the 'UCMINF/ algorithm for unconstrained nonlinear optimization (Nielsen and Mortensen, 2011, <doi:10.32614/CRAN.package.ucminf>), offering full API compatibility with the original 'ucminf' R package but developed independently. The optimizer core has been rewritten in 'C' with a modern header-only 'C++17' interface, zero-allocation line search, and an 'Rcpp' interface. The goal is numerical equivalence with improved performance, reproducibility, and extensibility. Includes extensive test coverage, performance regression tests, and compatibility checks against 'ucminf'. This package is not affiliated with the original maintainers but acknowledges their authorship of the algorithm and the original R interface.

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 3.5.0)

Imports Rcpp

LinkingTo Rcpp (>= 1.1.0)

Suggests testthat (>= 3.0.0), ucminf, microbenchmark

Config/testthat/edition 3

URL https://github.com/alrobles/ucminfcpp, https://alrobles.github.io/ucminfcpp/

BugReports https://github.com/alrobles/ucminfcpp/issues

RoxygenNote 7.3.3

NeedsCompilation yes

Author Angel Luis Robles Fernández [aut, cre],
Hans Bruun Nielsen [cph] (Original 'UCMINF' algorithm ('Fortran')),
Rex Brown [ctb] (Contributor to original R package 'ucminf'),
K Hervé Dakpo [ctb] (Contributor to original R package 'ucminf')

Maintainer Angel Luis Robles Fernández <a.l.robles.fernandez@gmail.com>

Repository CRAN

Date/Publication 2026-04-22 09:00:02 UTC

Contents

`print.ucminf` *Print method for ucminf objects*

Description

Print method for ucminf objects

Usage

```
## S3 method for class 'ucminf'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x`	A `ucminf` object.
`digits`	Number of significant digits to print.
`...`	Unused.

Value

The input object `x` is returned invisibly (called for side effects).

`ucminf` *General-Purpose Unconstrained Non-Linear Optimization (C/Rcpp)*

Description

An implementation of the UCMINF algorithm translated from Fortran to C and wrapped with Rcpp. The algorithm uses a quasi-Newton method with BFGS updating of the inverse Hessian and a soft line search with trust-region radius monitoring.

Usage

```
ucminf(
  par,
  fn = NULL,
  gr = NULL,
  ...,
  fdfun = NULL,
  control = list(),
  hessian = 0
)
```

Arguments

`par`	Initial estimate of the minimum (numeric vector).
`fn`	Objective function to be minimized. Must return a scalar. Ignored when `fdfun` is supplied.
`gr`	Gradient function. If `NULL` a finite-difference approximation is used. Ignored when `fdfun` is supplied.
`...`	Optional arguments passed to `fn` and `gr`.
`fdfun`	Optional combined value+gradient function `fdfun(x)` returning `list(f = scalar, g = numeric_vector)`. When supplied, `fn` and `gr` are ignored.
`control`	A list of control parameters (see Details), or a `ucminf_control` object.
`hessian`	Integer controlling Hessian output: **0** No Hessian (default). **2** Returns the final inverse-Hessian approximation from BFGS. **3** Returns both the inverse Hessian (2) and its inverse (Hessian).

Details

This package is a two-phase migration of the original **ucminf** R package:

- **Phase 1**: The Fortran core is translated to C (`src/ucminf_core.c`).
- **Phase 2**: The C core is wrapped with an Rcpp interface (`src/ucminf_rcpp.cpp`).

The interface is designed to be interchangeable with the original `ucminf::ucminf` and with `optim`.

The `control` argument accepts:

`trace` If positive, print convergence info after optimization. Default `0`.

`grtol` Stop when $\|g(x)\|_\infty \leq$ `grtol`. Default `1e-6`.

`xtol` Stop when $\|x - x_{\text{prev}}\|^2 \leq$ `xtol`*(`xtol` + $\|x\|^2$). Default `1e-12`.

`stepmax` Initial trust-region radius. Default `1`.

`maxeval` Maximum function evaluations. Default `500`.

`grad` Finite-difference type when `gr = NULL`: `"forward"` (default) or `"central"`.

`gradstep` Length-2 vector; step is $|x_i| \cdot$ `gradstep[1]` + `gradstep[2]`. Default `c(1e-6, 1e-8)`.

`invhessian.lt` A vector containing the lower triangle of the initial inverse Hessian (packed column-major). Default: identity.

Value

A list of class `"ucminf"` with elements:

`par` Computed minimizer.

`value` Objective value at the minimizer.

`convergence` Termination code:

- **1** Small gradient (`grtol`).
- **2** Small step (`xtol`).
- **3** Evaluation limit (`maxeval`).
- **4** Zero step from line search.
- **-2** n <= 0.
- **-4** stepmax <= 0.
- **-5** grtol or xtol <= 0.
- **-6** maxeval <= 0.
- **-7** Given inverse Hessian not positive definite.

`message` Human-readable termination message.

`invhessian.lt` Lower triangle of the final inverse-Hessian approximation (packed).

`invhessian` Full inverse-Hessian matrix (when `hessian >= 2`).

`hessian` Hessian matrix, inverse of `invhessian` (when `hessian == 3`).

`info` Named vector:

- **maxgradient** $\|g(x)\|_\infty$ at solution.
- **laststep** Length of the last step.
- **stepmax** Final trust-region radius.
- **neval** Number of function/gradient evaluations.

References

Nielsen, H. B. (2000). *UCMINF – An Algorithm for Unconstrained, Nonlinear Optimization*. Report IMM-REP-2000-19, DTU.

Examples

```
## Rosenbrock Banana function
fR <- function(x) (1 - x[1])^2 + 100 * (x[2] - x[1]^2)^2
gR <- function(x) c(-400 * x[1] * (x[2] - x[1] * x[1]) - 2 * (1 - x[1]),
                    200 * (x[2] - x[1] * x[1]))

## Find minimum with analytic gradient
ucminf(par = c(2, 0.5), fn = fR, gr = gR)

## Find minimum with finite-difference gradient
ucminf(par = c(2, 0.5), fn = fR)
```

---

ucminf_control *Build a validated control list for ucminf()*

Description

Returns a named list of control parameters that can be passed directly to `ucminf()`, `ucminf_xptr()`, etc.

Usage

```
ucminf_control(
  trace = 0,
  grtol = 1e-06,
  xtol = 1e-12,
  stepmax = 1,
  maxeval = 500L,
  grad = c("forward", "central"),
  gradstep = c(1e-06, 1e-08),
  invhessian.lt = NULL
)
```

Arguments

`trace`	Integer. If > 0, print convergence info. Default 0.
`grtol`	Gradient tolerance. Default 1e-6.
`xtol`	Step tolerance. Default 1e-12.
`stepmax`	Initial trust-region radius. Default 1.
`maxeval`	Maximum function evaluations. Default 500.
`grad`	Finite-difference type: "forward" or "central". Default "forward".
`gradstep`	Length-2 step vector. Default c(1e-6, 1e-8).
`invhessian.lt`	Packed lower-triangle of initial inverse Hessian. Default NULL.

Value

A list of class `"ucminf_control"`.

`ucminf_xptr` *Optimize using a compiled C++ objective (XPtr interface)*

Description

Calls the UCMINF optimizer with a compiled C++ objective function that has been wrapped in an `Rcpp::XPtr<ucminf::ObjFun>`. This path bypasses the R interpreter on every function evaluation, giving maximum performance for non-trivial objectives.

Usage

```
ucminf_xptr(par, xptr, control = list(), hessian = 0)
```

Arguments

`par`	Numeric starting vector.
`xptr`	An `externalptr` created by wrapping a `ucminf::ObjFun*` in `Rcpp::XPtr<ucminf::ObjFun>`.
`control`	A named list of control parameters (see `ucminf`), or a `ucminf_control` object.
`hessian`	Integer: 0 = none, 2 = inv-Hessian, 3 = both.

Value

A list of class `"ucminf"` (same structure as `ucminf`).

See Also

`ucminf`, `ucminf_control`

Index