

Package: ROI.format.cbf (via r-universe)

September 24, 2026

Type Package

Title Conic Benchmark Format (CBF) Plugin for the R Optimization Infrastructure

Version 0.1-0

Date 2026-03-19

Author Benjamin Schwendinger [aut, cre]

Maintainer Benjamin Schwendinger <benjaminschwe@gmail.com>

Description Provides read and write support for the Conic Benchmark Format (CBF, version 4) within the R Optimization Infrastructure ('ROI'). Supported cone types include the positive orthant, second-order (SOC), rotated second-order (bridged automatically to standard SOC), exponential (primal and dual), power (primal and dual), and semidefinite (symmetric-vectorised) cones, as well as their mixed-integer variants. The reader translates a .cbf file into an ROI 'OP' object, handling coordinate-convention differences between CBF and ROI transparently; the writer serialises an ROI 'OP' object back to CBF plain-text.

License GPL-3

URL <https://gitlab.com/roigrp/tools/roi.format.cbf>

BugReports <https://gitlab.com/roigrp/tools/roi.format.cbf/-/issues>

Imports ROI (>= 1.0-0), slam (>= 0.1-40)

Suggests tinytest, ROI.plugin.ecos, ROI.plugin.scs

Encoding UTF-8

RoxygenNote 7.3.2

Repository <https://roigrp.r-universe.dev>

Date/Publication 2026-04-18 14:47:22 UTC

RemoteUrl <https://gitlab.com/roigrp/tools/roi.format.cbf>

RemoteRef HEAD

RemoteSha e7944378416d198ec6d24d0cb76b8db97bb85864

Contents

cblib_download	2
cblib_download_all	3
read_cbf	4
write_cbf	5

Index	7
--------------	----------

cblib_download	<i>Download a Single CBLIB Instance</i>
----------------	---

Description

Downloads one Conic Benchmark Library (CBLIB) instance from <https://cblib.zib.de/download/all/>. By default the downloaded .cbf.gz archive is decompressed to a plain .cbf file so it can be passed directly to [read_cbf](#).

Usage

```
cblib_download(
  instance,
  folder,
  url = .CBLIB_DOWNLOAD_ALL_URL,
  method = NULL,
  quiet = TRUE,
  force = FALSE,
  decompress = TRUE,
  keep_archive = !decompress
)
```

Arguments

instance	Character string. Instance name, with or without the .cbf.gz suffix.
folder	Character string. Destination directory. No default is provided; pass tempdir() for a temporary location.
url	Character string. Base URL of the CBLIB download directory.
method	Character string passed to download.file .
quiet	Logical. Passed to download.file .
force	Logical. If TRUE, download again even if the target file already exists.
decompress	Logical. If TRUE (default), decompress the archive to .cbf after download.
keep_archive	Logical. If TRUE, keep the downloaded .cbf.gz archive after decompression.

Details

The CBLIB download README documents recursive retrieval from <https://cblib.zib.de/download/all/> using wget. These helpers provide the same access pattern directly from R for either a single instance or the full directory listing.

Value

Character string giving the path to the downloaded file. Returns the decompressed .cbf path when decompress = TRUE, otherwise the .cbf.gz archive path.

See Also

[cblib_download_all](#), [read_cbf](#)

Examples

```
## Not run:
path <- cblib_download("CBLIB_ex1.cbf", folder = tempdir())
op <- read_cbf(path)

## End(Not run)
```

cblib_download_all	<i>Download All CBLIB Instances</i>
--------------------	-------------------------------------

Description

Downloads every .cbf.gz instance linked from <https://cblib.zib.de/download/all/>. By default each archive is decompressed to a plain .cbf file.

Usage

```
cblib_download_all(
  folder,
  url = .CBLIB_DOWNLOAD_ALL_URL,
  method = NULL,
  quiet = TRUE,
  force = FALSE,
  decompress = TRUE,
  keep_archive = !decompress
)
```

Arguments

folder	Character string. Destination directory. No default is provided; pass tempdir() for a temporary location.
url	Character string. Base URL of the CBLIB download directory.
method	Character string passed to download.file .
quiet	Logical. Passed to download.file .
force	Logical. If TRUE, download again even if the target file already exists.
decompress	Logical. If TRUE (default), decompress the archive to .cbf after download.
keep_archive	Logical. If TRUE, keep the downloaded .cbf.gz archive after decompression.

Details

According to the CBLIB download README, the complete collection is intended to be fetched recursively from the `/download/all/` directory. This helper first reads that index page, extracts all linked `.cbf.gz` archives, and then downloads them one by one.

Value

Named character vector of downloaded file paths. Names are the CBLIB archive file names listed in the index.

See Also

[cblib_download](#), [read_cbf](#)

Examples

```
## Not run:
paths <- cblib_download_all(folder = tempdir())

## End(Not run)
```

read_cbf

Read a CBF File into an ROI OP Object

Description

Parses a file in the Conic Benchmark Format (CBF version 4) and returns an [OP](#) object representing the encoded optimisation problem.

Usage

```
read_cbf(file, ...)
```

Arguments

<code>file</code>	Character string. Path to the <code>.cbf</code> (or <code>.CBF</code>) file.
<code>...</code>	Currently unused; present for API compatibility with ROI_plugin_register_reader .

Details

Supported features

- Scalar variables (VAR) with all non-parametric cone types: `F.`, `L+.`, `L-.`, `L=.`
- Scalar constraints (CON) with non-parametric cones: `F.`, `L+.`, `L-.`, `L=.`, `Q.`, `QR.`, `EXP.`, `EXP*.`, `SVECPD.`, `GMEANABS.`, `GMEAN.`, and their duals.
- Parametric power cones (`@k:POW`, `@k:POW*`) via the `POWCONES` / `POW*CONES` lookup tables.

- Integer variables (INT section).
- Linear and constant-term objective coefficients (OBJACCOORD, OBJBCOORD).
- PSD matrix variables (PSDVAR): each $X_j \in S_+^{m_j \times m_j}$ is replaced by its symmetric vectorisation $\text{svec}(X_j)$ of length $m_j(m_j + 1)/2$, appended as new scalar variables with a $K_{\text{psd}}(m_j)$ cone constraint. Objective terms (OBJFCOORD) and scalar-constraint terms (FCOORD) involving X_j are translated via the trace inner-product identity $\langle F, X_j \rangle = \text{svec}(F)^\top \text{svec}(X_j)$.
- PSD matrix constraints (PSDCON / HCOORD / DCOORD): each LMI $G_p = \sum_j x_j H_{pj} + D_p \in \text{PSD}^{m_p}$ is svec-ed into a $K_{\text{psd}}(m_p)$ conic constraint on the scalar variables.

Features that generate a warning() and are skipped:

- Hotstart sequences (CHANGE keyword is treated as end-of-file).

Value

An [OP](#) object.

See Also

[write_cbf](#)

Examples

```
## round-trip: write a simple LP to a temp file then read it back
op <- ROI::OP(
  objective = ROI::L_objective(c(1, 1)),
  constraints = ROI::L_constraint(
    L = matrix(c(1, 1), nrow = 1),
    dir = ">=",
    rhs = 1
  )
)
f <- file.path(tempdir(), "lp.cbf")
write_cbf(op, f)
op2 <- read_cbf(f)
```

write_cbf

Write an ROI OP Object to a CBF File

Description

Serialises an ROI optimisation problem ([OP](#)) to a plain text file in the Conic Benchmark Format (CBF version 4).

Usage

```
write_cbf(op, file, ...)
```

Arguments

op	An OP object.
file	Character string. Destination file path (created or overwritten).
...	Currently unused.

Details

Supported OP components

Objective Linear objectives ([L_objective](#)).

Constraints [L_constraint](#) is mapped to L+ (" $\geq$ "), L- (" $\leq$ "), or L= (" $=$ ") cone chunks.

[C_constraint](#) is mapped to the appropriate CBF cone: Q., QR., EXP., EXP*., SVECPSD., @k:POW, or @k:POW*.

Variable types Continuous ("C") and integer ("I"); integer variables are written to the INT section.

Variable bounds Translated to F., L+, L-, or L= chunks in the VAR section.

Value

Invisibly returns file.

Sign convention

CBF encodes $Ax + b \in K$. ROI stores $Ax - \text{rhs} \in K$. Therefore write_cbf writes $b = -\text{rhs}$ to the BCOORD section.

See Also

[read_cbf](#)

Examples

```
## simple LP: min x + y s.t. x + y >= 1, x, y >= 0
op <- ROI::OP(
  objective = ROI::L_objective(c(1, 1)),
  constraints = ROI::L_constraint(
    L = matrix(c(1, 1), nrow = 1),
    dir = ">=",
    rhs = 1
  )
)
f <- file.path(tempdir(), "lp.cbf")
write_cbf(op, f)
```

Index

C_constraint, [6](#)
cblib_download, [2](#), [4](#)
cblib_download_all, [3](#), [3](#)

download.file, [2](#), [3](#)

L_constraint, [6](#)
L_objective, [6](#)

OP, [4–6](#)

read_cbf, [2](#), [3](#), [4](#), [4](#), [6](#)
ROI_plugin_register_reader, [4](#)

write_cbf, [5](#), [5](#)