# Package: sparselu (via r-universe)

October 6, 2026

**Type** Package

**Title** Sparse LU Decomposition via SuiteSparse

**Version** 0.3.0

**Maintainer** Kevin Michael Frick <kmfrick@proton.me>

**Description** Provides an interface to the SuiteSparse UMFPACK LU factorisation routines for sparse matrices stored in compressed column format. Implements the algorithm described in Davis (2004) <doi:10.1145/992200.992206>.

**License** GPL-3

**Depends** R (>= 3.6.0)

**LinkingTo** Rcpp

**SystemRequirements** SuiteSparse (UMFPACK, AMD, SuiteSparse_config)

**Imports** Rcpp (>= 0.11.0)

**Suggests** Matrix, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**Encoding** UTF-8

**OS_type** unix

**RoxygenNote** 7.3.2

**NeedsCompilation** yes

**Author** Kevin Michael Frick [aut, cre], Timothy A. Davis [ctb] (affiliation: SuiteSparse Project, contribution: SuiteSparse libraries and collaborators listed in dir(system.file(``doc", ``SuiteSparse", package = ``Matrix"), pattern = ``License", full.names = TRUE, recursive = TRUE))

**Config/pak/sysreqs** libsuitesparse-dev

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

**Date/Publication** 2026-03-10 20:50:11 UTC

**RemoteUrl** https://github.com/cran/sparselu

**RemoteRef** HEAD

**RemoteSha** 706f1a5348db3ee3c32c40eaa138e752b2279f6e

# Contents

sparseLU . . . . . 2
sparseLU_solve . . . . . 3

**Index** **4**

---

`sparseLU` *Sparse LU Decomposition*

---

## Description

Compute an LU factorisation of a sparse matrix stored in compressed column storage using the SuiteSparse UMFPACK routines.

## Usage

```
sparseLU(Ap, Ai, Ax)
```

## Arguments

| | |
|---|---|
| `Ap` | Integer vector of column pointers indexing into `Ai` and `Ax`. |
| `Ai` | Integer vector of row indices for each non-zero element. |
| `Ax` | Numeric vector of the non-zero values. |

## Details

The column pointers `Ap` and row indices `Ai` must use zero-based indexing as required by the SuiteSparse UMFPACK interface.

## Value

A named list with components `L`, `U`, `P`, and `Q` describing the LU factorisation returned by UMFPACK.

## Examples

```
Ap <- c(0L, 2L, 3L, 5L)
Ai <- c(0L, 2L, 1L, 0L, 2L)
Ax <- c(1, 4, 3, 2, 5)

sparseLU(Ap, Ai, Ax)
```

`sparseLU_solve` *Solve a Sparse Linear System*

## Description

Solve the sparse linear system $Ax = b$ using the SuiteSparse UMFPACK LU factorisation.

## Usage

```
sparseLU_solve(Ap, Ai, Ax, b)
```

## Arguments

| | |
|---|---|
| `Ap` | Integer vector of column pointers indexing into `Ai` and `Ax`. |
| `Ai` | Integer vector of row indices for each non-zero element. |
| `Ax` | Numeric vector of the non-zero values. |
| `b` | Numeric vector containing the right-hand side of the linear system. |

## Details

The sparse matrix is provided in compressed column storage using zero-based indexing in `Ap` and `Ai`, matching the expectations of the SuiteSparse UMFPACK interface.

## Value

Numeric vector with the solution to the system.

## Examples

```
Ap <- c(0L, 2L, 3L, 5L)
Ai <- c(0L, 2L, 1L, 0L, 2L)
Ax <- c(1, 4, 3, 2, 5)
b <- c(1, 2, 3)

sparseLU_solve(Ap, Ai, Ax, b)
```

# Index

sparseLU, 2
sparseLU_solve, 3