Stdlib Sparse matrix API

**Goals**

To be compact instead of being exhaustive. It aims at supplying Fortran users with a minimum (yet useful) number of routines and data structures related to sparse matrices storage and operations. This library is particularly targeted at a non-expert in numerical computation public. Thus we aim at having a simple and easy to use API.

**1 Sparse matrix representations supported**

This section is based on Saad (1994). In that work, a much more complete and extensive list of formats is described. Here we take only the ones that we think are most useful at the moment.

Some questions:

Support one-based indexing?

Support zero-based indexing?

**1.1 Coordinate format (COO)**

Given an $m$ by $n$ real or complex matrix $A$ containing $nnz$ nonzero elements with each element denoted by $a\_{ij}$ this format represents $A$ using a set of three arrays: *values*, *rows*, and *columns,* as described below.

*values* A real/complex array of size $nnz$ containing the matrix elements $a\_{ij}$ in any order.

*rows* An integer array of size $nnz$ containing the row indices of the elements $a\_{ij}$.

*columns* An integer array of size $nnz$ containing the column indices of the elements $a\_{ij}$.

**1.2 Compressed Sparse Row (CSR)**

Given an $m$ by $n$ real or complex matrix $A$ containing $nnz$ nonzero elements with each element denoted by $a\_{ij}$ this format represents $A$ using a set of three arrays: *values*, *ja*, and *ia,* as described below.

$values$A real/complex array of size $nnz$ containing the matrix elements $a\_{ij}$ stored row by row from row 1 to row $n$.

$ja$ An integer array of size $nnz$ containing the column indices of the elements $a\_{ij}$ as stored in the array *values*.

$ia$An integer array of size $n+1$ containing the index in the arrays $values$ and $ja$ where each row starts. The value at $ia(n+1)$ always has the value $ia\left(1\right)+nnz$.

**1.3 Compressed Sparse Column (CSC)**

This format is similar to the CSR format described previously. The difference is that instead of storing row values we store column values in the array $values$. The exact description of this format is given below.

Given an $m$ by $n$ real or complex matrix $A$ containing $nnz$ nonzero elements with each element denoted by $a\_{ij}$ this format represents $A$ using a set of three arrays: *values*, *ja*, and *ia,* as described below.

$values$A real/complex array of size $nnz$ containing the matrix elements $a\_{ij}$ stored *column* by *column* from column 1 to column $m$.

$ia$ An integer array of size $nnz$ containing the *row* indices of the elements $a\_{ij}$ as stored in the array *values*.

$ja$An integer array of size $m+1$ containing the index in the arrays $values$ and $ia$ where each column starts. The value at $ja(m+1)$ always has the value $ja\left(1\right)+nnz$.

**2 Creational subroutines**

**3 Conversion subroutines**

**4 Algebraic operations**

**4 Utilities**

**4 Input/Output**

References

Saad, Y., SPARSKIT: A basic tool kit for sparse matrix computation*, 1994.*<https://www-users.cs.umn.edu/~saad/software/SPARSKIT/>