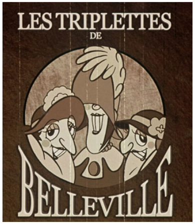
CS319: Scientific Computing (with MATLAB)

Sparse Matrices

Niall Madden

Week 8: **9am**, 01 March 2023



Useful reading:

- Learning MATLAB, Section 2.5 (brief overview)
  https://doi-org.nuigalway.idm.oclc.org/10.1137/1.9780898717662
- The MATLAB Guide, Chapters 9 (Linear Algebra) and 15 (Sparse Matrices) of
  https://doi-org.nuigalway.idm.oclc.org/10.1137/1.9781611974669

This week...

In Week 10 (approx) we'll study how graphs and networks are represented in MATLAB. We'll see that one of the key ideas is that one can represent a graph as a matrix, called the **adjacency matrix**.

Adjacency Matrix

The adjacency matrix of a graph with n vertices is the $n \times n$ matrix, A, where

$$a_{i,j} = \begin{cases} 1 & \text{if there is an edge between vertices } i \text{ and } j; \\ 0 & \text{otherwise} \end{cases}$$

Example:

Motivation

In most real world examples, such as graphs of social networks, the the **number of non-zeros** (**NNZs**) is relatively small. So it does not make sense to store them all.

There are schemes for storing such matrices, in a what that saves memory and speeds up calculation. The most common method is **Triplet**, which is used by MATLAB (kinda).

There are other methods, such as Compressed Row Storage (CRS), Compressed Column Storage (CCS), Compressed Diagonal Storage, Skyline...

What is a matrix sparse?

Although the representation and manipulation of sparse matrices is an major topic in Scientific Computing, there isn't a universally agreed definition of an (abstract) *sparse matrix*.
This is because, when coding, we should ask the question: **"When is it worth the effort to store a matrix in a sparse format, rather than in standard (dense) format?"**
The answer is often context-dependent. But roughly, use a sparse format when

- The memory required by the sparse format is less then the "dense" (or "full") one;
- The expense of updating the sparse format is not excessive;
- Computing a matrix-vector product ("`MatVec`") is faster for a sparse matrix than for full.

2: Triplet Format

The basic idea for triplet form is: to store a **sparse** matrix with `NNZ` non-zeros we ...

- define `integer` arrays `I[NNZ]` and `J[NNZ]`,
- a `double` array `X[NNZ]`.
- Then entry a_{ij} is stored as `I[k]=i`, `J[k]=j`, `X[k]=`a_{ij}, for some k.

Example: write down the triplet form of the following matrix:

$$\begin{pmatrix} 1 & 0 & 11 & 0 \\ 1 & 0 & 2 & 0 \\ 9 & 19 & 0 & 29 \\ 0 & 0 & 0 & 5 \end{pmatrix}$$

2: Triplet Format

Question: For a given N, for what value of NNZ is it worth using a sparse storage format?

3: CCS Format

Although MATLAB presents sparse matrices in triplet format, it actually stores them in a different way.

If we know that the entries in our matrix are stored in order, then it is possible to store the matrix more efficiently that in Triplet format. One way of doing this is to use **CCS**: **Compressed Column Storage**, also known as *Harwell-Boeing*
The matrix is stored in 3 vectors:

- a `double` array, x of length `nnz` ("number of nonzero entries") storing the non-zero entries matrix, in column-wise order.
- an `int` array, r of length `nnz` storing **row** index of the entries. That is, $x[k]$ would be found in row $r[k]$ of the full matrix.
- an `int` array, c of length $N+1$, where $c[k]$ stores that starting point of column k as it appears in the arrays x and r, and $c[N] = nnz$.

3: CCS Format

Example

Show how the matrix below would be stored in CCS

$$\begin{pmatrix} 2 & -1 & 0 & -2 \\ -3 & 5 & -1 & 0 \\ 0 & -2 & 4 & 0 \\ -3 & 0 & 0 & 4 \end{pmatrix}$$

4: Sparse matrices in MATLAB

[See live script]