About this class: Objectives

Set 1  An introduction to sparse matrices and sparse matrix computations.
- Sparse matrices;
- Sparse matrix direct methods;
- Graph theory viewpoint; graph theory methods;

Set 2  Iterative methods and eigenvalue problems
- Iterative methods for linear systems
- Algorithms for sparse eigenvalue problems and the SVD
- Possibly: nonlinear equations

Set 3  Applications of sparse matrix techniques
- Applications of graphs; Graph Laplaceans; Networks ...;
- Standard Applications (PDEs, ..)
- Applications in machine learning
- Data-related applications
- Other instances of sparse matrix techniques

Please fill out (now if you can)
This survey
short link url:
https://forms.gle/yiXjHGXrzkwaf2Ex9
Logistics:

- Lecture notes and minimal information will be located here:

  8314 at CSE-labs
  www-users.cselabs.umn.edu/classes/Spring-2021/csci8314/

- There you will find:
  - Lecture notes, Schedule of assignments/tests, class info
  - Canvas will contain the rest of the information: assignments, grades, etc.

About lecture notes:

- Lecture notes (like this first set) will be posted on the class web-site – usually before the lecture.
- Note: format used in lectures may be formatted differently – but same contents.
- Review them to get some understanding if possible before class.
- Read the relevant section(s) in the texts or references provided.
- Lecture note sets are grouped by topics (sections in the textbook) rather than by lecture.
- In the notes the symbol \( \blacklozenge \) indicates suggested easy exercises or questions – often [not always] done in class.
- Also: occasional practice exercises posted

Matlab

- We will often use matlab for testing algorithms.
- Other documents will be posted in the matlab section of the class web-site.
- Also:
  - I post the matlab diaries used for the demos (if any).

CSCI 8314: SPARSE MATRIX COMPUTATIONS

GENERAL INTRODUCTION

- General introduction - a little history
- Motivation
- Resources
- What will this course cover
What this course is about

- Solving linear systems and (to a lesser extent) eigenvalue problems with matrices that are sparse.
- Sparse matrices: matrices with mostly zero entries [details later]
- Many applications of sparse matrices...
- ... and we are seeing more with new applications everywhere

A brief history

Sparse matrices have been identified as important early on – origins of terminology is quite old. Gauss defined the first method for such systems in 1823. Varga used explicitly the term 'sparse' in his 1962 book on iterative methods.


- Special techniques used for sparse problems coming from Partial Differential Equations
  - One has to wait until the 1960s to see the birth of the general technology available today
  - Graphs introduced as tools for sparse Gaussian elimination in 1961 [Seymour Parter]

History: development of iterative methods

- Early work on reordering for banded systems, envelope methods
- Various reordering techniques for general sparse matrices introduced.
- Minimal degree ordering [Markowitz - 1957] ...
- ... later used in Harwell MA28 code [Duff] - released in 1977.
- Tinney-Walker Minimal degree ordering for power systems [1967]
- Nested Dissection [A. George, 1973]
- SPARSPAK [commercial code, Univ. Waterloo]
- Elimination trees, symbolic factorization, ...
History: eigenvalue problems

- Another parallel branch was followed in sparse techniques for large eigenvalue problems.
- A big problem in 1950s and 1960s: flutter of airplane wings. This leads to a large (sparse) eigenvalue problem.
- Overlap between methods for linear systems and eigenvalue problems [Lanczos, Arnoldi]

Resources

- Matrix market
  - http://math.nist.gov/MatrixMarket/
- SuiteSparse site (Formerly: Florida collection)
  - https://sparse.tamu.edu/
- SPARSKIT, etc. [SPARSKIT = old written in Fortran. + more recent ‘solvers’]
  - http://www.cs.umn.edu/~saad/software

Books: on sparse direct methods.

- Book by Tim Davis [SIAM, 2006] see syllabus for info
- Best reference [old, out-of print, but still the best]:
- Of interest mostly for references:
  - Some coverage in Golub and van Loan [John Hopinks, 4th edition, see chapters 10 to end]

Overall plan for this course

- We will begin by sparse matrices in general, their origin, storage, manipulation, etc..
- Graph theory viewpoint
- We will then spend some time on sparse direct methods
- .. back to graphs: Graph Laplaceans and applications; Networks; ...
- .. and then on eigenvalue problems and
- ... iterative methods for linear systems
- ... Plan is somewhat dynamic
- ... at the end of semester: a few lectures given by you
SPARSE MATRICES

- See Chap. 3 of text
- See the “links” page on the class web-site
- See also the various sparse matrix sites.
- Introduction to sparse matrices
- Sparse matrices in matlab –

What are sparse matrices?

Vague definition: matrix with few nonzero entries

- For all practical purposes: an $m \times n$ matrix is sparse if it has $O(\min(m, n))$ nonzero entries.
- This means roughly a constant number of nonzero entries per row and column -
- This definition excludes a large class of matrices that have $O(\log(n))$ nonzero entries per row.
- Other definitions use a slow growth of nonzero entries with respect to $n$ or $m$.

Pattern of a small sparse matrix

“..matrices that allow special techniques to take advantage of the large number of zero elements.” (J. Wilkinson)

A few applications which lead to sparse matrices:

Structural Engineering, Computational Fluid Dynamics, Reservoir simulation, Electrical Networks, optimization, Google Page rank, information retrieval (LSI), circuit simulation, device simulation, .....

1-17 Chap 3 – sparse
1-18 Chap 3 – sparse
**Goal of Sparse Matrix Techniques**

- To perform standard matrix computations economically i.e., without storing the zeros of the matrix.

**Example:** To add two square dense matrices of size $n$ requires $O(n^2)$ operations. To add two sparse matrices $A$ and $B$ requires $O(nnz(A) + nnz(B))$ where $nnz(X) =$ number of nonzero elements of a matrix $X$.

- For typical Finite Element / Finite difference matrices, number of nonzero elements is $O(n)$.

**Remark:** $A^{-1}$ is usually dense, but $L$ and $U$ in the LU factorization may be reasonably sparse (if a good technique is used).

---

**Nonzero patterns of a few sparse matrices**


SHERMAN5: Fully implicit black oil simulator 16 by 23 by 3 grid, 3 unk

PORES3: Unsymmetric matrix from pores

BP_1000: Unsymmetric basis from LP problem BP

---

**Exercises**

1. Look up Cayley-Hamilton’s theorem if you do not know about it.

2. Show that the inverse of a matrix (when it exists) can be expressed as a polynomial of $A$, where the polynomial is of degree $\leq n - 1$.

3. When is the degree $< n - 1$? [Hint: look-up minimal polynomial of a matrix]

4. What is the pattern of the inverse of a tridiagonal matrix? a bidiagonal matrix?
**Types of sparse matrices**

- Two types of matrices: structured (e.g. Sherman5) and unstructured (e.g. BP1000)
- The matrices PORES3 and SHERMAN5 are from Oil Reservoir Simulation. Often: 3 unknowns per mesh point (Oil, Water saturations, Pressure). Structured matrices.
- 40 years ago reservoir simulators used rectangular grids.
- Modern simulators: Finer, more complex physics (harder and larger systems). Also: unstructured matrices.
- A naive but representative challenge problem: $100 \times 100 \times 100$ grid + about 10 unknowns per grid point. Also: $N \approx 10^7$, and $nnz \approx 7 \times 10^8$.

**Solving sparse linear systems: existing methods**

- Direct sparse solvers
- Iterative methods: Preconditioned Krylov

- A naive challenge problem: $100 \times 100 \times 100$

**Two types of methods for general systems:**

- **Direct methods**: based on sparse Gaussian elimination, sparse Cholesky,\

- **Iterative methods**: compute a sequence of iterates which converge to the solution - preconditioned Krylov methods.

**Remark:** These two classes of methods have always been in competition.

- 40 years ago solving a system with $n = 10,000$ was a challenge.
- Now you can solve this in a fraction of a second on a laptop.

**Difficulty:**

- No robust 'black-box' iterative solvers.
- At issue: Robustness in conflict with efficiency.

- Iterative methods are starting to use some of the tools of direct solvers to gain 'robustness'.
Consensus:

1. Direct solvers are often preferred for two-dimensional problems (robust and not too expensive).
2. Direct methods lose ground to iterative techniques for three-dimensional problems, and problems with a large degree of freedom per grid point.

Sparse matrices in matlab

- Matlab supports sparse matrices to some extent.
- Can define sparse objects by conversion
  \[ A = \text{sparse}(X); \ X = \text{full}(A) \]
- Can show pattern
  \[ \text{spy}(X) \]
- Define the analogues of ones, eye:
  \[ \text{speye}(n,m), \ \text{spones}(\text{pattern}) \]

Graph Representations of Sparse Matrices

- A few reorderings functions provided.. [will be studied in detail later]
  \[ \text{symrcm}, \ \text{symamd}, \ \text{colamd}, \ \text{colperm} \]
- Random sparse matrix generator:
  \[ \text{sprand}(S) \text{ or } \text{sprand}(m,n, \text{density}) \]
  (also \text{textttsrandn}(...) )
- Diagonal extractor-generator utility:
  \[ \text{spdiags}(A), \ \text{spdiags}(B,d,m,n) \]
- Other important functions:
  \[ \text{spalloc}(..), \ \text{find}(..) \]

DEFINITION. A graph \( G \) is defined as a pair of sets \( G = (V, E) \) with \( E \subset V \times V \). So \( G \) represents a binary relation. The graph is undirected if the binary relation is reflexive. It is directed otherwise. \( V \) is the vertex set and \( E \) is the edge set.

Example: Given the numbers 5, 3, 9, 15, 16, show the two graphs representing the relations

R1: Either \( x < y \) or \( y \) divides \( x \).
R2: \( x \) and \( y \) are congruent modulo 3. [ \( \text{mod}(x,3) = \text{mod}(y,3) \)]
Adjacency Graph $G = (V, E)$ of an $n \times n$ matrix $A$:

- Vertices $V = \{1, 2, \ldots, n\}$.
- Edges $E = \{(i, j)|a_{ij} \neq 0\}$.

- Often self-loops $(i, i)$ are not represented [because they are always there].
- Graph is undirected if the matrix has a symmetric structure:
  \[ a_{ij} \neq 0 \text{ iff } a_{ji} \neq 0. \]

Example: (directed graph)

Example: (undirected graph)

Note: Matlab now has a graph function.

$G = \text{graph}(A)$ creates adjacency graph from $A$.

$G$ is a matlab class/

$G.\text{Nodes}$ will show the vertices of $G$

$G.\text{Edges}$ will show its edges.

plot($G$) will show a representation of the graph.
Do the following:

• Load the matrix 'Bmat.mat' located in the class web-site (see 'matlab' folder)
• Visualize pattern (spy(B)) + find: Number of nonzero elements, size, ...
• Generate graph - without self-edges:
  
  \[
  G = \text{graph}(B, \text{'OmitSelfLoops'})
  \]
• Plot the graph –
• $1M$ question: Any idea on how this plot is generated?