Matrices and Eigenvalues

Many scientific problems can be represented using matrix equations (includes PDE)

3 main types of matrix problems

1) Algebraic manipulations e.g. $A+B$ or AB
2) Solving systems of eq$\stackrel{\text{n}}{=}$s
   e.g. solve for $\vec{x}$ in $A\vec{x} = \vec{b}$
3) Eigenvalues $A\vec{x} = \alpha\vec{x}$

Matrix types include

- Hermitian $A_{ji} = A_{ij}^*$, $A^+ = A$, $A^+ \equiv (A^T)^*$
  Hermitian conjugate A^+ is complex conjugate of transpose.
- Real Symmetric: $A_{ji} = A_{ij}$ $A^T = A$
- Positive Definite: $Re\{z^+ M z\} > 0$ for all complex z

For M also Hermitian: $r = z^+ M z$ is real and positive and all eigenvalues are real and positive

- Unitary $U^+ U = I$ $U^{-1} = U^+$
- Diagonal $A_{ij} = 0$ for $i \neq j$
- Tridiagonal non-zero elements only for A_{ii} and $A_{i,i\pm1}$

$$\begin{pmatrix} a_{11} & a_{12} & 0 & 0 & \cdots \\ a_{21} & a_{22} & a_{23} & 0 & \cdots \\ 0 & a_{32} & a_{33} & & \\ 0 & 0 & & \ddots & \\ \vdots & & & & \end{pmatrix}$$

- Upper and Lower Triangular

$A_{ij} = 0$ for $i > j$ or $i < j$

$$\begin{pmatrix} x & x & x & \cdots \\ 0 & x & x & \\ 0 & 0 & x & \ddots \\ \vdots & \vdots & & \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} x & 0 & 0 & 0 & \cdots \\ x & x & 0 & 0 & \cdots \\ x & x & x & 0 & \\ x & x & x & x & \\ & & & \ddots & \end{pmatrix}$$

- Sparse Matrix – most elements are zero
  - Useful to recognize if number of non-zero elements is $\sim N$ (not N^2) in $N \times N$ matrix.

  Goal: manipulate matrix without storing all elements.

Matrix Algebra

Optimal matrix manipulation depends on language used.
In C, the matrix is stored row after row.
In Fortran, matrix is stored column after column

C: "row major" F: "column major"

Example: In Fortran

```
do j = 1, n
  do i = 1, n
    A(i,j) = B(i,j) + C(i,j)
  enddo
enddo
```

↑ run over all rows i in column j

is faster than

```
do i = 1, n
  do j = 1, n
    A(i,j) = B(i,j) + C(i,j)
  enddo
enddo
```

↑ run over j columns of row i

F90 has built-in matrix manipulation routines

- above is just $A = B + C$
- $A = B * C$ is $A_{ij} = B_{ij} * C_{ij}$
- $A =$ MATMUL(B, C) is usual matrix mult.

show InnerLoopROW.f90, InnerLoopCOL.f90 ...

Systems of Linear Equations

Consider solving for

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

A is $n \times n$

- for solution to exist, $\det(A) = |A| \neq 0$

We can use Gaussian elimination

- idea: transform set of eq$^{\underline{n}}$s so that coefficient matrix is upper (or lower) triangular
- at each step of algorithm, eliminate lower (upper) elements of the matrix

Ex 3×3

$$a_{11} x_1 + a_{12} x_2 + a_{13} x_3 = b_1 \qquad (1)$$
$$a_{21} x_1 + a_{22} x_2 + a_{23} x_3 = b_2 \qquad (2)$$
$$a_{31} x_1 + a_{32} x_2 + a_{33} x_3 = b_3 \qquad (3)$$

- multiply (1) by $\frac{a_{21}}{a_{11}}$ and subtract from (2)
- multiply (1) by $\frac{a_{31}}{a_{11}}$ and " " (3)

$$\Rightarrow \quad a_{11} x_1 + a_{12} x_2 + a_{13} x_3 = b_1$$
$$0 + a_{22}' x_2 + a_{23}' x_3 = b_2' \qquad (2')$$
$$0 + a_{32}' x_2 + a_{33}' x_3 = b_3'$$

where
$$\left.\begin{aligned} a_{ij}' &= a_{ij} - a_{1j} \frac{a_{i1}}{a_{11}} \\ b_i' &= b_i - b_1 \frac{a_{i1}}{a_{11}} \end{aligned}\right\} \quad \begin{aligned} i &= 2, \ldots n \\ j &= 1, \ldots n \end{aligned}$$

x_1 eliminated from (2) and (3)

Now eliminate x_2 from (3′) via

taking $\frac{a'_{32}}{a'_{22}} \times (2')$ and subtracting from $(3')$

$$\Rightarrow \quad \begin{aligned} a_{11} x_1 + a_{12} x_2 + a_{13} x_3 &= b_1 \\ a'_{22} x_2 + a'_{23} x_3 &= b_2' \\ 0 \quad + a''_{33} x_3 &= b_3'' \end{aligned}$$

where

$$a''_{ij} = a'_{ij} - a'_{2j} \frac{a'_{i2}}{a'_{22}} \qquad i = 3, \dots n$$

$$b_i'' = b_i' - b_2' \frac{a'_{i2}}{a'_{22}} \qquad j = 2, \dots n$$

For $n \times n$ matrix, this procedure is done $n-1$ times

Once in upper triangular form, we get $\vec{x}$ through "back substitution"

$$x_3 = b_3'' / a'_{33}$$

$$x_2 = \frac{1}{a'_{22}} \left(b_2' - a'_{23} x_3 \right)$$

In general

$$\begin{aligned} \alpha_{11} x_1 + \alpha_{12} x_2 + \alpha_{13} x_3 + \dots + \alpha_{1n} x_n &= \tilde{b}_1 \\ 0 \quad \alpha_{22} x_2 + \alpha_{23} x_3 + \dots + \alpha_{2n} x_n &= \tilde{b}_2 \\ 0 \quad 0 \quad \alpha_{33} x_3 + \dots + \alpha_{3n} x_n &= \tilde{b}_3 \\ &\vdots \\ \alpha_{nn} x_n &= \tilde{b}_n \end{aligned}$$

$$x_n = \tilde{b}_n / \alpha_{nn}$$

$$x_i = \frac{1}{\alpha_{ii}} \left(\tilde{b}_i - \sum_{j=i+1}^{n} \alpha_{ij} x_j \right) \qquad i = n-1, n-2, \dots 1$$

Number of operations required for a set of n eq^{ns} is proportional to n^3

Large matrices can take a lot of time

For matrix manipulations it is best to use efficient algorithms/routines freely available from

e.g. www.netlib.org (BLAS, LAPACK)

- such routines, available here as reference implementations, have been incorporated in various libraries like

GSL GNU Scientific Library & MKL Math Kernel Library

(But using libraries without any idea of how they work can lead to trouble sooner or later.)

Such routines are optimized, thoroughly tested, and can avoid or flag potential problems.

E.g. Our simple Gaussian elimination would fail if any diagonal elements were zero, and would amplify errors if small

These problems can be reduced by pivoting, or rearranging rows s.t. $a_{11} > a_{22} > a_{33} > \dots > a_{nn}$

Depending on type of matrix (symmetric, tridiagonal etc) different routines exist for carryout out the required manipulations.

Manipulating matrices is often the most time-consuming part of a computational problem and hence the limiting step. Therefore, spending time to find the best routines for your problem is well worth it, and it can change the way you do research.

browse through LAPACK

Note on SGESV LAPACK routine used to solve
$A\vec{x} = \vec{b}$ for A a general matrix

Employs

$$A = LU = \begin{pmatrix} \beta_{11} & 0 & 0 & 0 & \cdots \\ \beta_{21} & \beta_{22} & 0 & 0 & \cdots \\ \beta_{31} & \beta_{32} & \beta_{33} & 0 & \cdots \\ \vdots & & & \ddots & \end{pmatrix} \begin{pmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} & \cdots \\ 0 & \alpha_{22} & \alpha_{23} & \cdots \\ 0 & 0 & \alpha_{33} & \cdots \\ 0 & 0 & 0 & \ddots \\ \vdots & & & \end{pmatrix}$$

$Ax = b$

$LUx = b$

$L(Ux) = b \longrightarrow Ux = y$

$Ly = b \longrightarrow$ solve first for y using "forward substitution" $y_1 = \dfrac{b_1}{\beta_{11}}$, $y_i = \dfrac{1}{\beta_{ii}}\left[b_i - \sum_{j=1}^{i-1} \beta_{ij} y_j \right]$

$i = 2, 3, \ldots N$

then get x through back substitution.

Show use of SGESV to solve

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 3 & 8 & 9 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}$$

$x_1 = 0$

$x_2 = -2$

$x_3 = 5/3$