

Lecture 2: Systems of Linear Equations (see “Recipes” Chapter 2)

- Systems of linear equations arise in many contexts. As we’ll see later, their solution is a fundamental tool in numerical methods.
- Consider a system of M linear equations in N unknowns

$$a_{11}x_1 + a_{12}x_2 + \dots + a_{1N}x_N = b_1$$

$$a_{21}x_1 + a_{22}x_2 + \dots + a_{2N}x_N = b_2$$

$$a_{31}x_1 + a_{32}x_2 + \dots + a_{3N}x_N = b_3$$

.

.

$$a_{M1}x_1 + a_{M2}x_2 + \dots + a_{MN}x_N = b_M$$

a_{ij}, b_i are known constants

x_j is unknown

Matrix notation

- In vector notation: $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$
- For a square matrix (N equations and N unknowns), the behavior is critically affected by the determinant,

$$|A| = \sum \varepsilon_{i_1 i_2 i_3 \dots i_N} a_{1i_1} a_{2i_2} a_{3i_3} \dots a_{Ni_N}$$

where the sum is computed with $i_1, i_2, i_3 \dots i_N$ each taking all possible values over the range 1 to N, and

$\varepsilon = -1$ if $i_1, i_2, i_3, \dots, i_N$ is an odd permutation of $1, 2, 3, \dots, N$ (e.g. as in ε_{2134})

$\varepsilon = +1$ if $i_1, i_2, i_3, \dots, i_N$ is an even permutation of $1, 2, 3, \dots, N$ (e.g. as in ε_{2143})

$\varepsilon = 0$ otherwise (i.e. any two indices are the same)

There are N^N terms in the sum, of which $N!$ are non-zero

Singular versus non-singular matrices

- If $M = N$, there is a unique solution for \mathbf{x} , namely $\mathbf{x} = \mathbf{A}^{-1} \cdot \mathbf{b}$, if and only if $|\mathbf{A}| \neq 0$.

Matrices with $|\mathbf{A}| \neq 0$ are said to be non-singular:

Matrices with $|\mathbf{A}| = 0$ are singular (and they have no inverse)

- Example: consider a diagonal square matrix,

$$\mathbf{A} = \begin{pmatrix} a_{11} & & & & \\ & a_{22} & & & \\ & & a_{33} & & \\ & & & \dots & \\ & & & & a_{NN} \end{pmatrix}$$

Singular versus non-singular matrices

$$\mathbf{A}^{-1} = \begin{pmatrix} 1/a_{11} & & & & \\ & 1/a_{22} & & & \\ & & 1/a_{33} & & \\ & & & \dots & \\ & & & & 1/a_{NN} \end{pmatrix}$$

exists iff all the a_{ij} are non-zero, so that a unique solution $\mathbf{x} = \mathbf{A}^{-1} \cdot \mathbf{b}$ exists

But if one or more a_{ij} are zero, $|\mathbf{A}| = \prod a_{ij} = 0$, the matrix is singular, and \mathbf{A}^{-1} does not exist

Singular versus non-singular matrices

- Matrices are singular if two or more rows are not linearly independent
 - We then have fewer than N linearly independent equations to determine N unknowns
- Numerically, roundoff errors introduce noise that can have two effects:
 - A non-singular problem can become approximately singular
 - ➔ the solution blows up (yielding huge, crazy solutions)
 - A stable, non-singular problem can produce the wrong answer
 - ➔ always need to check solution
 - For well conditioned (far from singular) problems, single precision works OK for $N < 30$ and double precision for $N < 300$

Non-square matrices ($N \neq M$)

- Two cases

$M < N$ – there is insufficient information to define a unique solution, but we can find the $(N - M)$ dimensional subspace that the N dimensional vector \mathbf{x} inhabits (singular value decomposition SVD, to be considered later).

$M > N$ – the solution is overconstrained (unless $M - N$ equations can be written as linear combinations of the remaining N). There is no exact solution, but there is one that minimizes the summed squared error

$$E = | \mathbf{A} \cdot \mathbf{x} - \mathbf{b} |^2 = (\mathbf{A} \cdot \mathbf{x} - \mathbf{b})^T \cdot (\mathbf{A} \cdot \mathbf{x} - \mathbf{b})$$

Minimization of the error when the solution is overconstrained

- The summed squared error, E , can be written

$$(\mathbf{A} \cdot \mathbf{x} - \mathbf{b})^T \cdot (\mathbf{A} \cdot \mathbf{x} - \mathbf{b}) = \mathbf{x}^T \mathbf{A}^T \mathbf{A} \mathbf{x} - \mathbf{b}^T \mathbf{A} \mathbf{x} - \mathbf{x}^T \mathbf{A}^T \mathbf{b} + \mathbf{b}^T \mathbf{b}$$

$1 \times N \quad N \times M \quad M \times N \quad N \times 1$

All terms are scalar

- Minimum error, E , occurs when

$$\begin{aligned} 0 = \partial E / \partial x_j &= (\mathbf{A}^T \mathbf{A})_{ji} x_i + (\mathbf{A}^T \mathbf{A})_{ij} x_i - (\mathbf{b}^T \mathbf{A})_j - (\mathbf{A}^T \mathbf{b})_j \\ &= 2 (\mathbf{A}^T \mathbf{A})_{ji} x_i - 2 (\mathbf{A}^T \mathbf{b})_j \end{aligned}$$

$\mathbf{A}^T \mathbf{A}$ is a symmetric $N \times N$ matrix

The solution is therefore obtained from equation

$$(\mathbf{A}^T \mathbf{A}) \mathbf{x} = \mathbf{A}^T \mathbf{b} \quad (N \text{ equations and } N \text{ unknowns})$$

and the corresponding value of E is $\mathbf{b}^T (\mathbf{b} - \mathbf{A} \cdot \mathbf{x})$

Solution of the N x N problem

LU decomposition

- Write **A** as a product of two triangular matrices
(non-unique: we discuss how to do this later)

$$\mathbf{A} = \mathbf{L} \cdot \mathbf{U} = \begin{pmatrix} L_{11} & 0 & 0 & \dots & 0 \\ L_{21} & L_{22} & 0 & \dots & 0 \\ L_{31} & L_{32} & L_{33} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ L_{N1} & L_{N2} & \dots & \dots & L_{NN} \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} & \dots & U_{1N} \\ 0 & U_{22} & \dots & U_{2N} \\ 0 & 0 & U_{33} & \dots & U_{3N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & U_{NN} \end{pmatrix}$$

Once **L** and **U** are known, **x** can easily be found

- The problem **A . x = b** can be written **L . y = b**, with **y ≡ U . x**

We solve first for **y** by “forward-substitution”

$$L_{11} \mathbf{y}_1 = b_1$$

$$\rightarrow y_1 = L_{11}^{-1} b_1$$

$$L_{21} y_1 + L_{22} \mathbf{y}_2 = b_2$$

$$\rightarrow y_2 = L_{22}^{-1} [b_2 - L_{21} y_1]$$

...

$$L_{i1} y_1 + L_{i2} y_2 + \dots L_{ii} \mathbf{y}_i = b_N$$

**The unknowns (“not-yet-knowns”)
appear in red – only one in each eqn**

$$\rightarrow y_i = L_{ii}^{-1} [b_i - L_{i1}y_1 - L_{i2}y_2 - L_{i3}y_3 - \dots L_{i,i-1}y_{i-1}]$$

KEY POINT: Each component of **y** can be written easily in terms of components already computed

Solution of the $N \times N$ problem LU decomposition

- Once we have \mathbf{y} , we solve $\mathbf{U} \cdot \mathbf{x} = \mathbf{y}$ just as easily by “backsubstitution” (essentially the same method, but starting with the bottom row: $U_{NN} x_N = y_N$)

Again, whenever we come to a particular component of x , we get an equation that involves only components that have already been computed.

So how do we find the LU decomposition of matrix **A**?

A has N^2 elements, each of which can be written

$$A_{ij} = L_{i1}U_{1j} + L_{i2}U_{2j} + \dots + L_{ik}U_{jk}$$

Need only go up to $k = \min(i,j)$, since $L_{ik} = 0$ for $k > i$ and $U_{jk} = 0$ for $k > j$

L and U each have $\frac{1}{2} N(N+1)$ non-zero elements, for a total of $N^2 + N$ unknowns

But there are only N^2 constraints, so the problem is underdetermined \rightarrow we can *choose* N of the elements

So how do we find the LU decomposition of matrix **A**

Convention: choose $L_{ii} = 1$ for $i = 1, N$

Can now solve all the others in such a way that each element can be written in terms of elements already computed

$$A_{11} = L_{11} \mathbf{U}_{11} \rightarrow U_{11} = A_{11}/L_{11} = A_{11}$$

$$A_{1j} = L_{11} \mathbf{U}_{1j} \rightarrow U_{1j} = A_{1j}/L_{11} = A_{1j}$$

$$A_{i1} = \mathbf{L}_{i1} U_{11} \rightarrow L_{i1} = A_{i1}/U_{11} = A_{i1}/A_{11}$$

So how do we find the LU decomposition of matrix **A**?

We continue

$$A_{2j} = L_{21}U_{1j} + L_{22}U_{2j} \rightarrow U_{2j} = A_{2j} - L_{21}U_{1j}$$

$$A_{i2} = L_{i1}U_{12} + L_{i2}U_{22} \rightarrow L_{i2} = U_{22}^{-1} (A_{i2} - U_{12}L_{i1})$$

$$\text{In general: } U_{ij} = A_{ij} - \sum_{k=1}^{i-1} L_{ik} U_{kj} \quad \text{go upward in } i$$

$$L_{ij} = U_{jj}^{-1} (A_{ij} - \sum_{k=1}^{j-1} L_{ik} U_{kj}) \quad \text{go upward in } j$$


How do we minimize roundoff error?

- Very important trick: reorder rows of A before starting so that column 1 elements decrease with increasing i
- This minimizes roundoff error in the early stages and keeps its growth to a minimum

Computational cost of solving linear system

- To find any U_{ij} requires $2(i-1)$ operations
(1 addition plus 1 multiplication per term in the sum)

Total computation of U requires $\sum_{j=1}^N \sum_{i=1}^j 2(i-1) \sim 1/3 N^3$ operations

- Similarly, to find any L_{ij} requires $2(j-1) + 1$ operations \rightarrow total computation of L also requires $\sim 1/3 N^3$ operations 

- The final substitutions require $\sim N^2$ operations
(negligible additional amount for large N)

Summary of solution to $\mathbf{A}\mathbf{x} = \mathbf{b}$ using LU decomposition

- 1) Find $\mathbf{L}\mathbf{U} = \mathbf{A}$ ($\sim \frac{2}{3} N^3$ operations)
- 2) Solve $\mathbf{L}\mathbf{y} = \mathbf{b}$ for \mathbf{y} ($\sim \frac{1}{2} N^2$ operations)
- 3) Solve $\mathbf{U}\mathbf{x} = \mathbf{y}$ for \mathbf{x} ($\sim \frac{1}{2} N^2$ operations)

Iterative improvement

(correcting the solution for residual roundoff error)

- Let \mathbf{x} be the exact solution to $\mathbf{A}\mathbf{x} = \mathbf{b}$ and let $\mathbf{x} + \delta\mathbf{x}$ be the result of our numerical calculation
- When we actually substitute $\mathbf{x} + \delta\mathbf{x}$ into the original equation, we get something that isn't exactly \mathbf{b} ; call the result $\mathbf{b} + \delta\mathbf{b}$

$$\mathbf{A}(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b} + \mathbf{A}\delta\mathbf{x} = \mathbf{b} + \delta\mathbf{b}$$

- Thus we can estimate the error $\delta\mathbf{x}$ by solving $\mathbf{A}\delta\mathbf{x} = \delta\mathbf{b}$
This can be done very quickly since we already have the LU decomposition of \mathbf{A} !

Matrix inversion and determinants

- Once we have LU-decomposed \mathbf{A} , it is straightforward to obtain \mathbf{A}^{-1} , which is just the solution to

$$\mathbf{A} \mathbf{A}^{-1} = \mathbf{I}$$

We can obtain this column by column by solving $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ with

$$\mathbf{b} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \quad \text{etc}$$

- The determinant of \mathbf{A} is simply

$$|\mathbf{A}| = |\mathbf{L}| \cdot |\mathbf{U}| = (\prod L_{ii}) (\prod U_{ii}) = (\prod U_{ii})$$