Linear Algebraic Equations and Matrices

Berlin Chen
Department of Computer Science & Information Engineering
National Taiwan Normal University

Reference:

1. *Applied Numerical Methods with MATLAB for Engineers*, Chapter 8 & Teaching material

Chapter Objectives

- Understanding matrix notation
- Being able to identify the following types of matrices: identity, diagonal, symmetric, triangular, and tridiagonal
- Knowing how to perform matrix multiplication and being able to assess when it is feasible
- Knowing how to represent a system of linear equations in matrix form
- Knowing how to solve linear algebraic equations with left division and matrix inversion in MATLAB

Three Bungee Jumpers (1/2)

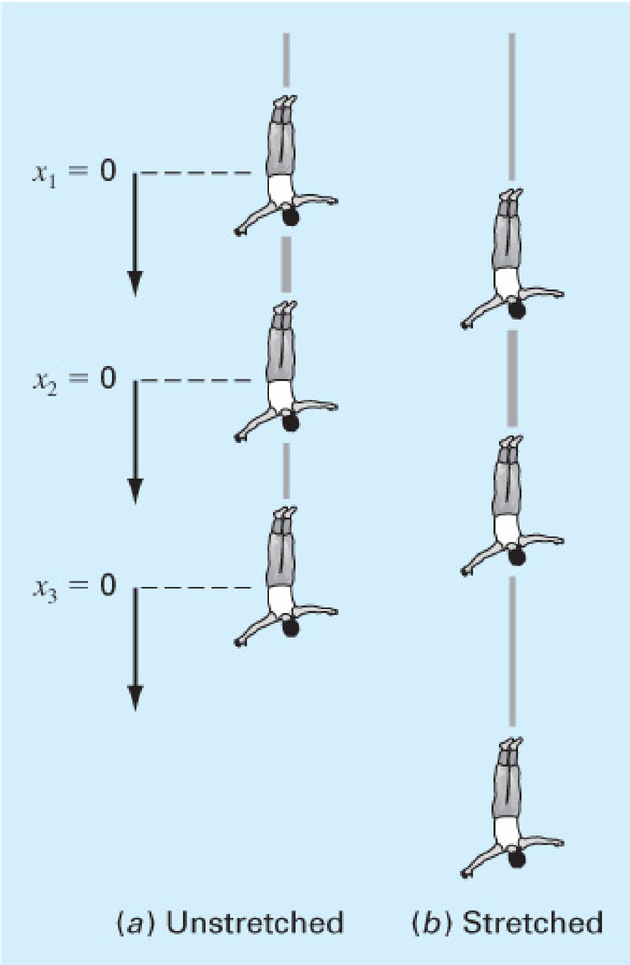


FIGURE 8.1
Three individuals connected by bungee cords.

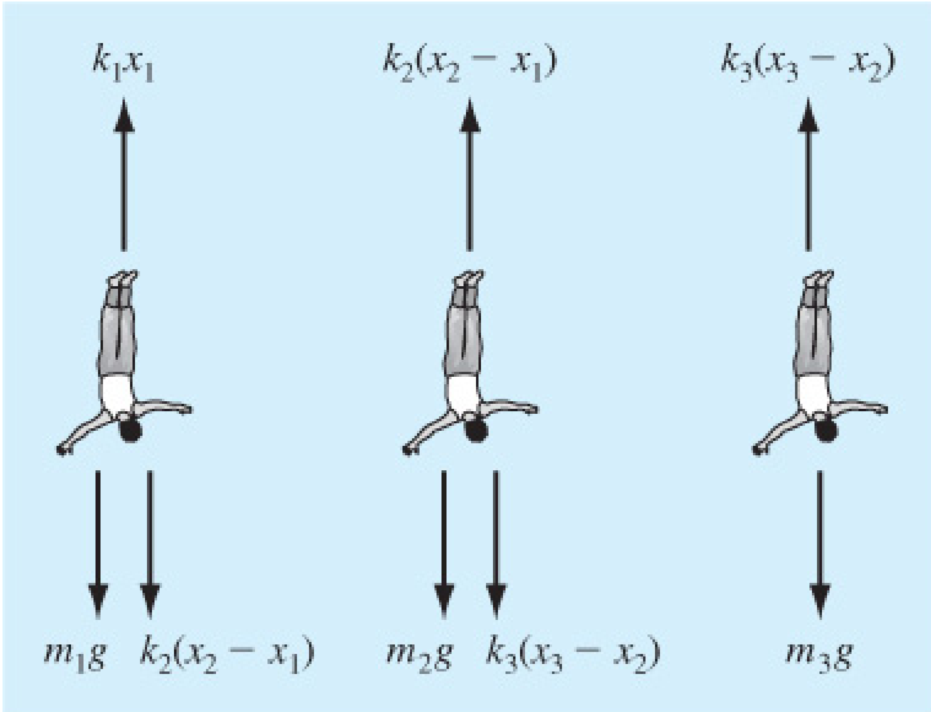


FIGURE 8.2
Free-body diagrams.

compute the displacement of each of the jumpers when coming to the equilibrium positions

Three Bungee Jumpers (2/2)

Using Newton's second law, force balances can be written for each jumper:

$$
\begin{aligned}
m_1 \frac{d^2 x_1}{dt^2} &= m_1 g + k_2(x_2 - x_1) - k_1 x_1 \\
m_2 \frac{d^2 x_2}{dt^2} &= m_2 g + k_3(x_3 - x_2) + k_2(x_1 - x_2) \\
m_3 \frac{d^2 x_3}{dt^2} &= m_3 g + k_3(x_2 - x_3)
\end{aligned}
\tag{8.1}
$$

where m_i = the mass of jumper i (kg), t = time (s), k_j = the spring constant for cord j (N/m), x_i = the displacement of jumper i measured downward from the equilibrium position (m), and g = gravitational acceleration (9.81 m/s^2). Because we are interested in the steady-state solution, the second derivatives can be set to zero. Collecting terms gives

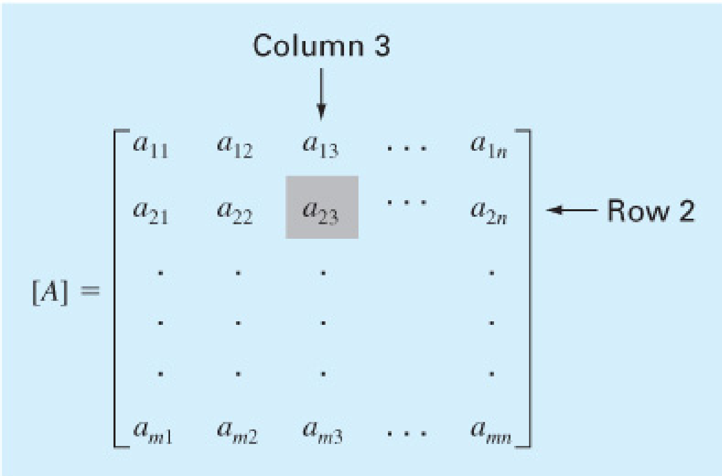
$$
\begin{aligned}
(k_1 + k_2)x_1 \quad - k_2 x_2 \quad &= m_1 g \\
-k_2 x_1 + (k_2 + k_3)x_2 - k_3 x_3 &= m_2 g \\
-k_3 x_2 + k_3 x_3 &= m_3 g
\end{aligned}
\tag{8.2}
$$

Thus, the problem reduces to solving a system of three simultaneous equations for the three unknown displacements. Because we have used a linear law for the cords, these equations are linear algebraic equations. Chapters 8 through 12 will introduce you to how MATLAB is used to solve such systems of equations.

Overview (1/2)

- A *matrix* consists of a rectangular array of elements represented by a single symbol (example: [*A*])
- An individual entry of a matrix is an ***element*** (example: a_{23})

FIGURE 8.3
A matrix.



[1] In addition to special brackets, we will use case to distinguish between vectors (lowercase) and matrices (uppercase).

Overview (2/2)

- A horizontal set of elements is called a ***row*** and a vertical set of elements is called a ***column***
- The first subscript of an element indicates the row while the second indicates the column
- The size of a matrix is given as *m* rows by *n* columns, or simply *m* by *n* (or *m* x *n*)
- 1 x *n* matrices are ***row vectors***
- *m* x 1 matrices are ***column vectors***

Special Matrices

- Matrices where *m*=*n* are called ***square matrices***
- There are a number of special forms of square matrices:

Symmetric

$$[A]=\begin{bmatrix} 5 & 1 & 2 \\ 1 & 3 & 7 \\ 2 & 7 & 8 \end{bmatrix}$$

Diagonal

$$[A]=\begin{bmatrix} a_{11} & & \\ & a_{22} & \\ & & a_{33} \end{bmatrix}$$

Identity

$$[A]=\begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix}$$

Upper Triangular

$$[A]=\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ & a_{22} & a_{23} \\ & & a_{33} \end{bmatrix}$$

Lower Triangular

$$[A]=\begin{bmatrix} a_{11} & & \\ a_{21} & a_{22} & \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

Banded

$$[A]=\begin{bmatrix} a_{11} & a_{12} & & \\ a_{21} & a_{22} & a_{23} & \\ & a_{32} & a_{33} & a_{34} \\ & & a_{43} & a_{44} \end{bmatrix}$$

Matrix Operations

- Two matrices are considered equal if and only if every element in the first matrix is equal to every corresponding element in the second
  - This means the two matrices must be the same size
- Matrix addition and subtraction are performed by adding or subtracting the corresponding elements
  - This requires that the two matrices be the same size
- Scalar matrix multiplication is performed by multiplying each element by the same scalar

Matrix Multiplication

- The elements in the matrix [C] that results from multiplying matrices [A] and [B] are calculated using:

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

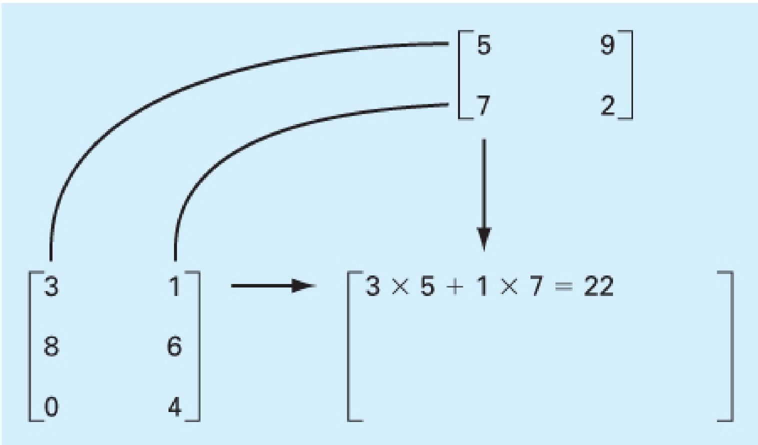


FIGURE 8.4
Visual depiction of how the rows and columns line up in matrix multiplication.

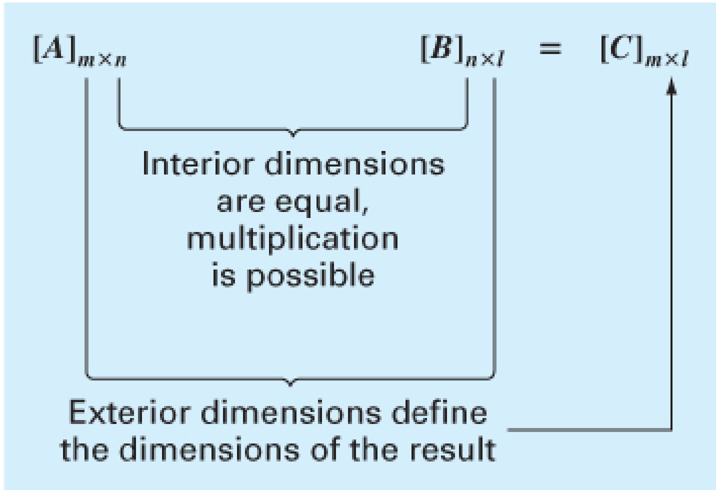


FIGURE 8.5
Matrix multiplication can be performed only if the inner dimensions are equal.

Matrix Inverse and Transpose

- The *inverse* of a square, nonsingular matrix $[A]$ is that matrix which, when multiplied by $[A]$, yields the identity matrix
  - $[A][A]^{-1}=[A]^{-1}[A]=[I]$

The inverse of a 2 × 2 matrix can be represented simply by

$$[A]^{-1} = \frac{1}{a_{11}a_{22} - a_{12}a_{21}} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix}$$

- The *transpose* of a matrix involves transforming its rows into columns and its columns into rows
  - $(a_{ij})^T=a_{ji}$

$$[A] = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

the transpose, designated $[A]^T$, is defined as

$$[A]^T = \begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \\ a_{13} & a_{23} & a_{33} \end{bmatrix}$$

Representing Linear Algebra

- Matrices provide a concise notation for representing and solving simultaneous linear equations:

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= b_1 \\
a_{21}x_1 + a_{22}x_2 + a_{23}x_3 &= b_2 \\
a_{31}x_1 + a_{32}x_2 + a_{33}x_3 &= b_3
\end{aligned}
$$

$$
\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}
\begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix}
=
\begin{Bmatrix} b_1 \\ b_2 \\ b_3 \end{Bmatrix}
$$

$$[A]\{x\} = \{b\}$$

Solving With MATLAB (1/2)

- MATLAB provides two direct ways to solve systems of linear algebraic equations [A]{x}={b}:
  - Left-division
    `x = A\b`
  - Matrix inversion
    `x = inv(A)*b`
- The matrix inverse is less efficient than left-division and also only works for square, non-singular systems

Solving With MATLAB (2/2)

$$\begin{bmatrix} 150 & -100 & 0 \\ -100 & 150 & -50 \\ 0 & -50 & 50 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 588.6 \\ 686.7 \\ 784.8 \end{Bmatrix}$$

Start up MATLAB and enter the coefficient matrix and the right-hand-side vector:

```
>> K = [150 -100 0;-100 150 -50;0 -50 50]

K =
   150  -100     0
  -100   150   -50
     0   -50    50

>> mg = [588.6; 686.7; 784.8]

mg =
  588.6000
  686.7000
  784.8000
```

Employing left division yields

```
>> x = K\mg

x =
   41.2020
   55.9170
   71.6130
```

Alternatively, multiplying the inverse of the coefficient matrix by the right-hand-side vector gives the same result:

```
>> x = inv(K)*mg

x =
   41.2020
   55.9170
   71.6130
```