Lecture 10

Linear algebra

1 Introduction

Engineers deal with many *vector* quantities, such as forces, positions, velocities, heat flow, stress and strain, gravitational and electric fields and on and on. In this lecture we want to review basic concepts and operations on vectors. We will see how a linear system of equations can naturally arise from physical constraints on linear combinations of vectors, and how the resulting bookkeeping naturally leads to the idea of a matrix and matrix-vector equations.

2 Vectors

Abstractly a vector is simply a one-dimensional array of numbers. The *dimension* of the vector is the number of elements in the array. When arranged vertically we call this a *column vector*. The following are three-dimensional column vectors

$$\mathbf{u}=\begin{pmatrix}u_1\\u_2\\u_3\end{pmatrix}\qquad \mathbf{v}=\begin{pmatrix}4\\-3\\1\end{pmatrix}$$

A horizontal arrangement is called a *row vector*, such as

$$\mathbf{w}=(-11,7,2)\qquad \mathbf{r}=(x,y,z)$$

In engineering applications a vector almost always represents a physical quantity that has both *magnitude* and *direction*. The elements of the array are the *components* of the vector along the corresponding coordinate axis. It's very useful to visualize a vector as an arrow in space with the same direction and the arrow length representing the magnitude.

As an example, the two-dimensional vector

$$\mathbf{r}=\begin{pmatrix}3\\2\end{pmatrix}$$

can be graphically represented (Fig. 1) as an arrow from the origin to a point with rectangular coordinates $(3,2)$. We say **r** has component 3 in the x direction and component 2 in the y direction. Converting the *x,y* coordinates into polar form

$$\begin{aligned}x&=\rho\cos\phi\\ y&=\rho\sin\phi\\ \rho&=\sqrt{x^2+y^2}=\sqrt{13}\approx 3.61\\ \phi&=\tan^{-1}\frac{y}{x}=\tan^{-1}\frac{2}{3}\approx 33.7^\circ\end{aligned}$$

We identify the length ρ as the magnitude of the vector and ϕ as its direction (relative to the x axis).

Here's a potential source of confusion. There is not necessarily anything physically significant about the arrow's endpoint, $(3,2)$ in this case. Say this vector represents a force acting on a

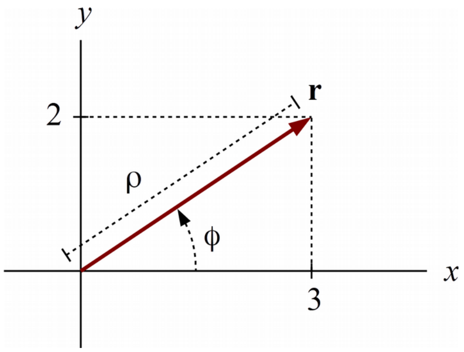

Fig. 1 Length and direction of a vector. In two dimensions these are just the polar coordinate representation of the vector components.

particle at the origin. Then the force exists at a single point, the origin. It does not exist at the point $(3,2)$ or anywhere except the origin. A force acting on a point particle has no extension in space. We are simply using a physical arrow to *visualize* the magnitude and direction of the force. The coordinates in this case might have units of newtons. On the other hand, suppose the vector **r** represents a displacement in which a particle originally at the origin is moved to the point $x=3, y=2$. In this case there *is* a physical significance to the arrow's endpoint. This is just something to keep in mind. We get so accustomed to representing physical quantities such as forces and velocities by arrows that it's easy to forget that they do not necessarily physically coincide with these arrows.

A vector does not have to “start” at the origin. Suppose a particle is at location $x=3\,\text{m}, y=2\,\text{m}$ and moving with velocity $v_x=2\,\text{m/s}, v_y=-1\,\text{m/s}$. We could illustrate this situation as shown in Fig. 2.

In this case it's more physically meaningful to put the tail of the **v** vector at the particle's location. Again, the location of the head of the **v** vector is not physically significant. In fact the **r** and **v** vectors don't even have the same units – m vs. m/s. However, *could* say that if the particle traveled at velocity **v** for 1 second it would end up at the head of the vector **v**, which is location $x=5, y=1$. In fact, *assuming* **v** remains constant, we could write the particle position as the vector

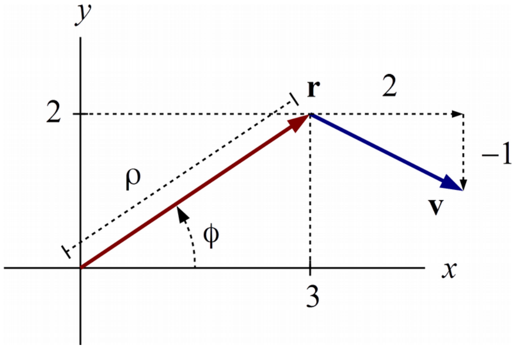

Fig. 2 Particle at position **r** *moving with velocity* **v**. *If the particle moved with constant velocity for 1 second it would arrive at the head of* **v**.

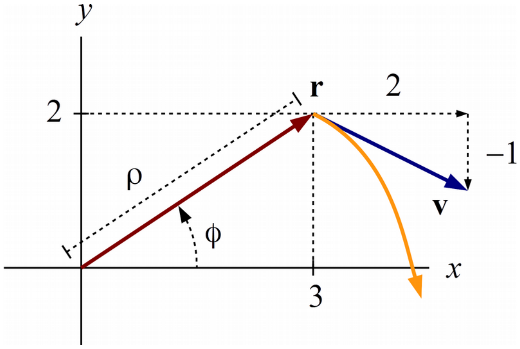

Fig. 3 If **v** *is an instantaneous velocity the particle does not follow the vector* **v** *for any extended length of time.*

$$\mathbf{r}(t)=\begin{pmatrix}x(t)\\y(t)\end{pmatrix}=\begin{pmatrix}3\\2\end{pmatrix}+\begin{pmatrix}2\\-1\end{pmatrix}t=\begin{pmatrix}3+2t\\2-t\end{pmatrix}$$

This illustrates the concept of vector addition. Often, however, **v** will represent the instantaneous velocity of a particle following a curved trajectory, as illustrated in Fig. 3. In this case the particle will not following the **v** vector (expect for an instant) or arrive at its head.

2.1 Vector norm (length)

In two- and three-dimensional geometry we have the Pythagorean theorem which gives use the length of a displacement with components *x,y* or *x,y,z* as

$$L=\sqrt{x^2+y^2} \text{ or } L=\sqrt{x^2+y^2+z^2}$$

We generalize this to an *n*-dimensional vector **u** to define the *vector norm* as

$$\|\mathbf{u}\|=\sqrt{u_1^2+u_2^2+\cdots+u_n^2} \tag{1}$$

Even though the “length” of a ten-dimensional vector doesn't have a direct physical meaning, the norm concept is very useful.

This formula is actually only one way to define a vector norm. The *p-norm* is defined as

$$\|\mathbf{u}\|_p=\left(|u_1|^p+|u_2|^p+\cdots+|u_n|^p\right)^{\frac{1}{p}}$$

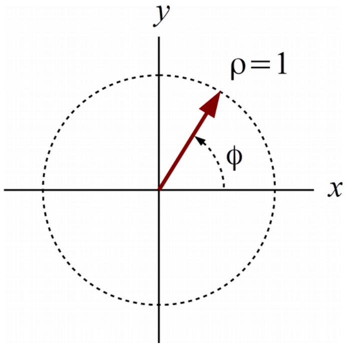

Fig. 4 unit vectors centered at the origin fall on the unit circle in 2D and on the unit sphere in 3D.

The *Euclidean norm* of the Pythagorean theorem would then be called the “2-norm.” When $p \to \infty$ we obtain the “infinity-norm”

$$\|\mathbf{u}\|_\infty = \max |u_k|$$

which has important applications in, among other things, control systems theory. Suppose the k^{th} element of **u** represents the distances traveled during the k^{th} segment of a trip. Then the 1-norm

$$\|\mathbf{u}\|_1 = |u_1| + |u_2| + \cdots + |u_n|$$

is just the total length of the trip. When vectors are represented by boldface letters the corresponding italic letter is often taken to represent the norm

$$w = \|\mathbf{w}\|$$

In both Scilab and Matlab the function `norm(x,p)` calculates the *p*-norm of vector **x**. For example

```
-->x = [1;2;3]
 x  =

    1.
    2.
    3.

-->norm(x,1)
 ans  =

    6.

-->norm(x,2)
 ans  =

    3.7416574

-->norm(x,'inf')
 ans  =

    3.

-->norm(x)
 ans  =

    3.7416574
```

Note that `norm(x)` gives the default 2-norm or Euclidean norm. In this class we will take “norm” to mean Euclidean norm, unless explicitly stated otherwise.

A vector with norm of 1 is called a *unit vector*. We can make any non-zero vector a unit vector by dividing it by its norm. Commonly a “hat” or the letter **a** with a subscript is used to denote a unit vector, for example

$$\mathbf{a}_{\mathbf{u}} = \hat{\mathbf{u}} = \frac{\mathbf{u}}{\|\mathbf{u}\|}$$

A unit vector represents a “pure direction.” In two and three dimensions this is literally a direction in space (Fig. 4), but in higher dimensions it's a “direction” only in an abstract sense. The concept is still very useful, however.

3 Scalar/inner/dot product

In three dimensions the *scalar product* (also called the *inner product* or *dot product*) of two vectors is

$$\mathbf{u}\cdot\mathbf{v}=u_1 v_1+u_2 v_2+u_3 v_3=u\,v\cos\theta_{\mathbf{uv}} \tag{2}$$

where $\theta_{\mathbf{uv}}$ is the angle between the vectors. The scalar product readily generalizes to n-dimensional vectors as

$$\mathbf{u}\cdot\mathbf{v}=\sum_{i=1}^{n} u_i v_i \tag{3}$$

We can still write

$$\mathbf{u}\cdot\mathbf{v}=u\,v\cos\theta_{\mathbf{uv}} \tag{4}$$

which *defines* the “angle” between **u** and **v** as

$$\cos\theta_{\mathbf{uv}}\equiv\frac{\mathbf{u}\cdot\mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|}=\hat{\mathbf{u}}\cdot\hat{\mathbf{v}} \tag{5}$$

Two vectors with a zero scalar product are said to be *orthogonal*. Since $\cos 90^{\circ}=0$ this means that the vectors form a “right angle,” they are “perpendicular.”

4 Vector/cross product

In three dimensions the *vector product* (also called the *cross product*) of two vectors is

$$\mathbf{w}=\mathbf{u}\times\mathbf{v}=\begin{pmatrix} u_2 v_3-u_3 v_2 \\ u_3 v_1-u_1 v_3 \\ u_1 v_2-u_2 v_1 \end{pmatrix} \tag{6}$$

The vector product is specific to three dimensions; it does not readily generalize to n dimensions. It is very important in many applications. For example torque about the origin is the cross product of force and position. The magnitude of the cross product is

$$w=u\,v\sin\theta_{\mathbf{uv}} \tag{7}$$

Since $\sin 0=0$ the vector product of parallel vectors is zero.

5 Matrix-vector product

Suppose we have two, two-dimensional vectors

$$\mathbf{u}=\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \qquad \mathbf{v}=\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \tag{8}$$

A certain displacement might be described as “travel x_1 times **u** followed by x_2 times **v** to end up at **y**.” Algebraically we have $x_1\mathbf{u}+x_2\mathbf{v}=\mathbf{y}$ or

$$x_1\begin{pmatrix} u_1 \\ u_2 \end{pmatrix}+x_2\begin{pmatrix} v_1 \\ v_2 \end{pmatrix}=\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \tag{9}$$

In terms of components

$$\begin{aligned} u_1 x_1 + v_1 x_2 &= y_1 \\ u_2 x_1 + v_2 x_2 &= y_2 \end{aligned} \tag{10}$$

For bookkeeping purposes we will write this as

$$\begin{pmatrix} u_1 & v_1 \\ u_2 & v_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \tag{11}$$

where the two-dimensional array is a *matrix*, the columns of which are the vectors **u** and **v**. Thinking of this array as a single entity **A** we can specify its elements using two indicies

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \tag{12}$$

This is just a different way to express the two equations

$$\begin{aligned} a_{11} x_1 + a_{12} x_2 &= y_1 \\ a_{21} x_1 + a_{22} x_2 &= y_2 \end{aligned} \tag{13}$$

Employing the notation

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \tag{14}$$

we can write (13) compactly as the matrix-vector equation

$$\mathbf{Ax} = \mathbf{y}$$

If **A** is an m-by-n matrix

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} \tag{15}$$

and **x** is an *n*-by-1 “matrix” (a column vector)

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \tag{16}$$

Then the product $\mathbf{y} = \mathbf{Ax}$ is an *m*-by-1 “matrix” (a column vector)

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} \tag{17}$$

with components

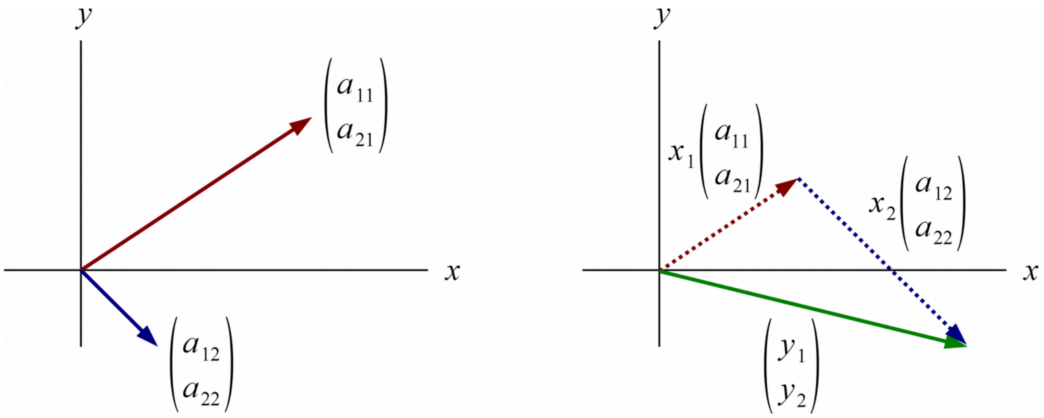
y
$\begin{pmatrix} a_{11} \\ a_{21} \end{pmatrix}$
x
$\begin{pmatrix} a_{12} \\ a_{22} \end{pmatrix}$
y
$x_1 \begin{pmatrix} a_{11} \\ a_{21} \end{pmatrix}$
$x_2 \begin{pmatrix} a_{12} \\ a_{22} \end{pmatrix}$
x
$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$

Fig. 5 The product **Ax=y** *can be thought of as summing scaled versions of the column vectors of* **A**.

$$y_i=\sum_{j=1}^{n} a_{ij} x_j \tag{18}$$

Note that the product **Ax** as defined by (18) only “works” if number of columns of **A** is equal to the number of elements of **x** (in this case both are *n*); each column of **A** gets multiplied by the corresponding element of **x**.

We are most often (but not always) interested in the case $m=n$ where the matrix **A** is *square*. In any case we can visualize the *linear system*

$$\mathbf{A}\mathbf{x}=\mathbf{y} \tag{19}$$

as (Fig. 5)

- think of each column of **A** as a vector
- scale the *j*th column by the factor x_j
- sum up all the scaled vectors to get **y**

In this visualization we assume that **x** is known and we want to calculate **y**. Often we are faced with the *inverse problem* where **y** is known and we want to calculate **x**. We formally write

$$\mathbf{x}=\mathbf{A}^{-1}\mathbf{y} \tag{20}$$

where $\mathbf{A}^{-1}$ is the *inverse* of matrix **A**. Solving this problem will be the topic of the next lecture. For now we want to motivate our study of linear systems of equations by considering two important problems which give rise to such systems.

6 Two-dimensional truss problem

In Civil Engineering a classic problem is to design a bridge structure that can withstand some specified loads (Fig. 6). In its simplified form this leads to the two-dimensional truss problem (Fig. 7). We assume that the structure consists of a number of linear *members* of negligible mass which are connected at their ends by frictionless pins at various *joints*. The members are either in compression or tension, so they either push or pull on the joints. External forces may be applied at the joints. Two joints, that we will take to be joints 1 and 2, connect the structure to the ground, and at these joints the ground exerts *reaction forces*. The truss problem is to calculate the member compression/tension forces and the reaction forces, given the truss geometry and

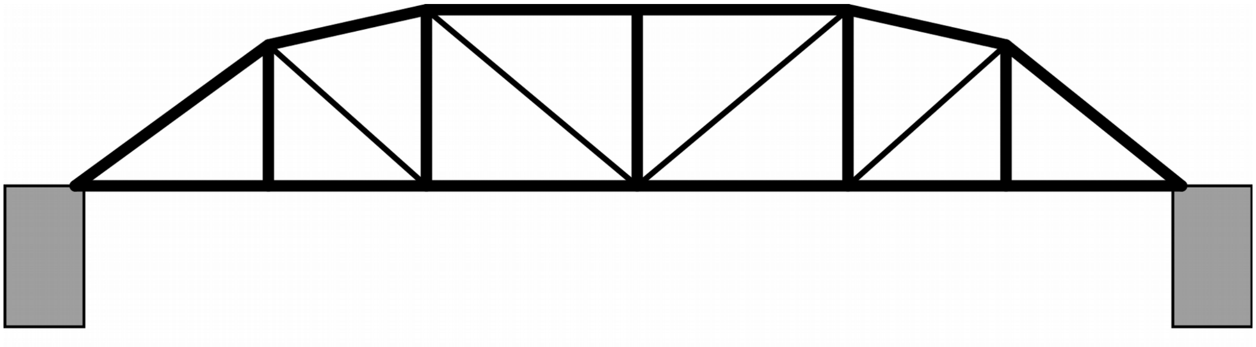

Fig. 6 A truss bridge formed of "members" connected at "joints."

external applied forces. The conditions for static equilibrium are that the net vector force at each joint is zero (the so-called "method of joints"). In two dimensions this gives us two equations

$$\begin{aligned}F_x^{\text{net}}&=0\\F_y^{\text{net}}&=0\end{aligned}\tag{21}$$

at each joint.

To be specific let's take the system illustrated in Fig. 7. There are four joints located at $(x_k,y_k),\ k=1,2,3,4$ and five members with compression forces $u_k,\ k=1,2,3,4,5$. An external force is applied at joint 4 with components (f_{4x},f_{4y}). A reaction force (due to the mounting of the system) is applied at joint 1 with components (r_{1x},r_{1y}) and a reaction force with y component r_{2y} is applied at joint 2.

Assuming the force (f_{4x},f_{4y}) is known, there are eight unknowns which form the components of an eight-dimensional vector:

$$\mathbf{u}=\begin{pmatrix}u_1\\u_2\\u_3\\u_4\\u_5\\u_6=r_{x1}\\u_7=r_{y1}\\u_8=r_{y2}\end{pmatrix}\tag{22}$$

This bookkeeping is an example of how higher-dimensional spaces arise. Our vector is a collection of various (arbitrarily arranged) physical parameters as opposed to a three-dimensional force or velocity with a direct physical significance.

A given member pushes on a given joint in a direction determined by the member's endpoints. For example, member m_3 pushes on joint j_2 at an angle θ_{23} where

$$\tan\theta_{23}=\frac{y_2-y_3}{x_2-x_3}\tag{23}$$

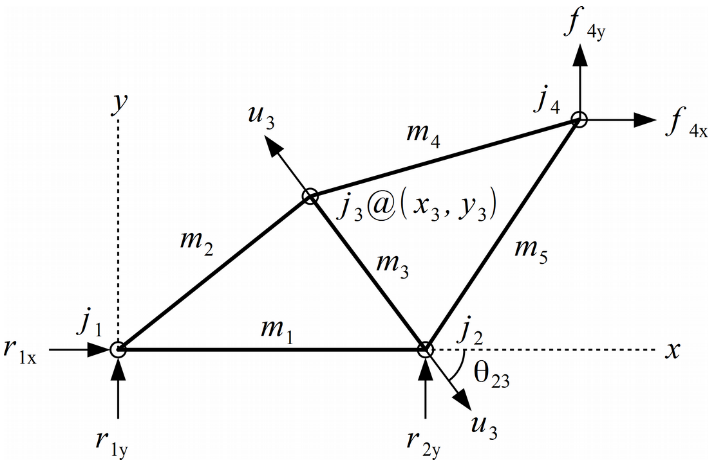

Fig. 7: Geometry of the truss problems. Member m_i has compression force u_i and connects to other members at some joints j_k. Applied forces f and reaction forces r also acts on two or more joints.

In general we will define

$$\tan\theta_{ij} = \frac{y_i - y_j}{x_i - x_j} \tag{24}$$

The equations of equilibrium are now as follows.

At j_1

$$\begin{gathered} u_1\cos\theta_{12} + u_2\cos\theta_{13} + u_6 = 0 \\ u_1\sin\theta_{12} + u_2\sin\theta_{13} + u_7 = 0 \end{gathered} \tag{25}$$

At j_2

$$\begin{gathered} u_1\cos\theta_{21} + u_3\cos\theta_{23} + u_5\cos\theta_{24} = 0 \\ u_1\sin\theta_{21} + u_3\sin\theta_{23} + u_5\sin\theta_{24} + u_8 = 0 \end{gathered} \tag{26}$$

At j_3

$$\begin{gathered} u_2\cos\theta_{31} + u_3\cos\theta_{32} + u_4\cos\theta_{34} = 0 \\ u_2\sin\theta_{31} + u_3\sin\theta_{32} + u_4\sin\theta_{34} = 0 \end{gathered} \tag{27}$$

At j_4

$$\begin{gathered} u_4\cos\theta_{43} + u_5\cos\theta_{42} + f_{4x} = 0 \\ u_4\sin\theta_{43} + u_5\sin\theta_{42} + f_{4y} = 0 \end{gathered} \tag{28}$$

Putting these together into an eight-by-eight linear system we have

$$\mathbf{A}\,\mathbf{u} = \mathbf{b} \tag{29}$$

with

$$\mathbf{A}=\begin{pmatrix} \cos\theta_{12} & \cos\theta_{13} & 0 & 0 & 0 & 1 & 0 & 0 \\ \sin\theta_{12} & \sin\theta_{13} & 0 & 0 & 0 & 0 & 1 & 0 \\ \cos\theta_{21} & 0 & \cos\theta_{23} & 0 & \cos\theta_{24} & 0 & 0 & 0 \\ \sin\theta_{21} & 0 & \sin\theta_{23} & 0 & \sin\theta_{24} & 0 & 0 & 1 \\ 0 & \cos\theta_{31} & \cos\theta_{32} & \cos\theta_{34} & 0 & 0 & 0 & 0 \\ 0 & \sin\theta_{31} & \sin\theta_{32} & \sin\theta_{34} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \cos\theta_{43} & \cos\theta_{42} & 0 & 0 & 0 \\ 0 & 0 & 0 & \sin\theta_{43} & \sin\theta_{42} & 0 & 0 & 0 \end{pmatrix} \tag{30}$$

and

$$\mathbf{b}=\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -f_{4x} \\ -f_{4y} \end{pmatrix} \tag{31}$$

Notice that the applied forces shown up in the **b** (the "knowns") vector while the member and reaction forces form the **u** vector (the "unknowns").

From a programming perspective, the challenge would be to generalize this process to allow the solution of an arbitrary truss problem. This would mostly involve figuring out a systematic way to do the bookkeeping involved in forming the **A** matrix and the **b** vector.

7 Laplace's equation in two dimensions

A very important equation in engineering analysis is *Laplace's equation*

$$\nabla^2 f=0$$

where f is a continuous function of space such as $f(x,y)$. This *field* might represent, say, temperature distribution over the surface of a metal plate. A solution of Laplace's equation has the property that the value of f at any point is equal to the average value of f at neighboring points. This is why it comes up so often – in equilibrium Nature usually "wants" physical quantities (temperature, pressure, electrical potential) to be as "smooth" or as "averaged-out" as possible.

One approach to solving Laplace's equation numerically is to specify u at a discrete grid of points and require that f at each point be equal to the average of f at neighboring points. Let's consider a specific two-dimensional case illustrated in Fig. 8. The indicies *i,j* specify *x,y* location, respectively. The condition that f at location *i,j* is equal to the average of f at its four nearest-neighbor points is

$$f_{i,j}=\frac{1}{4}\left(f_{i+1,j}+f_{i-1,j}+f_{i,j+1}+f_{i,j-1}\right) \tag{32}$$

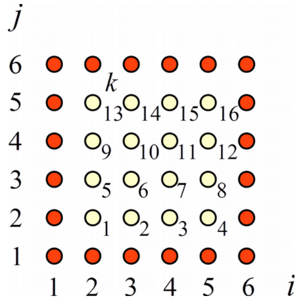

Fig. 8: Two-dimensional grid for solving Laplace's equation.

or equivalently

$$4f_{i,j}-f_{i+1,j}-f_{i-1,j}-f_{i,j+1}-f_{i,j-1}=0 \tag{33}$$

We assume the *f* values on the *boundary* are specified. These form the *boundary conditions*. Our task is then to calculate the interior *f* values such that Laplace's equation is satisfied.

To use our matrix-vector formalism (29) the unknowns need to be arranged in a one-dimensional column vector. As illustrated (Fig. 8), one way to do this is to number the interior points from 1 to 16 as unknowns

$$u_k=f_{i,j} \text{ where } k=4(j-2)+(i-1)$$

Just running through the 16 points and considering (33) we can, by inspection, obtain a 16-by-16 linear system of the form (29) with

$$\mathbf{A}=\begin{pmatrix}
4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
-1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 1 & 4 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & -1 & 4 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
-1 & 0 & 0 & 0 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 1 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 1 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & 0 & 0 & 0 & -1\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 4 & -1 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4
\end{pmatrix} \tag{34}$$

and

$$\mathbf{b}=\begin{pmatrix}
f_{1,2}+f_{2,1}\\
f_{3,1}\\
f_{4,1}\\
f_{5,1}\\
f_{1,3}\\
0\\
0\\
f_{6,3}\\
f_{1,4}\\
0\\
0\\
f_{6,4}\\
f_{1,5}+f_{2,6}\\
f_{3,6}\\
f_{4,6}\\
f_{5,6}+f_{6,5}
\end{pmatrix} \tag{35}$$

The k^{th} row of this system is a statement of (33) for unknown u_k. This illustrates a few things. First, the dimension *n* of our linear system is determined by the number of unknowns, not by the 2 or 3 dimensions of physical space. This number can easily be *very* large. Suppose we wanted to have a 100-by-100 two-dimensional grid of unknown field values. This is not actually very large, after all, a 100-by-100 pixel image is essentially a “thumbnail.” Yet this results in $n=10{,}000$ unknowns and a matrix **A** that is 10,000-by-10,000 in size. In three dimensions the problem dimension would be $n=100^3=1{,}000{,}000$, and our matrix would have $(10^6)^2$ or one trillion entries! Yet these are often the size of problems we need to solve in engineering applications.

Second, and fortunately for us, a glance at (34) and consideration of the way it was built using (33) reveals that the great majority of entries in **A** will be zeros. We say that **A** is a *sparse matrix*. So, even if it does contain a trillion elements, only a tiny fraction are non-zero. Sparse matrix techniques exploit this fact to store and manipulate such matrices using orders-or-magnitude less resources than would be needed for "dense" matrices, and they enable use to solve physically significant problems using available computing power. We will take a look at sparse matrix techniques in a later lecture.