are the columns of the identity matrix. We know the inverse $\mathbf{B}$ satisfies

$$\mathbf{AB} = \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \mathbf{e}_3 & \cdots & \mathbf{e}_n \\ \downarrow & \downarrow & \downarrow & & \downarrow \end{bmatrix}.$$

Let $\mathbf{b}_1$, $\mathbf{b}_2$, ..., $\mathbf{b}_n$ denote the columns of $\mathbf{B}$. Then an equivalent problem to finding the matrix $\mathbf{B}$ is to solve the n problems

$$\mathbf{Ab}_1 = \mathbf{e}_1 \ , \quad \mathbf{Ab}_2 = \mathbf{e}_2 \ , \quad \ldots \quad \mathbf{Ab}_n = \mathbf{e}_n \ ,$$

for the unknown vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$. By the method above, we would augment $\mathbf{A}$ with the full n-by-n identity matrix and perform elementary operations until the part to the left of the augmentation line was in row echelon form. That is, we would reduce $[\mathbf{A}|\mathbf{I}]$ to $[\mathbf{R}|\mathbf{C}]$, where $\mathbf{C}$ is an n-by-n matrix. (Note that if $\mathbf{R}$ does not have n pivots, then $\mathbf{A}$ is singular.) We can then solve each of the problems

$$\mathbf{Rb}_1 = \mathbf{c}_1, \quad \mathbf{Rb}_2 = \mathbf{c}_2, \quad \ldots \quad \mathbf{Rb}_n = \mathbf{c}_n$$

using backward substitution, and arrive at the inverse matrix $\mathbf{B}$ putting the solutions together:

$$\mathbf{B} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_n \\ \downarrow & \downarrow & & \downarrow \end{bmatrix}.$$

That's a clever method, and it's pretty much along the lines of how an inverse matrix is found when it is really desired. However, in most cases, $\mathbf{A}^{-1}$ is just an intermediate find on the way to solving a matrix problem $\mathbf{Ax} = \mathbf{b}$ for $\mathbf{x}$. If there is a more efficient way to find $\mathbf{x}$, one requiring fewer calculations, we would employ it instead. That is the content of the next section.

1.6 *LU* Factorization of a Matrix

We have three 'legal' elementary operations when using Gaussian elimination to solve the equation $\mathbf{Ax} = \mathbf{b}$. We seek to put the matrix $\mathbf{A}$ in *echelon form* via a sequence of operations consisting of

1. multiplying a row by a nonzero constant.
2. exchanging two rows.
3. adding a multiple of one row to another.

You may have noticed that, at least in theory, reduction to echelon form may be accomplished without ever employing operation 1. Let us focus on operation 3 for the moment. In practice the multiplier is always some nonzero constant β. Moreover, in Gaussian elimination we are primarily concerned with adding a multiple of a row to some other row which is *below* it. For a fixed β, let $\mathbf{E}_{ij} = \mathbf{E}_{ij}(\beta)$ be the matrix that only differs from the m-by-m identity matrix in that its $(i, j)^{\text{th}}$ entry is β. We call $\mathbf{E}_{ij}$ an **elementary matrix**. A user-defined function written in Octave code that returns such a matrix might look like the following:

```
function emat = elementary(m, i, j, val)
  emat = eye(m,m);
  emat(i,j) = val;
endfn
```

In Exercise 1.26 you are asked to show that

$$\mathbf{E}_{ij}(\beta)\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ \vdots & \vdots & & \vdots \\ a_{i-1,1} & a_{i-1,2} & \cdots & a_{i-1,n} \\ a_{i,1} + \beta a_{j,1} & a_{i,2} + \beta a_{j,2} & \cdots & a_{i,n} + \beta a_{j,n} \\ a_{i+1,1} & a_{i+1,2} & \cdots & a_{i+1,n} \\ \vdots & \vdots & & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix}.$$

In other words, pre-multiplication by $\mathbf{E}_{ij}$ performs an instance of operation 3 on the matrix $\mathbf{A}$, replacing row i with (row i) $+\beta$ (row j). Now, suppose $a_{11} \neq 0$. Then $\mathbf{E}_{21}(-a_{21}/a_{11})\mathbf{A}$ is a matrix whose entry in its $(2, 1)^{\text{th}}$ position has been made to be zero. More generally,

$$\mathbf{E}_{n1}\left(\frac{-a_{n1}}{a_{11}}\right) \cdots \mathbf{E}_{31}\left(\frac{-a_{31}}{a_{11}}\right)\mathbf{E}_{21}\left(\frac{-a_{21}}{a_{11}}\right)\mathbf{A}$$

is the matrix that results from retaining the first pivot of $\mathbf{A}$ and eliminating all entries below it. If our matrix $\mathbf{A}$ is such that no row exchanges occur during reduction to echelon form, then by a sequence of pre-multiplications by elementary matrices, we arrive at an upper-triangular matrix $\mathbf{U}$. We make the following observations:

- Each time we perform elementary operation 3 via pre-multiplication by an elementary matrix $\mathbf{E}_{ij}$, it is the case that $i > j$. Thus, the elementary matrices we use are lower-triangular.
- Each elementary matrix is invertible, and $\mathbf{E}_{ij}^{-1}$ is lower triangular when $\mathbf{E}_{ij}$ is. See Exercise 1.26.
- The product of lower triangular matrices is again lower triangular. See Exercise 1.28.

By these observations, when no row exchanges take place in the reduction of $\mathbf{A}$ to echelon form, we may amass the sequence of elementary matrices which achieve this reduction into a single matrix $\mathbf{M}$ which is lower-triangular. Let us denote the inverse of $\mathbf{M}$ by $\mathbf{L}$, also a lower-triangular matrix. Then

$$\mathbf{LM} = \mathbf{I}, \qquad \text{while} \qquad \mathbf{MA} = \mathbf{U},$$

where **U** is an upper-triangular matrix, an echelon form for **A**. Thus,

$$\mathbf{A} \;=\; (\mathbf{LM})\mathbf{A} \;=\; \mathbf{L}(\mathbf{MA}) \;=\; \mathbf{LU},$$

which is called the *LU* **factorization of A**, and

$$\mathbf{Ax} \;=\; \mathbf{b} \qquad \Leftrightarrow \qquad \mathbf{LUx} = \mathbf{b}\,.$$

Let $\mathbf{y} = \mathbf{Ux}$, so that $\mathbf{Ly} = \mathbf{b}$. Since **L** is lower-triangular, we may solve for **y** by a process known as **forward substitution**. Once we have **y**, we may solve for $\mathbf{Ux} = \mathbf{y}$ via backward substitution as in the previous section.

But let us not forget that the previous discussion was premised on the idea that no row exchanges take place in order to reduce A to echelon form. We are aware that, in some instances, row exchanges are absolutely necessary to bring a pivot into position. As it turns out, numerical considerations sometimes call for row exchanges even when a pivot would be in place without such an exchange. How does this affect the above discussion?

Suppose we can know in advance just which row exchanges will take place in reducing **A** to echelon form. With such knowledge, we can quickly write down an m-by-m matrix **P**, called a **permutation matrix**, such that **PA** is precisely the matrix **A** except that all of those row exchanges have been carried out. For instance, if we ultimately want the 1st row of **A** to wind up as row 5, we make the the 5th row of **P** be $(1, 0, 0, \ldots, 0)$. More generally, if we want the i^{th} row of **A** to wind up as the j^{th} row, we make the j^{th} row of **P** have a 1 in the i^{th} column and zeros everywhere else. To illustrate this, suppose

$$\mathbf{P} \;=\; \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

Then, for any 4-by-n matrix **A**, **PA** will be another 4-by-n whose 1^{st} row is equal to the 2^{nd} row of **A**, whose 2^{nd} row equals the 4^{th} row of **A**, whose 3^{rd} row equals the 3^{rd} row of **A**, and whose 4^{th} row equals the 1^{st} row of **A**.

Now, the full story about the *LU* decomposition can be told. There is a permutation matrix **P** such that **PA** will not need any row exchanges to be put into echelon form. It is this **PA** which has an *LU* decomposition. That is, $\mathbf{PA} = \mathbf{LU}$.

Example 10:

In Octave, the following commands were entered with accompanying output:

```
octave:1> A = [0 -1 3 1; 2 -1 1 4; 1 3 1 -1];
octave:2> [L,U,P] = lu(A)
L =
   1.00000   0.00000   0.00000
   0.50000   1.00000   0.00000
   0.00000  -0.28571   1.00000
```

```
U =
    2.00000   -1.00000    1.00000    4.00000
    0.00000    3.50000    0.50000   -3.00000
    0.00000    0.00000    3.14286    0.14286

P =
   0   1   0
   0   0   1
   1   0   0
```

We will use it to solve the matrix equation **Ax** = **b**, with

$$\mathbf{A} = \begin{bmatrix} 0 & -1 & 3 & 1 \\ 2 & -1 & 1 & 4 \\ 1 & 3 & 1 & -1 \end{bmatrix} \qquad \text{and} \qquad \mathbf{b} = \begin{bmatrix} -1 \\ 14 \\ 1 \end{bmatrix}.$$

Since we have been given the *LU* decomposition for **PA**, we will use it to solve **PAx** = **Pb**—that is, solve

$$\begin{bmatrix} 2 & -1 & 1 & 4 \\ 1 & 3 & 1 & -1 \\ 0 & -1 & 3 & 1 \end{bmatrix} \mathbf{x} = \begin{bmatrix} 14 \\ 1 \\ -1 \end{bmatrix}.$$

We first solve **Ly** = **Pb**, or[3]

$$\begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ 0 & -2/7 & 1 \end{bmatrix} \mathbf{y} = \begin{bmatrix} 14 \\ 1 \\ -1 \end{bmatrix}.$$

We call our manner of solving for **y** **forward substitution** because we find the components of **y** in forward order, y_1 then y_2 then y_3.

$$\begin{aligned} y_1 &= 14\,, \\ \frac{1}{2}y_1 + y_2 &= 1 \qquad \Rightarrow \qquad y_2 = -6\,, \\ -\frac{2}{7}y_2 + y_3 &= -1 \qquad \Rightarrow \qquad y_3 = -\frac{19}{7}\,, \end{aligned}$$

so **y** = (14, −6, −19/7). Now we solve **Ux** = **y**, or

$$\begin{bmatrix} 2 & -1 & 1 & 4 \\ 0 & 7/2 & 1/2 & -3 \\ 0 & 0 & 22/7 & 1/7 \end{bmatrix} \mathbf{x} = \begin{bmatrix} 14 \\ -6 \\ -19/7 \end{bmatrix},$$

[3]Asking Octave to display $7 * \mathbf{L}$ shows that this is an exact representation of **L**.

via backward substitution. The result is infinitely many solutions, all with the form

$$\mathbf{x} \;=\; \begin{bmatrix} 73/11 \\ -35/22 \\ -19/22 \\ 0 \end{bmatrix} + t \begin{bmatrix} -17/11 \\ 19/22 \\ -1/22 \\ 1 \end{bmatrix} , \qquad t \in \mathbb{R} .$$

■

Of course, it is possible to automate the entire process—not just the part of finding the *LU*-factorization of **A**, but also the forward and backward substitution steps. And there are situations in which, for a given coefficient matrix **A**, a different kind of solution process for the matrix equation $\mathbf{Ax} = \mathbf{b}$ may, indeed, be more efficient than using the factorization $\mathbf{LU} = \mathbf{PA}$. The Octave command

```
octave> A \ b
ans =
   2.30569
   0.82917
  -0.99101
   2.80220
```

(with `A` and `b` defined as in the Example 10) is sophisticated enough to look over the matrix `A` and choose a suitable solution technique, producing a result. In fact, the solution generated by the command is one that lies along the line of solutions

$$\mathbf{x} \;=\; \left(\frac{73}{11}, -\frac{35}{22}, -\frac{19}{22}, 0\right) + t\left(-\frac{17}{11}, \frac{19}{22}, -\frac{1}{22}, 1\right) , \qquad t \in \mathbb{R} ,$$

found in Example 10, one occurring when $t \doteq 2.8022$. This, however, reveals a shortcoming of the '`A \ b`' command. It can find a particular solution, but when multiple solutions exist, it cannot find them all.

1.7 Determinants

In a previous section we saw how to use Gaussian elimination to solve linear systems of n equations in n unknowns. It is a process one can carry out without any knowledge of whether the linear algebraic system in question *has* a solution or, if it does, whether that solution is *unique* (central questions, as we have already seen, when studying initial value problems for ODEs). So long as one knows what to look for, answers to these fundamental questions do reveal themselves during the process. In this section, however, we investigate whether something about the existence and/or uniqueness of solutions may be learned *in advance* of carrying out Gaussian elimination.