

Matrix Algebra Review

Eric Zivot
Department of Economics
University of Washington

January 3, 2000
This version: October 17, 2009

1 Matrix Algebra Review

This chapter reviews some basic matrix algebra concepts that we will use throughout the book.

1.1 Matrices and Vectors

A *matrix* is just an array of numbers. The *dimension* of a matrix is determined by the number of its rows and columns. For example, a matrix \mathbf{A} with n rows and m columns is illustrated below

$$\mathbf{A}_{(n \times m)} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \vdots & \vdots & \dots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{bmatrix}$$

where a_{ij} denotes the i^{th} row and j^{th} column element of \mathbf{A} .

A *vector* is simply a matrix with 1 column. For example,

$$\mathbf{x}_{(n \times 1)} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

is an $n \times 1$ vector with elements x_1, x_2, \dots, x_n . Vectors and matrices are often written in bold type (or underlined) to distinguish them from scalars (single elements of vectors or matrices).

Example 1 *Matrix creation in R*

In R, matrix objects are created using the `matrix()` function. For example, to create the 2×3 matrix

$$\underset{(2 \times 3)}{\mathbf{A}} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

use

```
> matA = matrix(data=c(1,2,3,4,5,6),nrow=2,ncol=3,byrow=TRUE)
> matA
      [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
> class(matA)
[1] "matrix"
```

The optional argument `byrow=TRUE` fills the matrix row by row. The default is `byrow=FALSE` which fills the matrix column by column:

```
> matrix(data=c(1,2,3,4,5,6),nrow=2,ncol=3)
      [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

Matrix objects have row and column dimension attributes which can be examined with the `dim()` function:

```
> dim(matA)
[1] 2 3
```

The rows and columns can be given names using

```
> dimnames(matA) = list(c("row1","row2"),c("col1","col2","col3"))
> matA
      col1 col2 col3
row1    1    2    3
row2    4    5    6
```

or

```
> colnames(matA) = c("Col1", "Col2", "Col3")
> rownames(matA) = c("Row1", "Row2")
> matA
      Col1 Col2 Col3
Row1    1    2    3
Row2    4    5    6
```

The elements of a matrix can be extracted or subsetted as follows:

```

> matA[1, 2]
[1] 2
> matA["Row1", "Col1"]
[1] 1
> matA[1, ]
Col1 Col2 Col3
  1    2    3
> matA[, 2]
Row1 Row2
  2    5

```

To preserve the dimension attributes when subsetting use the `drop=FALSE` option

```

> matA[1, , drop=FALSE]
  Col1 Col2 Col3
Row1  1    2    3
> matA[, 2, drop=FALSE]
  Col2
Row1  2
Row2  5

```



Example 2 *Creating vectors in R*

Vectors can be created in R using a variety of methods

```

> xvec = c(1,2,3)
> xvec
[1] 1 2 3
> xvec = 1:3
> xvec
[1] 1 2 3
> xvec = seq(from=1,to=3,by=1)
> xvec
[1] 1 2 3

```

Vectors in R are of class `numeric` and do not have a dimension attribute:

```

> class(xvec)
[1] "numeric"
> dim(xvec)
NULL

```

The elements of a vector can be assigned names using the `names()` function:

```

> names(xvec) = c("x1", "x2", "x3")
> xvec
x1 x2 x3
  1  2  3

```

To force a dimension attribute onto a vector, coerced it to a `matrix` object using `as.matrix()`:

```

> xvec = as.matrix(xvec)
> xvec
  [,1]
x1    1
x2    2
x3    3
> class(xvec)
[1] "matrix"
> dim(xvec)
[1] 3 1

```



The *transpose* of an $n \times m$ matrix \mathbf{A} is a new matrix with the rows and columns of \mathbf{A} interchanged, and is denoted \mathbf{A}' or \mathbf{A}^\top . For example,

$$\begin{aligned} \underset{(2 \times 3)}{\mathbf{A}} &= \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \quad \underset{(3 \times 2)}{\mathbf{A}'} = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \\ \underset{(3 \times 1)}{\mathbf{x}} &= \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \quad \underset{(1 \times 3)}{\mathbf{x}'} = [1 \ 2 \ 3]. \end{aligned}$$

A *symmetric* matrix \mathbf{A} is such that $\mathbf{A} = \mathbf{A}'$. Obviously, this can only occur if \mathbf{A} is a *square* matrix; i.e., the number of rows of \mathbf{A} is equal to the number of columns. For example, consider the 2×2 matrix

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}.$$

Then,

$$\mathbf{A}' = \mathbf{A} = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}.$$

Example 3 *Transpose of a matrix in R*

To take the transpose of a matrix or vector use the `t()` function

```

> matA = matrix(data=c(1,2,3,4,5,6),nrow=2,ncol=3,byrow=T)
> t(matA)
      [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
> xvec = c(1,2,3)
> t(xvec)
      [,1] [,2] [,3]
[1,]    1    2    3

```

Notice that, when applied to a vector with n elements, the `t()` function returns a `matrix` object with dimension $1 \times n$. ■

1.2 Basic Matrix Operations

In this section we review the basic matrix operations of addition, subtraction, scalar multiplication and multiplication.

1.2.1 Addition and subtraction

Matrix addition and subtraction are element by element operations and only apply to matrices of the same dimension. For example, let

$$\mathbf{A} = \begin{bmatrix} 4 & 9 \\ 2 & 1 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 2 & 0 \\ 0 & 7 \end{bmatrix}.$$

Then

$$\begin{aligned} \mathbf{A} + \mathbf{B} &= \begin{bmatrix} 4 & 9 \\ 2 & 1 \end{bmatrix} + \begin{bmatrix} 2 & 0 \\ 0 & 7 \end{bmatrix} = \begin{bmatrix} 4+2 & 9+0 \\ 2+0 & 1+7 \end{bmatrix} = \begin{bmatrix} 6 & 9 \\ 2 & 8 \end{bmatrix}, \\ \mathbf{A} - \mathbf{B} &= \begin{bmatrix} 4 & 9 \\ 2 & 1 \end{bmatrix} - \begin{bmatrix} 2 & 0 \\ 0 & 7 \end{bmatrix} = \begin{bmatrix} 4-2 & 9-0 \\ 2-0 & 1-7 \end{bmatrix} = \begin{bmatrix} 2 & 9 \\ 2 & -6 \end{bmatrix}. \end{aligned}$$

Example 4 *Matrix addition and subtraction in R*

Matrix addition and subtraction is straightforward in R:

```

> matA = matrix(c(4,9,2,1),2,2,byrow=T)
> matB = matrix(c(2,0,0,7),2,2,byrow=T)
> matA
      [,1] [,2]
[1,]    4    9
[2,]    2    1
> matB

```

```

      [,1] [,2]
[1,]    2    0
[2,]    0    7
> # matrix addition
> matC = matA + matB
> matC
      [,1] [,2]
[1,]    6    9
[2,]    2    8
> # matrix subtraction
> matC = matA - matB
> matC
      [,1] [,2]
[1,]    2    9
[2,]    2   -6

```



1.2.2 Scalar Multiplication

Here we refer to the multiplication of a matrix by a scalar number. This is also an element-by-element operation. For example, let $c = 2$ and

$$\mathbf{A} = \begin{bmatrix} 3 & -1 \\ 0 & 5 \end{bmatrix}.$$

Then

$$c \cdot \mathbf{A} = \begin{bmatrix} 2 \cdot 3 & 2 \cdot (-1) \\ 2 \cdot (0) & 2 \cdot 5 \end{bmatrix} = \begin{bmatrix} 6 & -2 \\ 0 & 10 \end{bmatrix}.$$

Example 5 *Scalar multiplication in R*

```

> matC = 2*matA
> matC
      [,1] [,2]
[1,]    8   18
[2,]    4    2

```



1.2.3 Matrix Multiplication

Matrix multiplication only applies to *conformable* matrices. \mathbf{A} and \mathbf{B} are conformable matrices if the number of columns in \mathbf{A} is equal to the number of rows in \mathbf{B} . For

example, if \mathbf{A} is $m \times n$ and \mathbf{B} is $n \times p$ then \mathbf{A} and \mathbf{B} are conformable. The mechanics of matrix multiplication is best explained by example. Let

$$\mathbf{A}_{(2 \times 2)} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \text{ and } \mathbf{B}_{(2 \times 3)} = \begin{bmatrix} 1 & 2 & 1 \\ 3 & 4 & 2 \end{bmatrix}.$$

Then

$$\begin{aligned} \mathbf{A}_{(2 \times 2)} \cdot \mathbf{B}_{(2 \times 3)} &= \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} 1 & 2 & 1 \\ 3 & 4 & 2 \end{bmatrix} \\ &= \begin{bmatrix} 1 \cdot 1 + 2 \cdot 3 & 1 \cdot 2 + 2 \cdot 4 & 1 \cdot 1 + 2 \cdot 2 \\ 3 \cdot 1 + 4 \cdot 3 & 3 \cdot 2 + 4 \cdot 4 & 3 \cdot 1 + 4 \cdot 2 \end{bmatrix} \\ &= \begin{bmatrix} 7 & 10 & 5 \\ 15 & 22 & 11 \end{bmatrix} = \mathbf{C}_{(2 \times 3)} \end{aligned}$$

The resulting matrix \mathbf{C} has 2 rows and 3 columns. In general, if \mathbf{A} is $n \times m$ and \mathbf{B} is $m \times p$ then $\mathbf{C} = \mathbf{A} \cdot \mathbf{B}$ is $n \times p$.

As another example, let

$$\mathbf{A}_{(2 \times 2)} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \text{ and } \mathbf{B}_{(2 \times 1)} = \begin{bmatrix} 2 \\ 6 \end{bmatrix}.$$

Then

$$\begin{aligned} \mathbf{A}_{(2 \times 2)} \cdot \mathbf{B}_{(2 \times 1)} &= \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 6 \end{bmatrix} \\ &= \begin{bmatrix} 1 \cdot 2 + 2 \cdot 6 \\ 3 \cdot 2 + 4 \cdot 6 \end{bmatrix} \\ &= \begin{bmatrix} 14 \\ 30 \end{bmatrix}. \end{aligned}$$

As a final example, let

$$\mathbf{x} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \mathbf{y} = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}.$$

Then

$$\mathbf{x}'\mathbf{y} = [1 \ 2 \ 3] \cdot \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} = 1 \cdot 4 + 2 \cdot 5 + 3 \cdot 6 = 32$$

Example 6 *Matrix multiplication in R*

In R, matrix multiplication is performed with the `%*%` operator. For example:

```

> matA = matrix(1:4,2,2,byrow=T)
> matB = matrix(5:8,2,2,byrow=T)
> matA
      [,1] [,2]
[1,]    1    2
[2,]    3    4
> matB
      [,1] [,2]
[1,]    5    6
[2,]    7    8
> dim(matA)
[1] 2 2
> dim(matB)
[1] 2 2
#
> matC = matA%*%matB
> matC
      [,1] [,2]
[1,]   19   22
[2,]   43   50
> # note: A%*%B is generally not equal to B%*%A
> matC = matB%*%matA
> matC
      [,1] [,2]
[1,]   23   34
[2,]   31   46

```

The next example shows matrix multiplication in R also works on numeric vectors:

```

> matA = matrix(c(1,2,3,4), 2, 2, byrow=TRUE)
> vecB = c(2,6)
> matA%*%vecB
      [,1]
[1,]   14
[2,]   30
> vecX = c(1,2,3)
> vecY = c(4,5,6)
> t(vecX)%*%vecY
      [,1]
[1,]   32

```



1.3 The Identity Matrix

The identity matrix plays a similar role as the number 1. Multiplying any number by 1 gives back that number. In matrix algebra, pre or post multiplying a matrix \mathbf{A} by a conformable identity matrix gives back the matrix \mathbf{A} . To illustrate, let

$$\mathbf{I}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

denote the 2 dimensional identity matrix and let

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

denote an arbitrary 2×2 matrix. Then

$$\begin{aligned} \mathbf{I}_2 \cdot \mathbf{A} &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \\ &= \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \mathbf{A}, \end{aligned}$$

and

$$\begin{aligned} \mathbf{A} \cdot \mathbf{I}_2 &= \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \mathbf{A}. \end{aligned}$$

Example 7 *The identity matrix in R*

Use the `diag()` function to create an identity matrix:

```
> matI = diag(2)
> matI
      [,1] [,2]
[1,]    1    0
[2,]    0    1
> matA = matrix(c(1,2,3,4), 2, 2, byrow=TRUE)
> matI%%matA
      [,1] [,2]
[1,]    1    2
[2,]    3    4
> matA%%matI
      [,1] [,2]
[1,]    1    2
[2,]    3    4
```

■

1.4 Representing Summation Using Vector Notation

Consider the sum

$$\sum_{k=1}^n x_k = x_1 + \cdots + x_n.$$

Let $\mathbf{x} = (x_1, \dots, x_n)'$ be an $n \times 1$ vector and $\mathbf{1} = (1, \dots, 1)'$ be an $n \times 1$ vector of ones. Then

$$\mathbf{x}'\mathbf{1} = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix} \cdot \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} = x_1 + \cdots + x_n = \sum_{k=1}^n x_k,$$

and

$$\mathbf{1}'\mathbf{x} = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = x_1 + \cdots + x_n = \sum_{k=1}^n x_k.$$

Next, consider the sum of squared x values

$$\sum_{k=1}^n x_k^2 = x_1^2 + \cdots + x_n^2.$$

This sum can be conveniently represented as

$$\mathbf{x}'\mathbf{x} = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = x_1^2 + \cdots + x_n^2 = \sum_{k=1}^n x_k^2.$$

Last, consider the sum of cross products

$$\sum_{k=1}^n x_k y_k = x_1 y_1 + \cdots + x_n y_n.$$

This sum can be compactly represented by

$$\mathbf{x}'\mathbf{y} = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = x_1 y_1 + \cdots + x_n y_n = \sum_{k=1}^n x_k y_k.$$

Note that $\mathbf{x}'\mathbf{y} = \mathbf{y}'\mathbf{x}$.

Example 8 *Computing sums in R*

In R, summing the elements in a vector can be done using matrix algebra.

```
# create vector of 1's and a vector x
> onevec = rep(1,3)
> onevec
[1] 1 1 1
> xvec = c(1,2,3)
> xvec
[1] 1 2 3
# sum elements in x
> t(xvec)%*%onevec
      [,1]
[1,]    6
```

The functions `crossprod()` and `sum()` are generally computationally more efficient:

```
> crossprod(xvec,onevec)
      [,1]
[1,]    6
> sum(xvec)
[1] 6
```

Sums of squares are best computed using

```
> crossprod(xvec)
      [,1]
[1,]   14
> sum(xvec^2)
[1] 14
```

The dot or cross-product of two vectors is best computed using the `crossprod()` function:

```
> yvec = 4:6
> xvec
[1] 1 2 3
> yvec
[1] 4 5 6
> crossprod(xvec,yvec)
      [,1]
[1,]   32
> crossprod(yvec,xvec)
      [,1]
[1,]   32
```



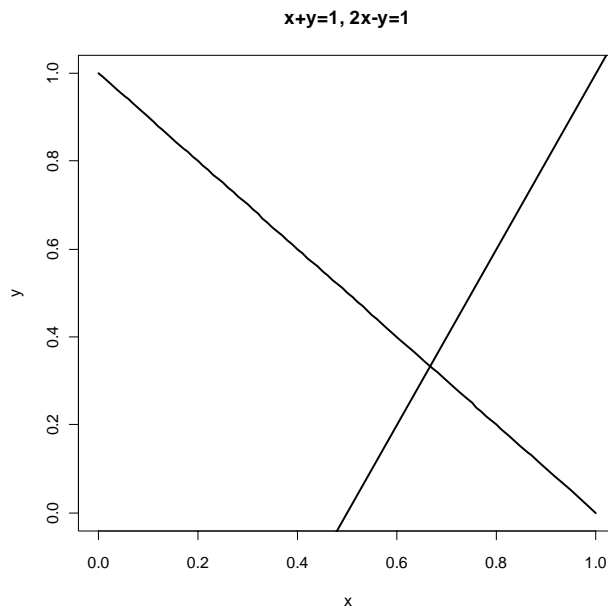


Figure 1: System of two linear equations.

1.5 Representing Systems of Linear Equations Using Matrix Algebra

Consider the system of two linear equations

$$x + y = 1 \tag{1}$$

$$2x - y = 1 \tag{2}$$

As shown in Figure , equations (1) and (2) represent two straight lines which intersect at the point $x = \frac{2}{3}$ and $y = \frac{1}{3}$. This point of intersection is determined by solving for the values of x and y such that $x + y = 2x - y$ ¹.

The two linear equations can be written in matrix form as

$$\begin{bmatrix} 1 & 1 \\ 2 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

or

$$\mathbf{A} \cdot \mathbf{z} = \mathbf{b},$$

where

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 2 & -1 \end{bmatrix}, \mathbf{z} = \begin{bmatrix} x \\ y \end{bmatrix} \text{ and } \mathbf{b} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

¹Solving for x gives $x = 2y$. Substituting this value into the equation $x + y = 1$ gives $2y + y = 1$ and solving for y gives $y = 1/3$. Solving for x then gives $x = 2/3$.

If there was a (2×2) matrix \mathbf{B} , with elements b_{ij} , such that $\mathbf{B} \cdot \mathbf{A} = \mathbf{I}_2$, where \mathbf{I}_2 is the (2×2) identity matrix, then we could *solve* for the elements in \mathbf{z} as follows. In the equation $\mathbf{A} \cdot \mathbf{z} = \mathbf{b}$, pre-multiply both sides by \mathbf{B} to give

$$\begin{aligned} \mathbf{B} \cdot \mathbf{A} \cdot \mathbf{z} &= \mathbf{B} \cdot \mathbf{b} \\ \implies \mathbf{I} \cdot \mathbf{z} &= \mathbf{B} \cdot \mathbf{b} \\ \implies \mathbf{z} &= \mathbf{B} \cdot \mathbf{b}, \end{aligned}$$

or

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} b_{11} \cdot 1 + b_{12} \cdot 1 \\ b_{21} \cdot 1 + b_{22} \cdot 1 \end{bmatrix}$$

If such a matrix \mathbf{B} exists it is called the *inverse* of \mathbf{A} and is denoted \mathbf{A}^{-1} . Intuitively, the inverse matrix \mathbf{A}^{-1} plays a similar role as the inverse of a number. Suppose a is a number; e.g., $a = 2$. Then we know that $\frac{1}{a} \cdot a = a^{-1}a = 1$. Similarly, in matrix algebra $\mathbf{A}^{-1}\mathbf{A} = \mathbf{I}_2$ where \mathbf{I}_2 is the identity matrix. Now, consider solving the equation $a \cdot x = 1$. By simple division we have that $x = \frac{1}{a}x = a^{-1}x$. Similarly, in matrix algebra if we want to solve the system of linear equations $\mathbf{A}\mathbf{x} = \mathbf{b}$ we multiply by \mathbf{A}^{-1} and get the solution $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$.

Using $\mathbf{B} = \mathbf{A}^{-1}$, we may express the solution for \mathbf{z} as

$$\mathbf{z} = \mathbf{A}^{-1}\mathbf{b}.$$

As long as we can determine the elements in \mathbf{A}^{-1} then we can solve for the values of x and y in the vector \mathbf{z} . The system of linear equations has a solution as long as the two lines intersect, so we can determine the elements in \mathbf{A}^{-1} provided the two lines are not parallel. If the two lines are parallel, then one of the equations is a multiple of the other.

There are general numerical algorithms for finding the elements of \mathbf{A}^{-1} (e.g., so-called Gaussian elimination) and matrix programming languages and spreadsheets have these algorithms available. However, if \mathbf{A} is a (2×2) matrix then there is a simple formula for \mathbf{A}^{-1} . Let \mathbf{A} be a (2×2) matrix such that

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}.$$

Then

$$\mathbf{A}^{-1} = \frac{1}{\det(\mathbf{A})} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix},$$

where $\det(\mathbf{A}) = a_{11}a_{22} - a_{21}a_{12}$ denotes the determinant of \mathbf{A} . By brute force matrix

multiplication we can verify this formula:

$$\begin{aligned}
 \mathbf{A}^{-1}\mathbf{A} &= \frac{1}{a_{11}a_{22} - a_{21}a_{12}} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \\
 &= \frac{1}{a_{11}a_{22} - a_{21}a_{12}} \begin{bmatrix} a_{22}a_{11} - a_{12}a_{21} & a_{22}a_{12} - a_{12}a_{22} \\ -a_{21}a_{11} + a_{11}a_{21} & -a_{21}a_{12} + a_{11}a_{22} \end{bmatrix} \\
 &= \frac{1}{a_{11}a_{22} - a_{21}a_{12}} \begin{bmatrix} a_{22}a_{11} - a_{12}a_{21} & 0 \\ 0 & -a_{21}a_{12} + a_{11}a_{22} \end{bmatrix} \\
 &= \begin{bmatrix} \frac{a_{22}a_{11} - a_{12}a_{21}}{a_{11}a_{22} - a_{21}a_{12}} & 0 \\ 0 & \frac{-a_{21}a_{12} + a_{11}a_{22}}{a_{11}a_{22} - a_{21}a_{12}} \end{bmatrix} \\
 &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.
 \end{aligned}$$

Let's apply the above rule to find the inverse of \mathbf{A} in our example:

$$\mathbf{A}^{-1} = \frac{1}{-1-2} \begin{bmatrix} -1 & -1 \\ -2 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{3} & \frac{1}{3} \\ \frac{2}{3} & \frac{-1}{3} \end{bmatrix}.$$

Notice that

$$\mathbf{A}^{-1}\mathbf{A} = \begin{bmatrix} \frac{1}{3} & \frac{1}{3} \\ \frac{2}{3} & \frac{-1}{3} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 2 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Our solution for \mathbf{z} is then

$$\begin{aligned}
 \mathbf{z} &= \mathbf{A}^{-1}\mathbf{b} \\
 &= \begin{bmatrix} \frac{1}{3} & \frac{1}{3} \\ \frac{2}{3} & \frac{-1}{3} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\
 &= \begin{bmatrix} \frac{2}{3} \\ \frac{1}{3} \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix}
 \end{aligned}$$

so that $x = \frac{2}{3}$ and $y = \frac{1}{3}$.

Example 9 Solving systems of linear equations in R

In R, the `solve()` function is used to compute the inverse of a matrix and solve a system of linear equations. The linear system $x + y = 1$ and $2x - y = 1$ can be represented using

```
matA = matrix(c(1,1,2,-1), 2, 2, byrow=TRUE)
vecB = c(1,1)
```

First we solve for \mathbf{A}^{-1} :

```

> matA.inv = solve(matA)
> matA.inv
      [,1] [,2]
[1,] 0.3333 0.3333
[2,] 0.6667 -0.3333
> matA.inv%%matA
      [,1] [,2]
[1,] 1 -5.551e-17
[2,] 0 1.000e+00
> matA%%matA.inv
      [,1] [,2]
[1,] 1 5.551e-17
[2,] 0 1.000e+00

```

Then we solve the system $\mathbf{z} = \mathbf{A}^{-1}\mathbf{b}$:

```

> z = matA.inv%%vecB
> z
      [,1]
[1,] 0.6667
[2,] 0.3333

```



In general, if we have n linear equations in n unknown variables we may write the system of equations as

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

which we may then express in matrix form as

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

or

$$\underset{(n \times n)}{\mathbf{A}} \cdot \underset{(n \times 1)}{\mathbf{x}} = \underset{(n \times 1)}{\mathbf{b}}.$$

The solution to the system of equations is given by

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

where $\mathbf{A}^{-1}\mathbf{A} = \mathbf{I}$ and \mathbf{I} is the $(n \times n)$ identity matrix. If the number of equations is greater than two, then we generally use numerical algorithms to find the elements in \mathbf{A}^{-1} .

2 Further Reading

Excellent treatments of portfolio theory using matrix algebra are given in Ingersol (1987), Huang and Litzenberger (1988) and Campbell, Lo and MacKinlay (1996).

References

- [1] Campbell, J.Y., Lo, A.W., and MacKinlay, A.C. (1997). *The Econometrics of Financial Markets*. Princeton, New Jersey: Princeton University Press.
- [2] Huang, C.-F., and Litzenberger, R.H. (1988). *Foundations for Financial Economics*. New York: North-Holland.
- [3] Ingersoll, J.E. (1987). *Theory of Financial Decision Making*. Totowa, New Jersey: Rowman & Littlefield.