

# An Introduction to Linear Algebra

Stephen D. Kachman

## 1 Introduction

Fundamental to the understanding of linear models is linear algebra. Without matrices, there is a tendency of viewing different designs as unrelated and we are quickly left with either a black box or sea of incomprehensible formulas. A purpose of this course is to make the black box of linear models into a gray box.

To handle the matrix manipulations we will be using `Proc IML`. `Proc IML` is a SAS procedure for manipulating matrices. With it we will be able manipulate matrices without worrying about the computational details. Ok, so we won't end up with a white box.

The objective of this handout is to develop an understanding of linear algebra and how to use `Proc IML`. Throughout this course we will be introducing additional components of linear algebra as needed.

### 1.1 Proc IML

A typical `Proc IML` program for this course will consist of three parts. First, a `Data` step to read in the data. Second, a `%include "module.sas"` to read in our general purpose modules. Third, the `Proc IML` code specific to the problem at hand. The rule for this class is

If you need it more than once, then make it a general purpose module!

A simple example is given below

```
Data simple;
input x y;
cards;
1 2
2 3
2 4
;
%include "970.sas"
Proc IML;
Load _ALL_; *Load the modules included above;
use simple;
Read All; *Use all of data set simple;
n=nrow(x);
ydev=center(n)*y;
ss=y'*ydev;
print "SS" ss;
quit; *Exit IML;
```

The file 970.sas contains

```
proc iml;
/*****
*   Center Matrix   *
*       INPUT       *
* n -- size of matrix*
*       RETURN      *
* C -- I-Jbar      *
*                   *
*****/
start center(n);
    c=i(n)-j(n,n,1/n);
return(c);
finish center;
store module=(center);
quit;
```

## 2 Matrices

A  $n \times m$  matrix is a rectangular array of numbers consisting of  $n$  rows and  $m$  columns and will be denoted by boldface capital letters (e.g.  $\mathbf{A}$ ).

$$\mathbf{A} = \begin{pmatrix} 1 & 5 \\ -.78 & 62 \end{pmatrix} \text{ and } \mathbf{B} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

In Proc IML this would be entered as  $\mathbf{A}=\{1 \ 5, -.78 \ 62 \}$ ; and  $\mathbf{b}=\{1 \ 2 \ 3 \ , \ 4 \ 5 \ 6\}$ ; . The matrix  $\mathbf{A}$  is a  $2 \times 2$  matrix and  $\mathbf{B}$  is a  $2 \times 3$  matrix.

A vector of length  $n$  is a column of  $n$  numbers (we will just be talking column vectors) and will be denoted by boldface lower case letters (e.g.  $\mathbf{a}$ ).

$$\mathbf{a} = \begin{pmatrix} 1 \\ -.78 \end{pmatrix}$$

In Proc IML this would be entered as  $\mathbf{avec}=\{1, -.78\}$ ; or possibly  $\mathbf{avec}=\mathbf{a}(:, 1)$ ; . The vector  $\mathbf{a}$  is of length 2. Note: The vector  $\mathbf{a}$  is also a  $2 \times 1$  matrix.

A scalar is a single number and will be denoted by a normal lower case letter (e.g.  $a$ ).

$$a = -.78$$

In Proc IML this would be entered as  $\mathbf{ascal}=-.78$ ; ,  $\mathbf{ascal}=\mathbf{avec}[2]$  ; ,  $\mathbf{ascal}=\mathbf{avec}[2, 1]$  ; , or  $\mathbf{ascal}=\mathbf{a}[2, 1]$  ; .

Often it is convenient to refer to parts of matrices. The individual element in row  $i$  and column  $j$  of matrix  $\mathbf{A}$  will be denoted by  $a_{ij}$ . For example  $a_{11} = 1$ ,  $a_{12} = 5$ ,  $a_{21} = -.78$ , and  $a_{22} = 62$ . Individual elements of vectors will be denoted by  $a_i$ . For example  $a_1 = 1$  and  $a_2 = -.78$ .

**Problem 2.1** In Proc IML enter the following matrices

$$\mathbf{A} = \begin{pmatrix} 12 & 23 & 77 \\ 645 & 54 & 2 \\ .01 & -3 & 8 \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} 98 \\ 76 \\ 12.03 \end{pmatrix} \quad c = 12.$$

**Problem 2.2** In Proc IML obtain  $\mathbf{a}_1$ ,  $\mathbf{a}_2$ , and  $\mathbf{a}_3$  which are the first three columns of  $\mathbf{A}$ .

## 2.1 Partitioned Matrices

We will also be building matrices out of other matrices. A partitioned matrix is a matrix consisting of a set of sub-matrices. For example,

$$D = (A \ B) = \begin{pmatrix} 1 & 5 & 1 & 2 & 3 \\ -0.78 & 62 & 4 & 5 & 6 \end{pmatrix}$$

where  $D$  is a  $2 \times 5$  matrix. In Proc IML this would be entered as `D=( A || B );`. Clearly, for this to make sense  $A$  and  $B$  must have the same number of rows. If we were to stack  $A$  on top of  $B$  then  $A$  and  $B$  must have the same number of columns. In Proc IML this would be entered as `E=(A)//(B);`.

**Problem 2.3** *In Proc IML construct the matrix*

$$E = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$$

where

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \quad B = \begin{pmatrix} 5 \\ 6 \end{pmatrix} \quad C = (7 \ 8) \quad D = 9.$$

What happens if you try to construct

$$F = \begin{pmatrix} A & C \\ B & D \end{pmatrix} ?$$

## 3 Simple Operators

### 3.1 Addition

To add matrices  $A$  and  $B$  they both must be of the same size ( $n \times m$ ). To add partitioned matrices each of the sub-matrices must be of the same size. If

$$C = A + B,$$

then  $c_{ij} = a_{ij} + b_{ij}$ . If  $A$ ,  $B$ , and  $C$  are partitioned into  $n_i \times m_j$  sub-matrices  $A_{ij}$ ,  $B_{ij}$ , and  $C_{ij}$  as follows

$$\begin{pmatrix} C_{11} & \dots & C_{1c} \\ \dots & \dots & \dots \\ C_{r1} & \dots & C_{rc} \end{pmatrix} = \begin{pmatrix} A_{11} & \dots & A_{1c} \\ \dots & \dots & \dots \\ A_{r1} & \dots & A_{rc} \end{pmatrix} + \begin{pmatrix} B_{11} & \dots & B_{1c} \\ \dots & \dots & \dots \\ B_{r1} & \dots & B_{rc} \end{pmatrix},$$

then  $C_{ij} = A_{ij} + B_{ij}$ . For example,

$$A_{11} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, A_{21} = (5 \ 6), B_{11} = \begin{pmatrix} 7 & 8 \\ 9 & 10 \end{pmatrix}, B_{21} = (11 \ 12),$$

then

$$C = \begin{pmatrix} C_{11} \\ C_{21} \end{pmatrix} = \begin{pmatrix} A_{11} + B_{11} \\ A_{21} + B_{21} \end{pmatrix} = \begin{pmatrix} 8 & 10 \\ 12 & 14 \\ 16 & 18 \end{pmatrix}.$$

This is accomplished in Proc IML as follows:

```

a11 = {1 2, 3 4};
a21 = {5 6}
a=(a11)/(a21);
b11={7 8, 9 10};
b21=(11 12);
b=(b11 || b21);
c=a+b;
print "A+B" c;

```

### 3.2 Multiplication

To multiply matrices  $\mathbf{A}$ :  $n \times m$  and  $\mathbf{B}$ :  $r \times c$  ( $\mathbf{C} = \mathbf{A}\mathbf{B}$ ) the number of columns in  $\mathbf{A}$  ( $m$ ) must be equal the number of rows in  $\mathbf{B}$  ( $r$ ). The resulting matrix  $\mathbf{C}$  is a  $n \times c$  matrix. If

$$\mathbf{C} = \mathbf{A}\mathbf{B},$$

then  $c_{ij} = \sum_{k=1}^m a_{ik} b_{kj}$ . This is accomplished in Proc IML as follows:  $\mathbf{c}=\mathbf{a}*\mathbf{b}$ ; . If  $\mathbf{A}$ ,  $\mathbf{B}$ , and  $\mathbf{C}$  are partitioned into sub-matrices  $\mathbf{A}_{ik}$ ,  $\mathbf{B}_{kj}$ , and  $\mathbf{C}_{ij}$  as follows

$$\begin{pmatrix} \mathbf{C}_{11} & \dots & \mathbf{C}_{1c} \\ \dots & \dots & \dots \\ \mathbf{C}_{n1} & \dots & \mathbf{C}_{nc} \end{pmatrix} = \begin{pmatrix} \mathbf{A}_{11} & \dots & \mathbf{A}_{1m} \\ \dots & \dots & \dots \\ \mathbf{A}_{n1} & \dots & \mathbf{A}_{nm} \end{pmatrix} \begin{pmatrix} \mathbf{B}_{11} & \dots & \mathbf{B}_{1c} \\ \dots & \dots & \dots \\ \mathbf{B}_{m1} & \dots & \mathbf{B}_{mc} \end{pmatrix},$$

then  $\mathbf{C}_{ij} = \sum_{k=1}^m \mathbf{A}_{ik} \mathbf{B}_{kj}$ . For example,

$$\mathbf{X} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}, \mathbf{Z} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \boldsymbol{\beta} = \begin{pmatrix} 10 \\ -1 \\ -2 \\ -3 \end{pmatrix}, \mathbf{u} = \begin{pmatrix} -2 \\ -1 \\ 0 \\ 1 \\ 2 \end{pmatrix},$$

then

$$(\mathbf{X} \quad \mathbf{Z}) \begin{pmatrix} \boldsymbol{\beta} \\ \mathbf{u} \end{pmatrix} = (\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{u}) = \begin{pmatrix} \overbrace{10-1}^{\mathbf{X}\boldsymbol{\beta}} & \overbrace{-2}^{\mathbf{Z}\mathbf{u}} & = 7 \\ 10-1 & -1 & = 8 \\ 10-2 & -0 & = 8 \\ 10-2 & -0 & = 8 \\ 10-2 & +1 & = 9 \\ 10-3 & +2 & = 9 \\ 10-3 & +2 & = 9 \\ 10-3 & +2 & = 9 \end{pmatrix}$$

This is accomplished in Proc IML as follows:  $(\mathbf{X}|\mathbf{Z})*(\mathbf{beta}/\mathbf{u})$ ; , assuming that  $\mathbf{X}$ ,  $\mathbf{Z}$ ,  $\mathbf{beta}$ , and  $\mathbf{u}$  have already been created.

### 3.3 Transpose

The transpose of a matrix flips a matrix over. The transpose of a  $n \times m$  matrix  $\mathbf{A}$  ( $\mathbf{B} = \mathbf{A}'$  or  $\mathbf{B} = \mathbf{A}^T$ ) is an  $m \times n$  matrix whose element in row  $i$  and column  $j$  is  $b_{ij} = a_{ji}$ . For example:

$$\mathbf{A} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \text{ and } \mathbf{A}' = \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

In Proc IML the transpose of  $\mathbf{A}$  is obtained as follows:  $\mathbf{a}'$ .<sup>1</sup> If  $\mathbf{A}$  and  $\mathbf{B}$  are partitioned into sub-matrices  $\mathbf{A}_{ji}$  and  $\mathbf{B}_{ij}$  as follows

$$\begin{pmatrix} \mathbf{B}_{11} & \dots & \mathbf{B}_{1n} \\ \dots & \dots & \dots \\ \mathbf{B}_{m1} & \dots & \mathbf{B}_{mn} \end{pmatrix} = \begin{pmatrix} \mathbf{A}_{11} & \dots & \mathbf{A}_{1m} \\ \dots & \dots & \dots \\ \mathbf{A}_{n1} & \dots & \mathbf{A}_{nm} \end{pmatrix}' = \begin{pmatrix} \mathbf{A}'_{11} & \dots & \mathbf{A}'_{n1} \\ \dots & \dots & \dots \\ \mathbf{A}'_{1m} & \dots & \mathbf{A}'_{nm} \end{pmatrix}.$$

**Problem 3.1** Using  $\mathbf{X}$ ,  $\mathbf{Z}$ ,  $\beta$ , and  $\mathbf{u}$  on page 4 and

$$\mathbf{y} = (8.4 \ 11.4 \ 7.0 \ 8.3 \ 7.0 \ 9.6 \ 8.2 \ 6.4)'$$

calculate

$$(\mathbf{X} \ \mathbf{Z})'(\mathbf{X} \ \mathbf{Z}), \mathbf{C} = \begin{pmatrix} \mathbf{X}'\mathbf{X} & \mathbf{X}'\mathbf{Z} \\ \mathbf{Z}'\mathbf{X} & \mathbf{Z}'\mathbf{Z} + \mathbf{I}.4 \end{pmatrix} \text{ and } (\mathbf{X} \ \mathbf{Z})' \mathbf{y}.$$

The matrix  $\mathbf{I}$  is an identity matrix. An identity matrix is a  $n \times n$  matrix with ones along the diagonal and zeros every where else. The  $5 \times 5$  identity matrix in this problem is

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

and can be obtained by the following Proc IML command `i(5)`.

### 3.4 Trace

The trace of a  $n \times n$  matrix is the sum of its diagonal elements. A matrix must be a square matrix (the number of columns is the same as the number of rows) for the trace operator to be defined. The trace of  $\mathbf{A}$  denoted by  $\text{tr}(\mathbf{A})$  is equal to  $\sum_{i=1}^n a_{ii}$ . If  $\mathbf{A}$  is partitioned into  $n_i \times n_j$  sub-matrices then the trace of  $\mathbf{A}$  is equal to  $\sum_{i=1}^r \text{tr}(\mathbf{A}_{ii}) = \sum_{i=1}^r \sum_{j=1}^{n_i} a_{ij}$ . In Proc IML the trace of matrix is obtained as `trace(a)`.

**Problem 3.2** Find the trace of  $\mathbf{C}$  in problem 3.1.

## 4 Systems of Equations

Often we will come across the situation  $\mathbf{A}\mathbf{b} = \mathbf{c}$  where  $\mathbf{A}$  and  $\mathbf{c}$  are known and we need to find  $\mathbf{b}$ . If  $\mathbf{A}$  is non-singular then this can be written as  $\mathbf{b} = \mathbf{A}^{-1}\mathbf{c}$  where  $\mathbf{A}^{-1}$  is the inverse of  $\mathbf{A}$ . When  $\mathbf{A}$  is non-singular then solution for  $\mathbf{b}$  is unique. However, it is often the case that  $\mathbf{A}$  is singular. When  $\mathbf{A}$  is singular then there may not be a solution for  $\mathbf{b}$  and if there is it will not be unique. Assuming that a solution does exist, then  $\mathbf{b} = \mathbf{A}^- \mathbf{c}$  where  $\mathbf{A}^-$  is a generalized inverse for  $\mathbf{A}$  and  $\mathbf{b}$  is a solution for the system of equations.

<sup>1</sup>Proc IML uses the left single quote ' for transpose and the right single quote ' for a character string.

## 4.1 $\mathbf{A}$ non-singular

The inverse of a  $n \times n$  matrix  $\mathbf{A}$  ( $\mathbf{A}^{-1}$ ) is the solution to the following set of equations

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

where  $\mathbf{I}$  is the  $n \times n$  identity matrix in problem 3.1. Notice that  $\mathbf{A}$  is a square matrix. The identity matrix is also useful in that for any  $n \times m$  matrix  $\mathbf{B}$ ,

$$\mathbf{I} \mathbf{B} = \mathbf{B}.$$

So that for non-singular  $\mathbf{A}$  we can solve for  $\mathbf{b}$  above as follows

$$\begin{aligned} \mathbf{A} \mathbf{b} &= \mathbf{c} \\ \mathbf{A}^{-1} \mathbf{A} \mathbf{b} &= \mathbf{A}^{-1} \mathbf{c} \\ \mathbf{I} \mathbf{b} &= \mathbf{A}^{-1} \mathbf{c} \\ \mathbf{b} &= \mathbf{A}^{-1} \mathbf{c} \end{aligned}$$

When  $\mathbf{A}$  is non-singular we can obtain the inverse of  $\mathbf{A}$  in Proc IML using the following command `inv(A)`. If we are interested only with solving a set of equations, then the command `b=solve(a,c)`; is better than `b=inv(a)*c`;

## 4.2 $\mathbf{A}$ singular

A matrix  $\mathbf{A}$  is said to be singular if the inverse of  $\mathbf{A}$  does not exist. This implies that at least one of the rows or columns is a linear function of the remaining rows or columns. For example,

$$\mathbf{A} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} = (\mathbf{a}_1 \quad \mathbf{a}_2 \quad \mathbf{a}_3)$$

is singular ( $\mathbf{a}_3 = 2 \mathbf{a}_2 - \mathbf{a}_1$ ). After removing the third column, the remaining two columns are linearly independent. Therefore, we say that  $\mathbf{A}$  has rank 2. The rank of a matrix can be obtained in Proc IML using the following command `trace(a*sweep(a'*a)*a')`.

If the rank of a matrix is equal to the number of columns then the matrix is said to be of full column rank.

Any matrix that is not square must be singular.

### 4.2.1 Generalized inverse

A generalized inverse of a  $n \times m$  matrix  $\mathbf{A}$  ( $\mathbf{A}^-$ ) is any matrix  $\mathbf{A}^-$ :  $m \times n$  that satisfies the following relationship

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

Generalized matrices exist for any matrix. Unless  $\mathbf{A}$  is non-singular, generalized inverses are not unique. If  $\mathbf{A}$  is non-singular, then the  $\mathbf{A}^- = \mathbf{A}^{-1}$  and is unique. A generalized inverse of a symmetric matrix can be obtained in Proc IML with the following command `sweep(a)`.

A generalized inverse for  $\mathbf{A}$  given above is

$$\mathbf{A}^- = \frac{1}{3} \begin{pmatrix} -5 & 2 & 0 \\ 4 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

A system of equations  $\mathbf{A} \mathbf{b} = \mathbf{c}$  is said to be consistent, if there exist at least one  $\mathbf{b}$  that satisfies the system of equations. If  $\mathbf{A} \mathbf{b} = \mathbf{c}$  is consistent, then  $\mathbf{b} = \mathbf{A}^- \mathbf{c}$  is a solution.

**Problem 4.1** Verify that  $\mathbf{A}^-$  is a generalized inverse for  $\mathbf{A}$ . Using `ginv`, find another generalized inverse for  $\mathbf{A}$ . Verify that it is a generalized inverse.

**Problem 4.2** Solve  $\mathbf{A} \mathbf{b} = \mathbf{c}$  for  $\mathbf{c}$  equal to  $(3 \ 9 \ 15)'$ ,  $(3 \ 6 \ 9)'$ , and  $(3 \ 8 \ 12)'$ . Using both generalized inverses. Verify that the  $\mathbf{b}$ s are solutions. What does it mean if the  $\mathbf{b}$ s you obtained are not solutions?

**Problem 4.3** Using in  $\mathbf{X}$ ,  $\mathbf{Z}$ ,  $\beta$ ,  $\mathbf{u}$ , and  $\mathbf{y}$  problem 3.1 find  $\hat{\beta}$  and  $\hat{\mathbf{u}}$  such that

$$\begin{pmatrix} \mathbf{X} & \mathbf{Z} \end{pmatrix} \begin{pmatrix} \hat{\beta} \\ \hat{\mathbf{u}} \end{pmatrix} = \mathbf{y}$$

**Problem 4.4** Using in  $\mathbf{X}$ ,  $\mathbf{Z}$ ,  $\beta$ ,  $\mathbf{u}$ , and  $\mathbf{y}$  problem 3.1 find  $\hat{\beta}$  and  $\hat{\mathbf{u}}$  such that

$$\begin{pmatrix} \mathbf{X}'\mathbf{X} & \mathbf{X}'\mathbf{Z} \\ \mathbf{Z}'\mathbf{X} & \mathbf{Z}'\mathbf{Z} + \mathbf{I} \end{pmatrix} \begin{pmatrix} \hat{\beta} \\ \hat{\mathbf{u}} \end{pmatrix} = \begin{pmatrix} \mathbf{X}'\mathbf{y} \\ \mathbf{Z}'\mathbf{y} \end{pmatrix}$$