

Fall 2003 Math 308/501–502
9 Matrix Methods for Linear Systems
9.3 Matrices and Vectors
 Mon, 10/Nov ©2003, Art Belmonte

Summary

(Read Chapter 11 of your lab manual for additional material.)

Types of matrices

An $m \times n$ **matrix** is a rectangular collection of elements arranged in m rows and n columns. These elements are real or complex numbers or symbolic expressions. We'll denote matrices with uppercase boldface roman letters; e.g., **A**, **B**, etc. Thus $\mathbf{A} = [a_{ij}]$ is a matrix whose ij -th entry is a_{ij} . A **row vector** is a $1 \times n$ matrix, whereas a **column vector** is an $n \times 1$ matrix. Vectors will be signified by lowercase boldface roman letters; e.g., **v**, **x**, etc. A matrix is square if $m = n$ and diagonal if it is square with all off-diagonal entries being zeros. The elements of a **zero matrix** or a **zero vector** are all zeros. Either is denoted by **0**.

The Algebra of matrices

Matrix Addition and Scalar Multiplication Provided that matrices **A** and **B** have the same dimensions, their **sum** is given by $\mathbf{A} + \mathbf{B} = [a_{ij}] + [b_{ij}] = [a_{ij} + b_{ij}]$. **Scalar multiplication** is defined by $r\mathbf{A} = r[a_{ij}] = [ra_{ij}]$. A **scalar** is a real or complex number or a symbolic expression. The expression $-\mathbf{A}$ stands for $(-1)\mathbf{A}$ and matrix subtraction is defined by $\mathbf{A} - \mathbf{B} = \mathbf{A} + (-1)\mathbf{B}$. Here are some properties of matrix addition and scalar multiplication.

$$\begin{aligned} \mathbf{A} + (\mathbf{B} + \mathbf{C}) &= (\mathbf{A} + \mathbf{B}) + \mathbf{C} \\ \mathbf{A} + \mathbf{B} &= \mathbf{B} + \mathbf{A} \\ \mathbf{A} + \mathbf{0} &= \mathbf{A} \\ \mathbf{A} + (-\mathbf{A}) &= \mathbf{0} \\ r(\mathbf{A} + \mathbf{B}) &= r\mathbf{A} + r\mathbf{B} \\ (r + s)\mathbf{A} &= r\mathbf{A} + s\mathbf{A} \\ r(s\mathbf{A}) &= (rs)\mathbf{A} = s(r\mathbf{A}) \end{aligned}$$

Matrix multiplication Let $\mathbf{A} = [a_{ik}]$ be an $m \times n$ matrix and $\mathbf{B} = [b_{kj}]$ an $n \times p$ matrix. The **matrix product** is defined by

$$\mathbf{C} = \mathbf{AB} = \mathbf{A} * \mathbf{B} = [c_{ij}], \text{ where } c_{ij} = \sum_{k=1}^n a_{ik}b_{kj}. \text{ In other}$$

words, c_{ij} is the dot product of the i th row of **A** with the j th column of **B**. (NOTE: This is different that the **array product** in MATLAB, defined by $\mathbf{P} * \mathbf{Q} = [p_{ij}] * [q_{ij}] = [p_{ij}q_{ij}]$, for matrices **P** and **Q** having the same dimensions.) Here are some properties of matrix multiplication. They are respectively known as associativity, left distributivity, right distributivity, and another instance of associativity.

$$(\mathbf{AB})\mathbf{C} = \mathbf{A}(\mathbf{BC})$$

$$\mathbf{A}(\mathbf{B} + \mathbf{C}) = \mathbf{AB} + \mathbf{AC}$$

$$(\mathbf{A} + \mathbf{B})\mathbf{C} = \mathbf{AC} + \mathbf{BC}$$

$$(r\mathbf{A})\mathbf{B} = \mathbf{A}(r\mathbf{B})$$

NOTE what is missing: commutativity! In general, $\mathbf{AB} \neq \mathbf{BA}$, even if both these products are defined. (Indeed, one or both of them may *not* be defined.)

Matrices as Linear Operators A consequence of the aforementioned properties of matrix multiplication is that an $m \times n$ matrix **A** defines a *linear* operator from \mathbb{R}^n to \mathbb{R}^m (or else \mathbb{C}^n to \mathbb{C}^m): $\mathbf{A}(\mathbf{x} + \mathbf{y}) = \mathbf{Ax} + \mathbf{Ay}$ and $\mathbf{A}(r\mathbf{x}) = r\mathbf{Ax}$. Furthermore, **AB** constitutes a composition of the linear operators **A** and **B** and is itself a linear operator; i.e., $\mathbf{AB} = \mathbf{A} \circ \mathbf{B}$. Geometric examples of linear operations that act on vectors are stretching, contracting, rotation, and reflection.

Matrix Formulation of Linear Algebraic Systems (This was the content of Section 9.2, *q.v.*)

Matrix Identity The identity in matrix multiplication is the $n \times n$ **identity matrix I** (or \mathbf{I}_n). This is a diagonal matrix all of whose diagonal entries are 1.

Matrix Inverse If **A** and **B** are square $n \times n$ matrices such that $\mathbf{AB} = \mathbf{I} = \mathbf{I}_n$, then **B** is an **inverse** of **A**. Indeed, when this occurs, it so happens that we also have $\mathbf{BA} = \mathbf{I} = \mathbf{I}_n$, whence $\mathbf{B} = \mathbf{A}^{-1}$ is well and truly said to be *the* inverse of **A**. A matrix that has no inverse is said to be **singular**. So invertible matrices are **nonsingular**. One way to find the inverse of a matrix **A** (if it exists) is to transform the augmented matrix $\mathbf{M} = [\mathbf{A}|\mathbf{I}]$ into reduced row echelon form. This results in $[\mathbf{I}|\mathbf{A}^{-1}]$, from which we may read off the inverse of **A** in the right half of the resulting augmented matrix. Of course, the easy way to compute the inverse of **A** is to use the MATLAB command `inv(A)`.

Determinants The determinant of a square $n \times n$ matrix $\mathbf{A} = [a_{ij}]$ is $\sum_{\sigma} (-1)^{\sigma} \prod_{k=1}^n a_{k\sigma_k}$, where the sum is over all permutations of the first n integers. In particular, for a 2×2 matrix we have $\det \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = a_{11}a_{22} - a_{21}a_{12}$. In general, just use `det(A)` in MATLAB to render the needful.

Theorem on Matrices and Linear Systems of Equations

Let **A** be an $n \times n$ matrix. Then the following are equivalent.

1. **A** is singular; i.e., **A** does not have an inverse.
2. $\det \mathbf{A} = 0$.
3. The system $\mathbf{Ax} = \mathbf{0}$ has nontrivial solutions; i.e., it has solutions $\mathbf{x} \neq \mathbf{0}$.
4. The columns of **A** form a linearly dependent set.
5. The rows of **A** form a linearly dependent set.

(An Equivalent Theorem)

1. \mathbf{A} is nonsingular; i.e., \mathbf{A} has an inverse.
2. $\det \mathbf{A} \neq 0$.
3. The system $\mathbf{Ax} = \mathbf{0}$ has only the trivial solution; i.e., the solution $\mathbf{x} = \mathbf{0}$.
4. The columns of \mathbf{A} form a linearly independent set.
5. The rows of \mathbf{A} form a linearly independent set.

A consequence of this theorem is that if \mathbf{A} is nonsingular, then $\mathbf{Ax} = \mathbf{b}$ has the unique solution $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b} = \mathbf{A}\backslash\mathbf{b}$.

Calculus of Matrices Let \mathbf{A} be a matrix whose elements are symbolic expressions in t . Then \mathbf{A} is a **matrix function** of t . Similarly, a vector \mathbf{x} so constituted is a **vector function** of t . Moreover, a matrix or vector function is **continuous**, **differentiable**, or **integrable** if and only if each of its elements are. Here are some differentiation formulas for matrices (or vectors).

$$\begin{aligned} \frac{d}{dt}(\mathbf{CA}) &= \mathbf{C} \frac{d\mathbf{A}}{dt} \\ \frac{d}{dt}(\mathbf{A} + \mathbf{B}) &= \frac{d\mathbf{A}}{dt} + \frac{d\mathbf{B}}{dt} \\ \frac{d}{dt}(\mathbf{AB}) &= \mathbf{A} \frac{d\mathbf{B}}{dt} + \frac{d\mathbf{A}}{dt} \mathbf{B} \end{aligned}$$

MATLAB Examples

Here are most even exercises from Section 9.3 worked via MATLAB. Rather than state each problem separately, we'll dispense with statements altogether and just show you one long concatenation of diary files!

I'll briefly talk you through these either in lecture or during a supplemental MATLAB session. We'll work several together using MATLAB.

My point is that this stuff is quite intuitive. Moreover, this is precisely what MATLAB was designed to facilitate!

```
%
%
% NSS4-521/2
%
A = [2 0 5; 2 1 1]
A =
     2     0     5
     2     1     1
B = [1 -1 2; 0 3 -2]
B =
     1    -1     2
     0     3    -2
ApB = A + B
ApB =
     3    -1     7
     2     4    -1
linear_combo = 7*A - 4*B
linear_combo =
    10     4    27
    14    -5    15
%
```

```
echo off; diary off
%
% NSS4-522/4
%
A = [2 1; 0 4; -1 3]
A =
     2     1
     0     4
    -1     3
B = [1 1 -1; 0 3 1]
B =
     1     1    -1
     0     3     1
% NOTE: This is *MATRIX* multiplication.
AB = A*B
AB =
     2     5    -1
     0    12     4
    -1     8     4
BA = B*A
BA =
     3     2
    -1    15
%
echo off; diary off
%
% NSS4-522/6
%
A = [1 2; 1 1]
A =
     1     2
     1     1
B = [0 3; 1 2]
B =
     0     3
     1     2
C = [1 -4; 1 1]
C =
     1    -4
     1     1
AB = A*B
AB =
     2     7
     1     5
ABC = AB*C
ABC =
     9    -1
     6     1
ApBxC = (A+B) * C
ApBxC =
     6     1
     5    -5
%
echo off; diary off
%
% NSS4-522/8
%
A = [2 -1; -3 4]
A =
     2    -1
    -3     4
B = [1 2; 3 2]
B =
     1     2
     3     2
AB = A*B
AB =
    -1     2
     9     2
BA = B*A
BA =
    -4     7
     0     5
% We see that MATRIX multiplication is NOT commutative!
%
echo off; diary off
%
% NSS4-522/10
```

```

%
A = [4 1; 5 9]
A =
     4     1
     5     9
% The default decimal format is SHORT
Ainv = inv(A)
Ainv =
     0.2903    -0.0323
    -0.1613     0.1290
% We'll often/mostly use RATIONAL format in Chapter 9.
format rat
Ainv =
     9/31    -1/31
    -5/31     4/31
% Revert to default format
format short
% The product of A with its inverse (or vice versa)
% is the identity matrix
check1 = A * Ainv
check1 =
     1.0000    -0.0000
         0     1.0000
check2 = Ainv * A
check2 =
     1.0000    -0.0000
         0     1.0000
%
echo off; diary off
:::::::::::::
n522x12.txt
:::::::::::::
%
% NSS4-522/12
%
A = [1 1 1; 1 2 3; 0 1 1]
A =
     1     1     1
     1     2     3
     0     1     1
Ainv = inv(A)
Ainv =
     1     0    -1
     1    -1     2
    -1     1    -1
% Integer entries; how quaint!
%
% The product of A with its inverse (or vice versa)
% is the identity matrix
check1 = A * Ainv
check1 =
     1     0     0
     0     1     0
     0     0     1
check2 = Ainv * A
check2 =
     1     0     0
     0     1     0
     0     0     1
%
echo off; diary off
:::::::::::::
n522x14.txt
:::::::::::::
%
% NSS4-522/14
%
A = [1 1 1; 1 -1 2; 1 1 4]
A =
     1     1     1
     1    -1     2
     1     1     4
Ainv = inv(A)
Ainv =
     1.0000     0.5000    -0.5000
     0.3333    -0.5000     0.1667
    -0.3333         0     0.3333
format rat
Ainv =
     1         1/2    -1/2
    1/3        -1/2     1/6
    -1/3         0     1/3
format short
%
% The product of A with its inverse (or vice versa)
% is the identity matrix
check1 = A * Ainv
check1 =

```

```

     1.0000         0    -0.0000
         0     1.0000         0
     0.0000         0     1.0000
check2 = Ainv * A
check2 =
     1.0000         0         0
     0.0000     1.0000         0
         0         0     1.0000
%
echo off; diary off
:::::::::::::
n522x16.txt
:::::::::::::
%
% NSS4-522/16
%
% Part (a)
A = [2 -1 1; -1 2 1; 1 1 2]
A =
     2    -1     1
    -1     2     1
     1     1     2
% This matrix is singular. That is, it does NOT
% have an inverse. Note that its determinant is 0.
Ainv = inv(A)
Warning: Matrix is singular to working precision.
(Type "warning off MATLAB:singularMatrix"
to suppress this warning.)
> In /u/belmonte/2003c/308/T/c9/s3/n522x16/n522x16.m
at line 10
Ainv =
     Inf     Inf     Inf
     Inf     Inf     Inf
     Inf     Inf     Inf
detA = det(A)
detA =
     0
% Part (b)
b = [3; 1; 3]
b =
     3
     1
     3
M = [A b]
M =
     2    -1     1     3
    -1     2     1     1
     1     1     2     3
% This system has no solutions lest 0 = 1!
MR = rref(M)
MR =
     1     0     1     0
     0     1     1     0
     0     0     0     1
% Part (c)
b = [3; 0; 3]
b =
     3
     0
     3
M = [A b]
M =
     2    -1     1     3
    -1     2     1     0
     1     1     2     3
% This system has infinitely many solutions...
MR = rref(M)
MR =
     1     0     1     2
     0     1     1     1
     0     0     0     0
syms s
% ...having the following form.
x = [2-s; 1-s; s]

x =

[ 2-s]
[ 1-s]
[   s]

%
echo off; diary off
:::::::::::::
n522x18.txt
:::::::::::::
%
% NSS4-522/18
%
syms t

```

```

v = [sin(2*t), cos(2*t)];
% The following is a SYMBOLIC matrix.
A = wron(v,t)

A =

[ sin(2*t), cos(2*t)]
[ 2*cos(2*t), -2*sin(2*t)]

% Symbolic inverse? You bet!
Ainv = simple(inv(A))

Ainv =

[ sin(2*t), 1/2*cos(2*t)]
[ cos(2*t), -1/2*sin(2*t)]

%
check1 = simple(A * Ainv)

check1 =

[ 1, 0]
[ 0, 1]

check2 = simple(Ainv * A)

check2 =

[ 1, 0]
[ 0, 1]

%
echo off; diary off
::::::::::::::::::
n523x20.txt
::::::::::::::::::
%
% NSS4-523/20
%
syms t
v = [exp(3*t), 1, t];
% The following is a SYMBOLIC matrix.
A = wron(v,t)

A =

[ exp(3*t), 1, t]
[ 3*exp(3*t), 0, 1]
[ 9*exp(3*t), 0, 0]

% Symbolic inverse? You bet!
Ainv = simple(inv(A))

Ainv =

[ 0, 0, 1/9/exp(3*t)]
[ 1, -t, -1/9+1/3*t]
[ 0, 1, -1/3]

%
check1 = simple(A * Ainv)

check1 =

[ 1, 0, 0]
[ 0, 1, 0]
[ 0, 0, 1]

check2 = simple(Ainv * A)

check2 =

[ 1, 0, 0]
[ 0, 1, 0]
[ 0, 0, 1]

%
echo off; diary off
::::::::::::::::::
n523x22.txt
::::::::::::::::::
%
% NSS4-523/22
%
A = [12 8; 3 2]
A =
12 8
3 2
detA = det(A)

```

```

detA =
0
%
echo off; diary off
::::::::::::::::::
n523x24.txt
::::::::::::::::::
%
% NSS4-523/24
%
A = [1 0 2; 0 3 -1; -1 2 1]
A =
1 0 2
0 3 -1
-1 2 1
detA = det(A)
detA =
11
%
echo off; diary off
::::::::::::::::::
n523x26.txt
::::::::::::::::::
%
% NSS4-523/26
%
A = [1 4 4; 3 0 -3; 1 6 2]
A =
1 4 4
3 0 -3
1 6 2
detA = det(A)
detA =
54
%
echo off; diary off
::::::::::::::::::
n523x28.txt
::::::::::::::::::
%
% NSS4-523/28
%
syms r
A = [3 3; 2 4]
A =
3 3
2 4
I = eye(2)
I =
1 0
0 1
AmrI = A - r*I

AmrI =

[ 3-r, 3]
[ 2, 4-r]

det_AmrI = det(AmrI)

det_AmrI =

6-7*r+r^2

r = solve(det_AmrI)

r =

[ 6]
[ 1]

%
echo off; diary off
::::::::::::::::::
n523x32.txt
::::::::::::::::::
%
% NSS4-523/32
%
syms t
x = exp(-t)*sin(3*t) * [1; 0; 1]

x =

[ exp(-t)*sin(3*t)]
[ 0]
[ exp(-t)*sin(3*t)]

dx_dt = diff(x,t);
pretty(dx_dt)

```

