Selected Solutions for Exercises in

Numerical Methods with MATLAB: Implementations and Applications

Gerald W. Recktenwald

Chapter 7

A Review of Linear Algebra

7.3 Manually compute $C = AB$ for

$$A = \begin{bmatrix} 1 & 1 \\ 2 & 3 \end{bmatrix} \qquad B = \begin{bmatrix} 3 & -1 \\ -2 & 1 \end{bmatrix}$$

What is the relationship between A and B?.

Solution: Using Algorithm 7.6 to compute the elements of AB as an inner product between rows of A and columns of B gives

$$AB = \begin{bmatrix} 1 & 1 \\ 2 & 3 \end{bmatrix} \begin{bmatrix} 3 & -1 \\ -2 & 1 \end{bmatrix} = \begin{bmatrix} (1)(3)+(1)(-2) & (1)(-1)+(1)(1) \\ (2)(3)+(3)(-2) & (2)(-1)+(3)(1) \end{bmatrix}$$

$$= \begin{bmatrix} (3-2) & (-1+1) \\ (6-6) & (-2+3) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Since, $AB = I$ then $B = A^{-1}$. Or, if you prefer, $A = B^{-1}$.

Alternatively, we can use Algorithm 7.5 to produce the same result. When we manually apply Algorithm 7.5 we merely proceed down the rows of the result matrix. To make the organization of Algorithm 7.5 obvious, we can separately compute each column of the result matrix. Though this helps to visualize the algorithm, it tends to make for hand calculations that take up more space. Nonetheless, we show how it can be done. The first column of AB is

$$AB_{(:,1)} = \begin{bmatrix} 1 & 1 \\ 2 & 3 \end{bmatrix} \begin{bmatrix} 3 \\ -2 \end{bmatrix} = \begin{bmatrix} (1)(3)+(1)(-2) \\ (2)(3)+(3)(-2) \end{bmatrix} = \begin{bmatrix} 3-2 \\ 6-6 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

The second column of AB is

$$AB_{(:,2)} = \begin{bmatrix} 1 & 1 \\ 2 & 3 \end{bmatrix} \begin{bmatrix} 3 & -1 \\ -2 & 1 \end{bmatrix} = \begin{bmatrix} (1)(-1)+(1)(1) \\ (2)(-1)+(3)(1) \end{bmatrix} = \begin{bmatrix} -1+1 \\ -2+3 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Reassembling the columns of AB gives $AB = I$, as before.

7.11 The *trace* of square matrix A is usually denoted $\mathrm{tr}(A)$, and is defined as the sum of the diagonal elements of A.

$$\mathrm{tr}(A) = \sum_{i=1}^{n} a_{ii}$$

Write a one-line MATLAB expression to compute the trace of a matrix, A. *Do not* use any `for...end` loops in your expression.

Solution: `trA = sum(diag(A))`

7.14 Write an m-file function (say, `matVecCol`) that evaluates a matrix-vector product (vector on the right) using the column view, Algorithm 7.1, and only one `for...end` loop. This is achieved by replacing the inner loop of Algorithm 7.1 with a single line of MATLAB code that uses colon notation. Test your function by comparing the output with `A*x` for random, but compatible, `A` and `x`.

Typographical Error: In Algorithm 7.1, the statement "initialize $b =$ `zeros`$(n, 1)$" should be "initialize $b =$ `zeros`$(m, 1)$"

Partial Solution: The `matVecCol` function is listed below. Note that the initialization statement `y = zeros(m,1);` is required since `y` must already be defined to evaluate the expression `y = x(j)*A(:,j) + y` when `j=1`. Testing of `matVecCol` is left for the reader.

```
function y = matVecCol(A,x)
% matVecCol  Matrix-vector product as a linear combination of columns of A
%
% Synopsis:  y = matVecCol(A,x)
%
% Input:     A,x = compatible matrix and column vector
%
%            y = A*x

[m,n] = size(A);    [mx,nx] = size(x);
if mx~=n
  error(sprintf('A and x are incompatible: A is %d by %d  and x is %d by %d\n',...
        size(A),size(x)));
end

y = zeros(m,1);   %  initialization required
for j=1:n
  y = x(j)*A(:,j) + y;
end
```

7.21 Write a `rowScale` function that uses a single `for...end` loop to perform the diagonal scaling of a matrix. The function definition should be

```
function B = rowScale(A,d)
```

where `A` is the matrix to be scaled, `d` is a vector of scale factors. (See Example 7.6, page 330.) The single loop is possible with judicious use of colon wild cards. Before performing any calculations test to make sure the dimensions of `A` and `d` are compatible.

Partial Solution: The `rowScale` function is listed below. Testing is left to the reader.

```
function B = rowScale(A,d)
% rowScale  Scale the rows of a matrix with the elements of a vector
%
% Synopsis:  B = rowScale(A,d)
%
% Input:     A = m-by-n matrix
%            d = m-by-1 (column) vector
%
% Output:  B = m-by-n matrix obtained by multiplying all elements in row i
%              of matrix A by d(i).  In other words, B = diag(d)*A

[m,n] = size(A);
if length(d)~=m, error('d and A not compatible');  end

% --- loop over rows of A
for i=1:m
  B(i,:) = d(i)*A(i,:);    %  scalar times a row vector
end
```

7.25 (1) Define a matrix with columns given by the vectors in equation (7.18). Use the `rank` function to determine the numbers of linearly independent columns vectors in the matrix.

Partial Solution: Since $\text{rank}(A) = 3$, A has three linearly independent columns. A is said to be *full rank*.