Math.375
IV- Matrices

Vageli Coutsias

Matrix displays

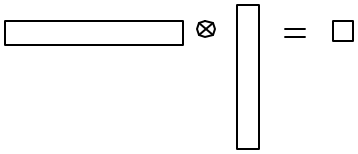
- spy(A)
- image(A)
- mesh(A)
- pltmat(A,'name',colormap,font)
- load gatlin, image(X);colormap(map),…
- bar3(A)
- >>demo: graphs & matrices, intro
- whos shows workspace

```
%-------------------------------------------------
% Block Matrices
%-------------------------------------------------
function A = MakeBlock(A_scalar,p)
% A = MakeBlock(A_scalar,p)
% A_scalar is an nXn matrix and p divides n.
% A is an (n/p)X(n/p)
% cell array that represents A_scalar
% as a block matrix with pXp blocks.
[n,n] = size(A_scalar);m = n/p;A = cell(m,m);
for i = 1:m
  for j = 1:m
  A{i,j} = A_scalar(1+(i-1)*p:i*p,1+(j-1)*p:j*p);
  end
end
%-------------------------------------------------
```

```
%-----------------------------------------------
% Banded Matrices
%-----------------------------------------------
% script tridiag
A=[ 0  1  2  3;
   -1  0  1  2;
   -2 -1  0  1;..
   -3 -2 -1  0;
   -4 -3 -2 -1];
v=diag(A,-1)
C=diag(v,3)
B1 = tril(A,1)
B2 = triu(A,-1)
T1 =-triu(tril(ones(6,6),1),-1)+3*eye(6,6)
T2 =-diag(ones(5,1),-1)+diag(ones(6,1),0)…
    -diag(ones(5,1),1)
T3 = toeplitz([2;-1;zeros(4,1)])
%-----------------------------------------------
```

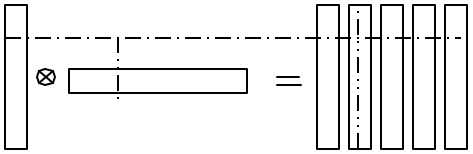
Inner product: u '*v = dot(u,v) ; sum=0;
for i=1:length(u), sum = sum+u(i)*v(i); end

$$< u, v >= u'v = \sum_{i=1}^{n} u_i v_i$$

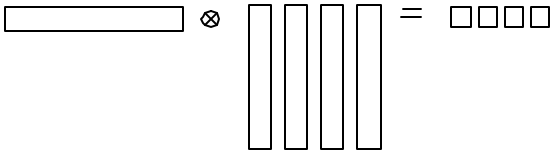


Outer Product: u*v';
for i=1:length(u), for j=1:length(v),
uv(i,j) = u(i)*v(j); end,end

$$u \otimes v = uv' = (u_i v_j)$$

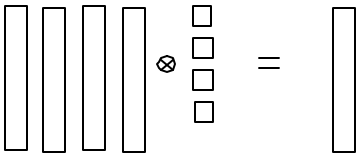


1xN row X Nx4 matrix = 1x4 row



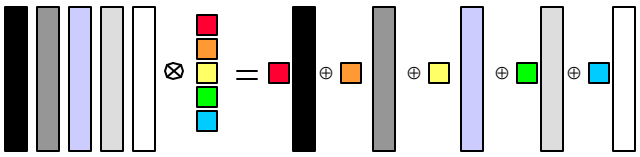
1

Nx4 matrix X 4x1 column = Nx1 column

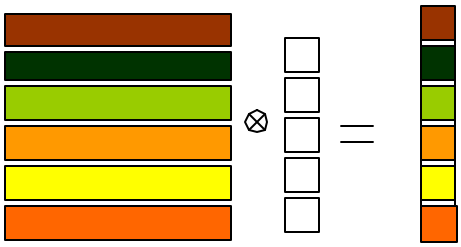


```
%----------------------------------------
% (1) matrix * vector  Sum A(:,i)*x(i)
%     Matrix*vector
%----------------------------------------
%     MatVecCo
function  y = MatVecCO(A,x)
% y = MatVexCO(A,x)
[m,n] = size(A);y = zeros(m,1);
for j = 1:n   y = y + A(:,j)*x(j);end
%
% /$\       / --$----- \      -
% |$|       | --$----- |      -
% |$|   ___ | --$----- |      $
% |$|   \   | --$----- |      -
% |$| =  >  | --$----- |  *   -
% |$|   /__ | --$----- |      -
% |$|       | --$----- |      -
% |$|       | --$----- |      -
% \$/       \ --$----- /
%----------------------------------------
```

Matrix-Vector Multiplication: The COLUMN picture



```
%-----------------------------------------------
%   matrix*vector  by rows A(i,:)*x
%-----------------------------------------------
%        MatVecRo
function  y = MatVecRO(A,x)
% y = MatVexRO(A,x)
[m,n] = size(A);y = zeros(m,1);
for i = 1:m   y(i) = A(i,:)*x;end
%   /-\    / -------- \     /$\
%   |-|    | -------- |     |$|
%   |$| =  | $$$$$$$$ |     |$|
%   |-|    | -------- |  *  |$|
%   |-|    | -------- |     |$|
%   |-|    | -------- |     |$|
%   \_/    \ -------- /     \$/
```

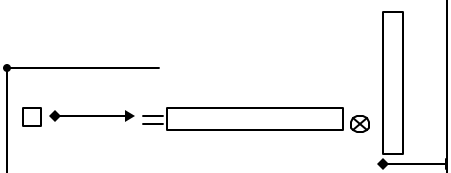


Matrix-Vector multiplication:
The ROW picture

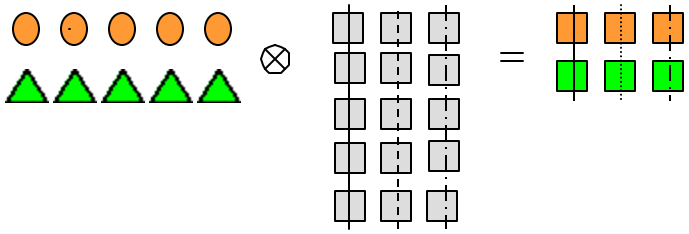
Four ways for matrix products

```
%-------------------------------
% Matrix-Matrix multiplication
% (1) MatMatDot (row or column)
% (2) MatMatOuter
% (3) MatMatSax (row or column)
% (4) MatMatVec (row or column)
%-------------------------------
```

```
%-----------------------------------------------
function C = MatMatDotr(A,B)
% C = MatMatDot(A,B)
% computes matrix*matrix product C =A*B(via dot products)
% A(mXp), B(pXn), c(mXn). For i=1:m, build i-th row of C,
% by computing the dot product of i-th row of A by columns of B;
[m,p] = size(A);[p,n] = size(B);C = zeros(m,n);
for i = 1:m
% Compute i-th row of C
  for j = 1:n      C(i,j) = A(i,:)*B(:,j); end
end
%       3  6             8          3  6
%    /--|---\   / -------- \     /--$---\
%    |--|---|   | -------- |     |--$---|
% i  |--|---|   | -------- |     |--$---|
%   4|--$---| = | $$$$$$$$ |4    |--$---|
%    |--|---|   | -------- |  *  |--$---|
%    |--|---|   | -------- |     |--$---|
%   7\--|---/  7\ -------- /     |--$---|        i-th row    i-th row     columns
%       j->                   8\--$---/          of C        of A        of B
%       C            A           B  j->
%-----------------------------------------------
```



A{2x5} x B{5x3} = C{2x3}



The rows of the product are formed by combining the rows of B in proportions specified by the rows of A:

in forming the j-th row of C:

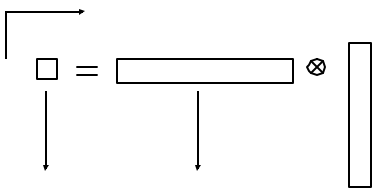
the rows of B are the “ingredients”

the j–th row of A is the “recipe”

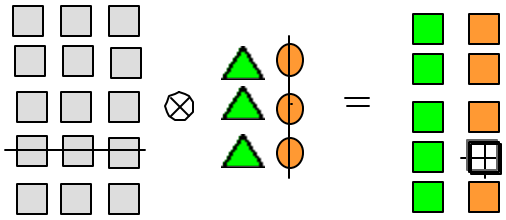
THE ROW PICTURE

```
MatVecMat(A,B)
% computes matrix*matrix product C = A*B via vector-matrix products
% A is an mXp matrix, B a pXn matrix.
[m,p] = size(A);[p,n] = size(B);
C = zeros(m,n);
for i = 1:m
% Compute i-th row of C
C(i,:) = A(i,:)*B;
end
%     6               8             6
%  /------\     / -------- \     /------\
%  |------|     | -------- |     |------|
%  |------|     | -------- |     |------|
%  |$$$$$$| =   | SSSSSSSS |     |------|
%  |------|     | -------- |  *  |------|
%  |------|     | -------- |     |------|
% 7\------/    7\ -------- /     |------|
%     j=3                       8\------/
%                                  j=3
```

```
%-----------------------------------------------
function C = MatMatDot(A,B)
% C = MatMatDot(A,B)
% computes matrix*matrix product C =A*B(via dot products)
% A(mXp), B(pXn), c(mXn). For j=1:n, build j-th column of C,
% by computing the dot product of rows 1:m of A by j-th column of B;
[m,p] = size(A);[p,n] = size(B);C = zeros(m,n);
for j = 1:n
% Compute j-th column of C
  for i = 1:m      C(i,j) = A(i,:)*B(:,j); end
end
%       3  6             8          3  6
%    /--|---\   / -------- \     /--$---\
%    |--|---|   | -------- |     |--$---|
% i  |--|---|   | -------- |     |--$---|
%   4|--$---| = | $$$$$$$$ |4    |--$---|
%    |--|---|   | -------- |  *  |--$---|
%    |--|---|   | -------- |     |--$---|
%   7\--|---/  7\ -------- /     |--$---|        j-th col    rows
%       j->                   8\--$---/          of C      of A
%       C             A           B  j->
%-----------------------------------------------
```



5x3 matrix X 3x2 matrix = 5x2 matrix



The columns of the product are formed by combining the columns of A in proportions specified by the columns of B:

in forming the i-th column of C:

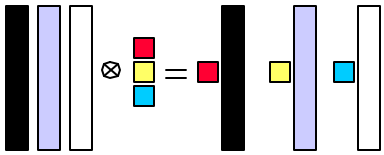
the columns of A are the “ingredients”

the i–th column of B is the “recipe”

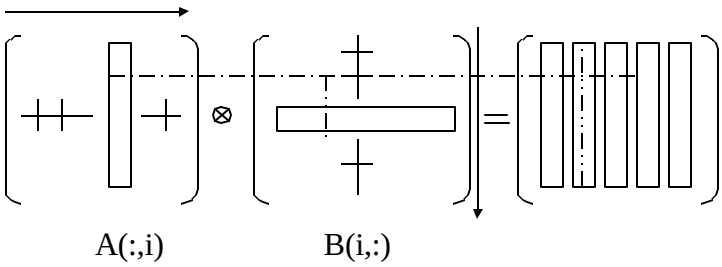
THE COLUMN PICTURE

```
MatMatVec(A,B)
% computes matrix*matrix product C = A*B via matrix-vector products
% A is an mXp matrix, B a pXn matrix.
[m,p] = size(A);[p,n] = size(B);
C = zeros(m,n);
for j = 1:n
% Compute j-th column of C
C(:,j) = A*B(:,j);%C contains zeros:  C(:,j) = C(:,j) + A*B(:,j);
end
%     6               8             6
%  /--$---\     / -------- \     /--$---\
%  |--$---|     | -------- |     |--$---|
%  |--$---|     | -------- |     |--$---|
%  |--$---| =   | -------- |     |--$---|
%  |--$---|     | -------- |  *  |--$---|
%  |--$---|     | -------- |     |--$---|
% 7\--$---/    7\ -------- /     |--$---|
%     j=3                       8\--$---/
%                                  j=3
```

```
function C = MatMatSax(A,B)
% computes matrix*matrix product C = A*B (via saxpys) where
% A is an mXp matrix, B a pXn matrix.[m,p] = size(A);
[p,n] = size(B);C = zeros(m,n);
for j = 1:n
% Compute j-th column of C
   for k = 1:p      C(:,j) = C(:,j) + A(:,k)*B(k,j); end
end
%        6                 8             6
%  /--$---\        / ----$--- \     /--.---\
%  |--$---|   _8_  | ----$--- |     |--.---|
%  |--$---|   \    | ----$--- |     |--.---|
%  |--$---| =  >   | ----$--- |     |--.---|
%  |--$---|   /__  | ----$--- |  * k|--$---|
%  |--$---|   k=1  | ----$--- |     |--.---|
% 7\--$---/       7\ ----$--- /     |--.---|
%     j=3               k=4       8\--.---/
%                                     j=3
%     C                 A             B
% here, a column of C is built by adding the contributions from
% all the columns of A, each multiplied by the corresponding element in B
```



```
function C = MatMatOuter(A,B)
% C = MatMatOuter(A,B)
% computes matrix*matrix product C = A*B
%     (via outer products) where
% A is an mXp matrix, B a pXn matrix.
[m,p] = size(A);[p,n] = size(B);C = zeros(m,n);
for k = 1:p
% Add-in k-th outer product
C = C + A(:,k)*B(k,:);
end
%        6                    8             6
%  /------\          / ----$--- \     /------\
%  |------|   _8_   | ----$--- |     |------|
%  |------|   \     | ----$--- |     |------|
%  |------| =  >    | ----$--- |     |------|
%  |------|   /__   | ----$--- |  *  |$$$$$$|k=5
%  |------|   k=1   | ----$--- |     |------|
% 7\------/        7\ ----$--- /     |------|
%                        k=5       8\------/
%-------------------------------------------------------------
```



A(:,i) B(i,:)

```
%-------------------------------------------------
Review Cells
%-------------------------------------------------
Script: cellsNYC_Lat  =
struct('d',40,'m',45,'s',27);
NYC_Long = struct('d',75,'m',12,'s',32);
C1       = struct('name','New York','lat',…
                  NYC_Lat,'long',NYC_Long);
C1.nameC1.long.mC=cell(2,2);
C{1,1}=[1 2;3 4];
C{1,2}=[5;6];
C{2,1}=[7 8];
C{2,2}=9;
M=[C{1,1} C{1,2};C{2,1} C{2,2}]
%-------------------------------------------------
```

The Pascal Triangle

1 … … ……
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1

$$P\{1\} = [1]$$
$$P\{2\} = [1 \quad 1]$$
$$P\{3\} = [1 \quad 2 \quad 1]$$
$$P\{4\} = [1 \quad 3 \quad 3 \quad 1]$$
$$P\{5\} = [1 \quad 4 \quad 6 \quad 4 \quad 1]$$
$$P\{k+1\} = [0 \quad P\{k\}] + [P\{k\} \quad 0]$$

$$(x+1)^{k+1} = (x+1)*(x+1)^k \Rightarrow$$
$$(a_{k+1,0} + a_{k+1,1}x + \cdots + a_{k+1,k}x^k + a_{k+1,k+1}x^{k+1}) = (x+1)(a_{k,0} + a_{k,1}x + \cdots + a_{k,k}x^k) \Rightarrow$$
$$a_{k+1,l} = a_{k,l-1} + a_{k,l}$$

Polynomial Recurrence Relations

$$xP_n = a_{n,n+1}P_{n+1} + a_{n,n}P_n + a_{n,n-1}P_{n-1}$$

$$P_{n+1} = \frac{(x - a_{n,n})}{a_{n,n+1}}P_n - \frac{a_{n,n-1}}{a_{n,n+1}}P_{n-1} = (b_n x + c_n)P_n + d_n P_{n-1}$$

$$P_0 = q_{0,0} \qquad P_1 = q_{1,0} + q_{1,1}x$$

$$P_n = q_{n,0} + q_{n,1}x + q_{n,2}x^2 + \cdots + q_{n,n-1}x^{n-1} + q_{n,n}x^n$$

$$P_n \rightarrow [q_{n,0}, q_{n,1}, q_{n,2}, \ldots, q_{n,n-1}, q_{n,n}] \ldots\ldots (n+1)$$

$$xP_n \rightarrow [0, q_{n,0}, q_{n,1}, q_{n,2}, \ldots, q_{n,n-1}, q_{n,n}] \ldots (n+2)$$

$$P_{n+1} = \left[q_{n+1,0}, q_{n+1,1}, q_{n+1,2}, \ldots, q_{n+1,n}, q_{n+1,n+1}\right].....(n+2)$$

$$(b_n x + c_n)P_n = \left[q_{n,0}, q_{n,1}, q_{n,2}, \ldots, q_{n,n-1}, q_{n,n}\right] =$$
$$\_\_\_\_ = b\left[0, q_{n,0}, q_{n,1}, \ldots, q_{n,n-1}, q_{n,n}\right]....(n+2)$$
$$\_\_\_\_ + c\left[q_{n,0}, q_{n,1}, q_{n,2}, ..., q_{n,n}, 0\right].....(n+2)$$

$$d_n P_{n-1} = d_n\left[q_{n-1,0}, q_{n-1,1}, q_{n-1,2}, \ldots, q_{n-1,n-1}\right].....(n)$$
$$\_\_\_\_ = d_n\left[q_{n-1,0}, q_{n-1,1}, q_{n-1,2}, \ldots, q_{n-1,n-1}, 0, 0\right].....(n+2)$$

$$P\{n+1\} = b_n\left[0, P\{n\}\right] + c_n\left[P\{n\}, 0\right] + d_n\left[P\{n-1\}, 0, 0\right]$$

This relationship is correct, but the indices are "mathematical"; to convert to Matlab, add 1 to everything!

Summary

- Matrices of special structure
- 2 forms for Mat*Vec
- 6(7) forms for Mat*Mat
- Using Cell Arrays for Polynomial recurrence
- Matlab commands for displaying matrices

References

www.whatisthematrix.com
C.F.van Loan,
Intro to Sci.Comp. (5.1-2)
Higham & Higham, Matlab Guide, Ch.9