Introduction to MATLAB and Linear Algebra

Ralph C. Smith
Department of Mathematics
North Carolina State University

Overview:

1. MATLAB examples throughout tutorial
2. Elements of linear algebra
    - Fundamental properties of vectors and matrices
    - Eigenvalues, eigenvectors and singular values

Linear Algebra and Numerical Matrix Theory:

- Vector properties including orthogonality
- Matrix analysis, inversion and solving Ax = b for very large systems
- Eigen- and singular value decompositions

Linear Algebra and Numerical Matrix Theory

Topics: Illustrate with MATLAB as topics are introduced

- Basic concepts
- Linear transformations
- Linear independence, basis vectors, and span of a vector space
- Fundamental Theorem of Linear Algebra
- Determinants and matrix rank
- Eigenvalues and eigenvectors
- Solving Ax = b and condition numbers
- Singular value decomposition (SVD)
- Cholesky and QR decompositions

Linear Algebra has become as basic and as applicable as calculus, and fortunately it is easier.

--Gilbert Strang, MIT

Vectors and Matrices

Note:

- *Vector* in R^n is an ordered set of n real numbers.
  - e.g. v = (1,6,3,4) is in R^4
  - "(1,6,3,4)" is a column vector: $\begin{pmatrix} 1 \\ 6 \\ 3 \\ 4 \end{pmatrix}$
  - as opposed to a row vector: $\begin{pmatrix} 1 & 6 & 3 & 4 \end{pmatrix}$
- m-by-n *matrix* is an object with m rows and n columns, each entry fill with a real number:

$$\begin{pmatrix} 1 & 2 & 8 \\ 4 & 78 & 6 \\ 9 & 3 & 2 \end{pmatrix}$$

Vector Norms

Vector Norms: The norm ||x|| quantifies the ``length''

$$||x||_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

Common Norms:

$$||x||_1 = \sum_{i=1}^{n} |x_i|$$

$$||x||_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$$ Euclidean: Vector length

$$||x||_\infty = \max_i |x_i|$$

Vector and Matrix Products

Vector products:

- Dot product: $u \bullet v = u^T v = \begin{pmatrix} u_1 & u_2 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = u_1 v_1 + u_2 v_2$

Note: $A \cdot B = \|A\| \, \|B\| \cos(\theta)$

If u•v=0, $||u||_2$!= 0, $||v||_2$!= 0 → *u and v are orthogonal*

If u•v=0, $||u||_2$ = 1, $||v||_2$ = 1 → *u and v are orthonormal*

- Outer product: $uv^T = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \begin{pmatrix} v_1 & v_2 \end{pmatrix} = \begin{pmatrix} u_1 v_1 & u_1 v_2 \\ u_2 v_1 & u_2 v_2 \end{pmatrix}$

Matrix Product:

$$A \in \mathbb{R}^{m \times n} \quad B \in \mathbb{R}^{n \times p}$$

$$C_{ij} = \sum_{k=1} A_{ik} B_{kj} \quad C = AB \in \mathbb{R}^{m \times p}$$

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, B = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$$

$$AB = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{pmatrix}$$

Special Matrices

$$\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix}$$ diagonal

$$\begin{pmatrix} a & b & c \\ 0 & d & e \\ 0 & 0 & f \end{pmatrix}$$ upper-triangular

$$\begin{pmatrix} a & b & 0 & 0 \\ c & d & e & 0 \\ 0 & f & g & h \\ 0 & 0 & i & j \end{pmatrix}$$ tri-diagonal

$$\begin{pmatrix} a & 0 & 0 \\ b & c & 0 \\ d & e & f \end{pmatrix}$$ lower-triangular

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$ I (identity matrix)

Special Matrices

- Matrix A is *symmetric* if $A = A^T$
- A is *positive definite* if $x^TAx>0$ for all non-zero x (*positive semi-definite* if inequality is not strict)

$$\begin{pmatrix} a & b & c \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} a \\ b \\ c \end{pmatrix} = a^2 + b^2 + c^2 \quad \begin{pmatrix} a & b & c \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} a \\ b \\ c \end{pmatrix} = a^2 - b^2 + c^2$$

- Useful fact: Any matrix of form A^TA is positive semi-definite.

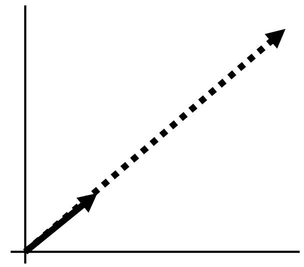
  To see this, $x^T(A^TA)x = (x^TA^T)(Ax) = (Ax)^T(Ax) \geq 0$

Recall: Covariance matrix

$$V = \sigma^2(X^TX)^{-1}$$

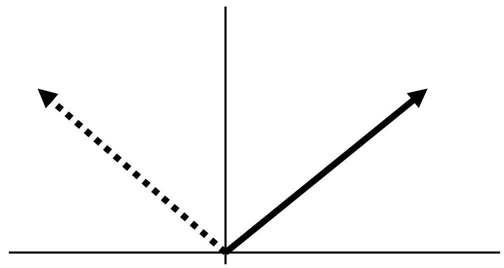
Matrices as Linear Transformations

$$\begin{pmatrix} 5 & 0 \\ 0 & 5 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 5 \\ 5 \end{pmatrix}$$



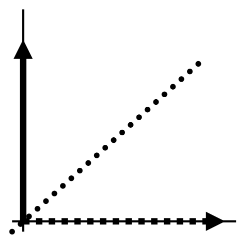
(stretching)

$$\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$



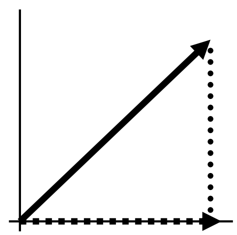
(rotation)

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



(reflection)

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



(projection)

Linear Independence

- A set of vectors is linearly independent if none of them can be written as a linear combination of the others.
- Vectors $v_1,\ldots,v_k$ are linearly independent if $c_1v_1+\ldots+c_kv_k = 0$ implies $c_1=\ldots=c_k=0$

$$\begin{pmatrix} | & | & | \\ v_1 & v_2 & v_3 \\ | & | & | \end{pmatrix}\begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

e.g.

$$\begin{pmatrix} 1 & 0 \\ 2 & 3 \\ 1 & 3 \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

(u,v)=(0,0), i.e. the columns are linearly independent.

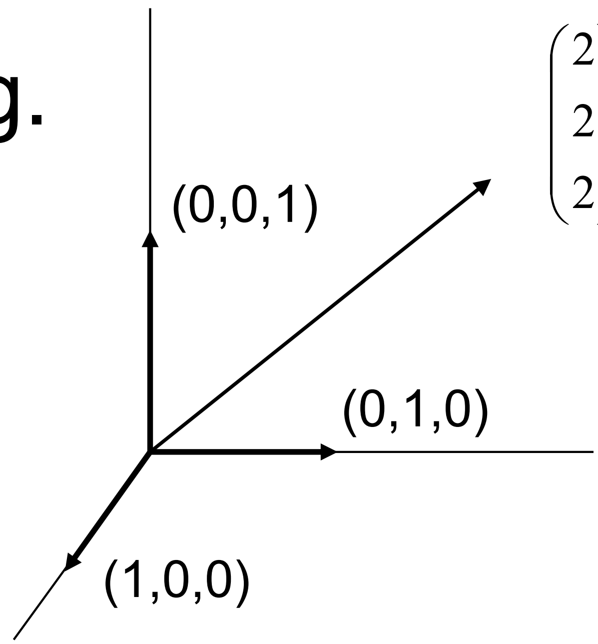
$$x_1 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \quad x_2 = \begin{bmatrix} 4 \\ 1 \\ 5 \end{bmatrix} \quad x_3 = \begin{bmatrix} 2 \\ -3 \\ -1 \end{bmatrix}$$

x3 = −2x1 + x2

Basis Vectors

- If all vectors in a vector space may be expressed as linear combinations of a set of vectors $v_1,\ldots,v_k$, then $v_1,\ldots,v_k$ spans the space.
- The cardinality of this set is the dimension of the vector space.

e.g.



$$\begin{pmatrix}2\\2\\2\end{pmatrix} = 2\begin{pmatrix}1\\0\\0\end{pmatrix} + 2\begin{pmatrix}0\\1\\0\end{pmatrix} + 2\begin{pmatrix}0\\0\\1\end{pmatrix}$$

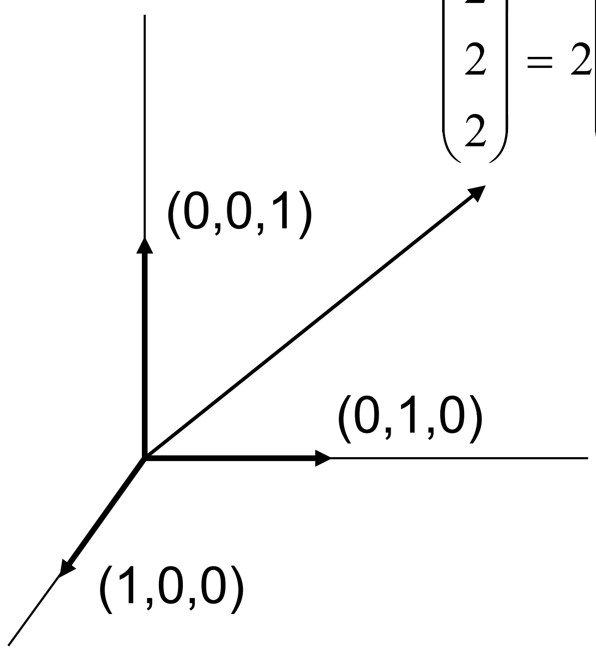
- A basis is a maximal set of linearly independent vectors and a minimal set of spanning vectors of a vector space

Basis Vectors

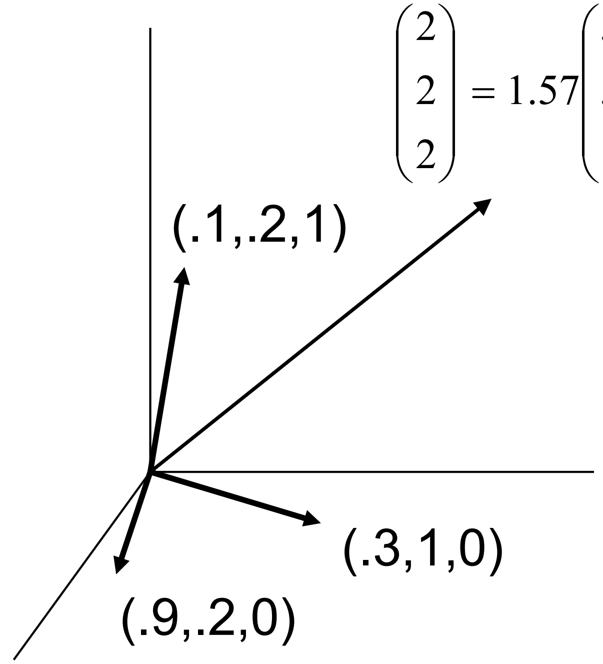
Note:

- An *orthonormal basis* consists of orthogonal vectors of unit length.

$$\begin{pmatrix}2\\2\\2\end{pmatrix} = 2\begin{pmatrix}1\\0\\0\end{pmatrix} + 2\begin{pmatrix}0\\1\\0\end{pmatrix} + 2\begin{pmatrix}0\\0\\1\end{pmatrix}$$



$$\begin{pmatrix}2\\2\\2\end{pmatrix} = 1.57\begin{pmatrix}.9\\.2\\0\end{pmatrix} + 1.29\begin{pmatrix}.3\\1\\0\end{pmatrix} + 2\begin{pmatrix}.1\\.2\\1\end{pmatrix}$$



Rank of a Matrix

- The *rank* of A is the dimension of the column space of A.
- It also equals the dimension of the *row space* of A (the subspace of vectors which may be written as linear combinations of the rows of A).

$$\begin{pmatrix} 1 & 0 \\ 2 & 3 \\ 1 & 3 \end{pmatrix}$$

(1,3) = (2,3) – (1,0)

Only 2 linearly independent rows, so rank = 2.

Fundamental Theorem of Linear Algebra:

If A is m x n with rank r,

Column space(A) has dimension r

Nullspace(A) has dimension n-r (= *nullity* of A)

Row space(A) = Column space(A^T) has dimension r

Left nullspace(A) = Nullspace(A^T) has dimension m - r

Rank-Nullity Theorem: rank + nullity = n

Matrix Inverse

Note:

- To solve $Ax=b$, we can write a closed-form solution if we can find a matrix A^{-1} such that $AA^{-1} = A^{-1}A = I$ (Identity matrix)
- Then $Ax=b$ iff $x=A^{-1}b$:
  $x = Ix = A^{-1}Ax = A^{-1}b$
- A is *non-singular* iff A^{-1} exists iff $Ax=b$ has a unique solution.
- Note: If A^{-1}, B^{-1} exist, then $(AB)^{-1} = B^{-1}A^{-1}$, and $(A^T)^{-1} = (A^{-1})^T$

- For orthonormal matrices

$$A^{-1} = A^T$$

Matrix Determinants

Note:

- If det(A) = 0, then A is singular.
- If det(A) ≠ 0, then A is invertible.
- To compute:
  - Simple example:

$$\det\begin{pmatrix} a & b \\ c & d \end{pmatrix} = ad - bc$$

  - Matlab: det(A)

MATLAB Interlude

Special Variables:

- Special variables:
  - ans : default variable name for the result
  - pi: π = 3.1415926…………
  - eps: $\in$ = 2.2204e-016, smallest amount by which 2 numbers can differ.
  - Inf or inf : ∞, infinity
  - NaN or nan: not-a-number

Vectors: Example:

x is a row vector.

```
>> x = [ 0   0.25*pi   0.5*pi   0.75*pi   pi ]
x =
        0    0.7854    1.5708    2.3562    3.1416
```

y is a column vector.

```
>> y = [ 0; 0.25*pi; 0.5*pi; 0.75*pi; pi ]
y =
        0
   0.7854
   1.5708
   2.3562
   3.1416
```

Vectors

- Vector Addressing – A vector element is addressed in MATLAB with an integer index enclosed in parentheses.
- Example:

```
>> x(3)
ans =
    1.5708
```
← 3rd element of vector x

- The colon notation may be used to address a block of elements.

(start : increment : end)

start is the starting index, increment is the amount to add to each successive index, and end is the ending index. A shortened format (start : end) may be used if increment is 1.

- Example:

```
>> x(1:3)
ans =
         0    0.7854    1.5708
```
← 1st to 3rd elements of vector x

NOTE: MATLAB index starts at 1.

Vectors

Some useful commands:

x = start:end	create row vector x starting with start, counting by one, ending at end
x = start:increment:end	create row vector x starting with start, counting by increment, ending at or before end
linspace(start,end,number)	create row vector x starting with start, ending at end, having number elements
length(x)	returns the length of vector x
y = x’	transpose of vector x
dot (x, y)	returns the scalar dot product of the vector x and y.

Matrices

- A Matrix array is two-dimensional, having both multiple rows and multiple columns, similar to vector arrays:
  - it begins with [, and end with]
  - spaces or commas are used to separate elements in a row
  - semicolon or enter is used to separate rows.

- Matrix Addressing:
  -- *matrixname(row, column)*
  -- **colon** may be used in place of a row or column reference to select the entire row or column.

•Example:
•>> f = [1 2 3; 4 5 6]

```
f =
    1    2    3
    4    5    6
```

- Example:

```
>> f(2,3)

ans =

    6

>> h(:,1)

ans =

    2

    1
```

recall:

```
f =
     1    2    3
     4    5    6

h =
     2    4    6
     1    3    5
```

Matrices

more commands

Transpose	B = A′
Identity Matrix	eye(n) ➔ returns an n x n identity matrix eye(m,n) ➔ returns an m x n matrix with ones on the main diagonal and zeros elsewhere.
Addition and subtraction	C = A + B C = A – B
Scalar Multiplication	B = αA, where α is a scalar.
Matrix Multiplication	C = A*B
Matrix Inverse	B = inv(A), A must be a square matrix in this case. rank (A) ➔ returns the rank of the matrix A.
Matrix Powers	B = A.^2 ➔ squares each element in the matrix C = A * A ➔ computes A*A, and A must be a square matrix.
Determinant	det (A), and A must be a square matrix.

A, B, C are matrices, and m, n, α are scalars.

Matrices

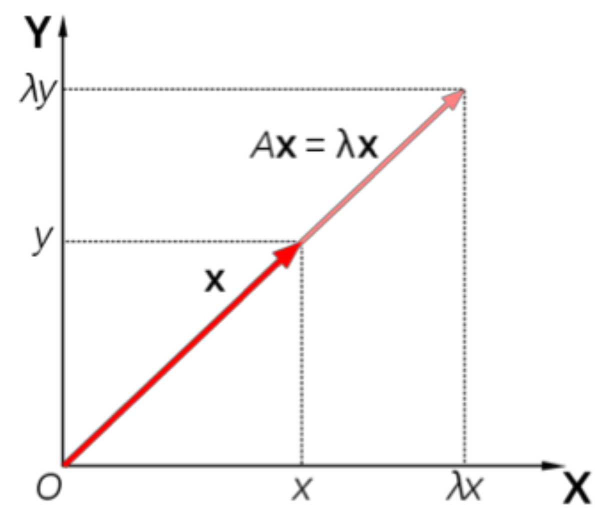
```
>> A = [1 2;3 4]
 A =
      1     2
      3     4

>> B = [2 0;2 1]
B =
     2     0
     2     1

>> A*B
ans =
     6     2
    14     4

>> A.*B
ans =
     2     0
     6     4
```

```
>> A = [1 2 3;0 2 0]
A =
     1     2     3
     0     2     0

>> B = [1;-1;0]
B =
     1
    −1
     0

>> A*B
ans =
    −1
    −2
```

Eigenvalues and Eigenvectors

Note:

- How can we characterize matrices?
- The solutions to Ax = λx in the form of eigenpairs (λ,x) = (eigenvalue,eigenvector) where x is non-zero
- To solve this, (A – λI)x = 0
- λ is an eigenvalue iff det(A – λI) = 0



Example:

$$A = \begin{pmatrix} 1 & 4 & 5 \\ 0 & 3/4 & 6 \\ 0 & 0 & 1/2 \end{pmatrix}$$

$$\det(A - \lambda I) = \begin{pmatrix} 1-\lambda & 4 & 5 \\ 0 & 3/4-\lambda & 6 \\ 0 & 0 & 1/2-\lambda \end{pmatrix} = (1-\lambda)(3/4-\lambda)(1/2-\lambda)$$

$$\lambda = 1, \lambda = 3/4, \lambda = 1/2$$

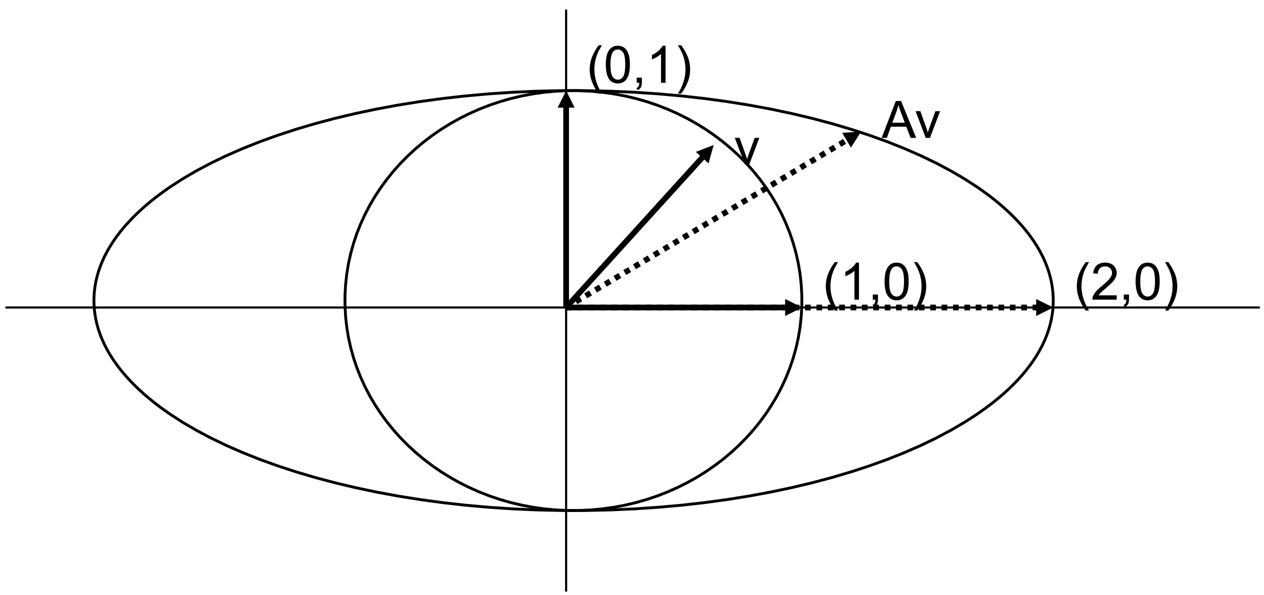
Eigenvalues and Eigenvectors

Example:

$$A = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}$$

Eigenvalues λ = 2, 1 with eigenvectors (1,0), (0,1)

Eigenvectors of a linear transformation A are not rotated (but will be scaled by the corresponding eigenvalue) when A is applied.



Important Results:

- Eigenvalues of nxn symmetric matrix are real but may not be distinct. There are n orthogonal eigenvectors.

Eigenvalues and Eigenvectors

Important Results:

- Eigenvalues of nxn symmetric matrix are real but may not be distinct. There are n orthogonal eigenvectors; e.g., Identity matrix
- Eigenvalues of a positive definite matrix are positive.
- MATLAB Command: [V,D] = eig(A)
- Suppose A has n linearly independent eigenvectors. Then

$$Q = [v_1 \quad v_2 \ \cdots \ v_n]$$

$$\Rightarrow AQ = [\lambda_1 v_1 \quad \lambda_2 v_2 \ \cdots \ \lambda_n v_n]$$

$$\Rightarrow AQ = Q\Lambda$$

$$\Rightarrow A = Q\Lambda Q^{-1}$$

$$\Rightarrow Q^{-1}AQ = \Lambda$$

Note:

- The matrix Q thus provides a new basis for A.

Eigenvalues and Eigenvectors

Example:

```
>> A = [2 1 3;-2 0 -2;5 5 2]
A =
     2     1     3
    −2     0    −2
     5     5     2

>> X = A'*A
X =
    33    27    20
    27    26    13
    20    13    17

>> [Q,D] = eig(X)
Q =
   −0.7178    0.0398    0.6952
    0.5287   −0.6185    0.5813
    0.4531    0.7848    0.4229

D =
    0.4864         0         0
         0    7.7683         0
         0         0   67.7452
```

```
>>Q*D*inv(Q)
ans =
   33.0000   27.0000   20.0000
   27.0000   26.0000   13.0000
   20.0000   13.0000   17.0000

>> Q'
ans =
   −0.7178    0.5287    0.4531
    0.0398   −0.6185    0.7848
    0.6952    0.5813    0.4229

>> inv(Q)
ans =
   −0.7178    0.5287    0.4531
    0.0398   −0.6185    0.7848
    0.6952    0.5813    0.4229
```

Eigenvalues and Eigenvectors

Cholesky Decomposition: Matrix must be positive definite

```
>> A = [2 1 3;-2 0 -2;5 5 2]
A =
     2     1     3
    -2     0    -2
     5     5     2

>> X = A'*A
X =
    33    27    20
    27    26    13
    20    13    17

>> R = chol(X)
R =
   5.7446    4.7001    3.4816
        0    1.9771   -1.7013
        0         0    1.4087

>> R'*R
ans =
   33    27    20
   27    26    13
   20    13    17
```

Solving Ax = b

Notes:

- MATLAB: x = A\b;
  - How stable is the solution?
  - If A or b are changed slightly, how much does it effect x?
  - The *condition number* c of A measures this:

    $$c = \lambda_{max} / \lambda_{min}$$

  - Values of c near 1 are good.

```
>> A = [1 0;0 1e-10]
A =
    1.0000         0
         0    0.0000

>> cond(A)
ans =
    1.0000e+10>>
```

MATLAB

- <u>Example</u>: a system of 3 linear equations with 3 unknowns (x_1, x_2, x_3):

$$\begin{aligned} 3x_1 + 2x_2 - x_3 &= 10 \\ -x_1 + 3x_2 + 2x_3 &= 5 \\ x_1 - x_2 - x_3 &= -1 \end{aligned}$$

Let :

$$A = \begin{bmatrix} 3 & 2 & 1 \\ -1 & 3 & 2 \\ 1 & -1 & -1 \end{bmatrix} \qquad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad b = \begin{bmatrix} 10 \\ 5 \\ -1 \end{bmatrix}$$

Then, the system can be described as:

$$Ax = b$$

MATLAB

- <u>Solution by Matrix Inverse:</u>

  $Ax = b$

  $A^{-1}Ax = A^{-1}b$

  $x = A^{-1}b$

- MATLAB:

```
>> A = [ 3 2 -1; -1 3 2; 1 -1 -1];
>> b = [ 10; 5; -1];
>> x = inv(A)*b
x =
   -2.0000
    5.0000
   -6.0000
```

<u>Answer</u>:

x1 = -2, x2 = 5, x3 = -6

- <u>Solution by Matrix Division:</u>

The solution to the equation

$$Ax = b$$

can be computed using left division.

- MATLAB:

```
>> A = [ 3 2 -1; -1 3 2; 1 -1 -1];
>> b = [ 10; 5; -1];
>> x = A\b
x =
   -2.0000
    5.0000
   -6.0000
```

<u>Answer</u>:

x1 = -2, x2 = 5, x3 = -6

MATLAB: Flow Control

For Loops:

```
for j=1:5                % use for-loops to execute iterations / repetitions
    for i=1:3
        a(i, j) = i + j ;
    end
end
```

If Conditional:

```
a = zeros(3);  b = zeros(3);
for j=1:3
   for i=1:3
      a(i,j) = rand;          % use rand to generate a random number
      if a(i,j) > 0.5
         b(i,j) = 1;
      end
   end
end
```

MATLAB

Cell Arrays:

A cell array is a special array of arrays. Each element of the cell array may point to a scalar, an array, or another cell array.

```
>> C = cell(2, 3);   % create 2x3 empty cell array
>> M = magic(2);
>> a = 1:3; b = [4;5;6]; s = 'This is a string.';
>> C{1,1} = M; C{1,2} = a; C{2,1} = b; C{2,2} = s; C{1,3} = {1};
C =
   [2x2 double]     [1x3 double]        {1x1 cell}
   [2x1 double]     ‘This is a string.‘    []
>> C{1,1}     % prints contents of a specific cell element
ans =
    1     3
    4     2
>> C(1,:)      % prints first row of cell array C; not its content
```

MATLAB

Structures:

Ideal layout for grouping arrays that are related.

```
>> name(1).last = ‘Smith’;  name(2).last  = ‘Hess’;
>> name(1).first = ‘Mary’;   name(2).first = ‘Robert’;
>> name(1).sex = ‘female’; name(2).sex = ‘male’;
>> name(1).age = 45;         name(2).age = 50;
>> name(2)
ans =
     last: 'Hess'
    first: 'Robert'
      sex: 'male'
      age: 50
```

MATLAB

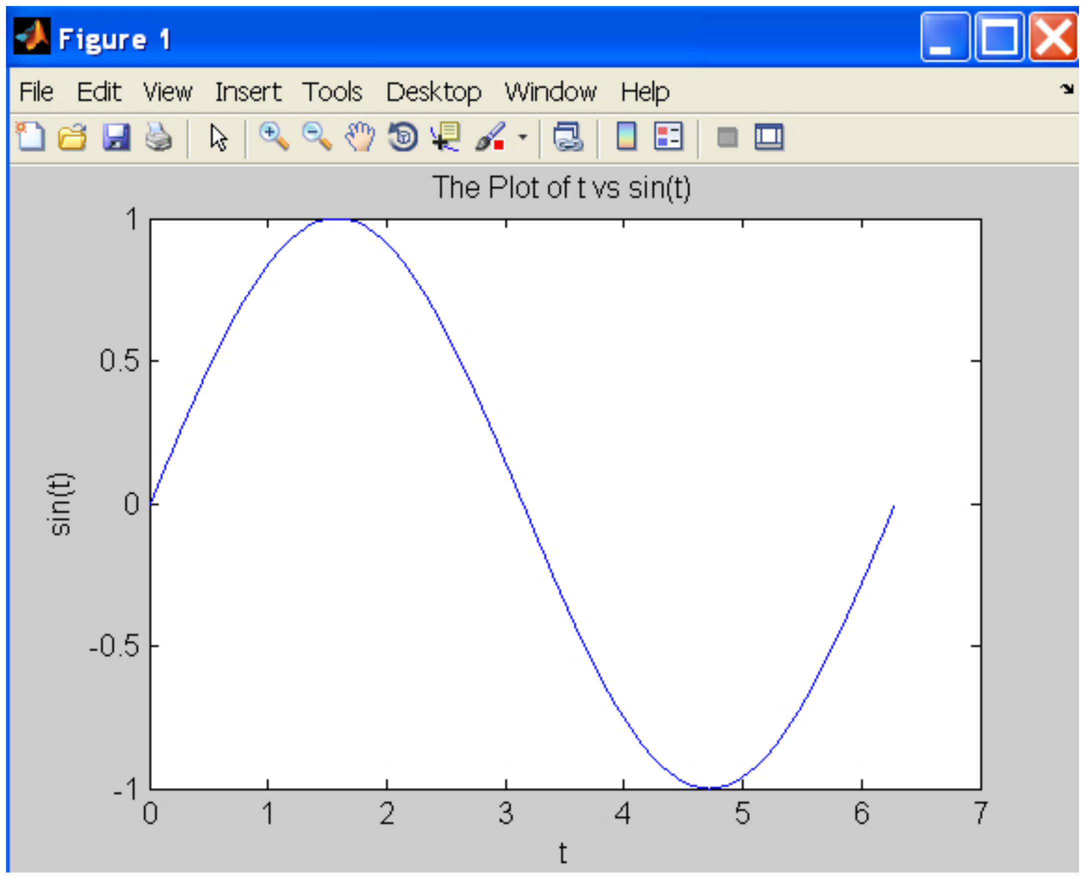
Frequently Used Functions:

```
>> magic(n)       % creates a special n x n  matrix; handy for testing
>> zeros(n,m)     % creates n x m matrix of zeroes (0)
>> ones(n,m)      % creates n x m matrix of ones (1)
>> rand(n,m)      % creates n x m matrix of random numbers
>> repmat(a,n,m)  % replicates a by n rows and m columns
>> diag(M)        % extracts the diagonals of a matrix M
>> help elmat     % list all elementary matrix operations ( or elfun)
>> abs(x);        % absolute value of x
>> exp(x);        % e to the x-th power
>> fix(x);        % rounds x to integer towards 0
>> log10(x);      % common logarithm of x to the base 10
>> rem(x,y);      % remainder of x/y
>> mod(x, y);     % modulus after division – unsigned rem
>> sqrt(x);       % square root of x
>> sin(x);        % sine of x; x in radians
>> acoth(x)       % inversion hyperbolic cotangent of x
```

Plotting

Line Plot:

```
>> t = 0:pi/100:2*pi;
>> y = sin(t);
>> plot(t,y)
>> xlabel('t');
>> ylabel('sin(t)');
>> title('The plot of t vs sin(t)');
```



Plotting

Customizing Graphical Effects

Generally, MATLAB's default graphical settings are adequate which make plotting fairly effortless. For more customized effects, use the *get* and *set* commands to change the behavior of specific rendering properties.

```
>> hp1 = plot(1:5)              % returns the handle of this line plot
>> get(hp1)                     % to view line plot's properties and their values
>> set(hp1, 'lineWidth')        % show possible values for lineWidth
>> set(hp1, 'lineWidth', 2)     % change line width of plot to 2
>> gcf              % returns current figure handle
>> gca              % returns current axes handle
>> get(gcf)         % gets current figure's property settings
>> set(gcf, 'Name', 'My First Plot')        % Figure 1 => Figure 1: My First Plot
>> get(gca)         % gets the current axes'  property settings
>> figure(1)        % create/switch to Figure 1 or pop Figure 1 to the front
>> clf              % clears current figure
>> close            % close current figure; "close 3" closes Figure 3
>> close all        % close all figures
```

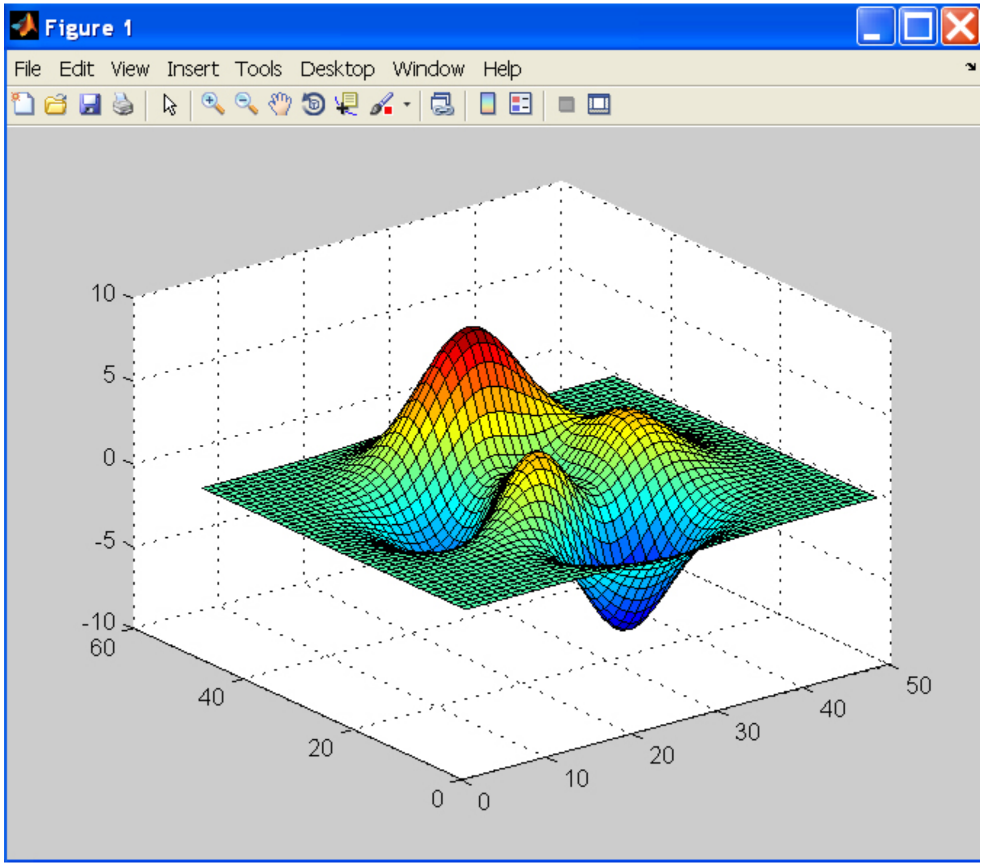
Plotting

Surface Plot

>> *Z = peaks;*   *% generate data for plot; peaks returns function values*
>> *surf(Z)*   *% surface plot of Z*

Try these commands also:

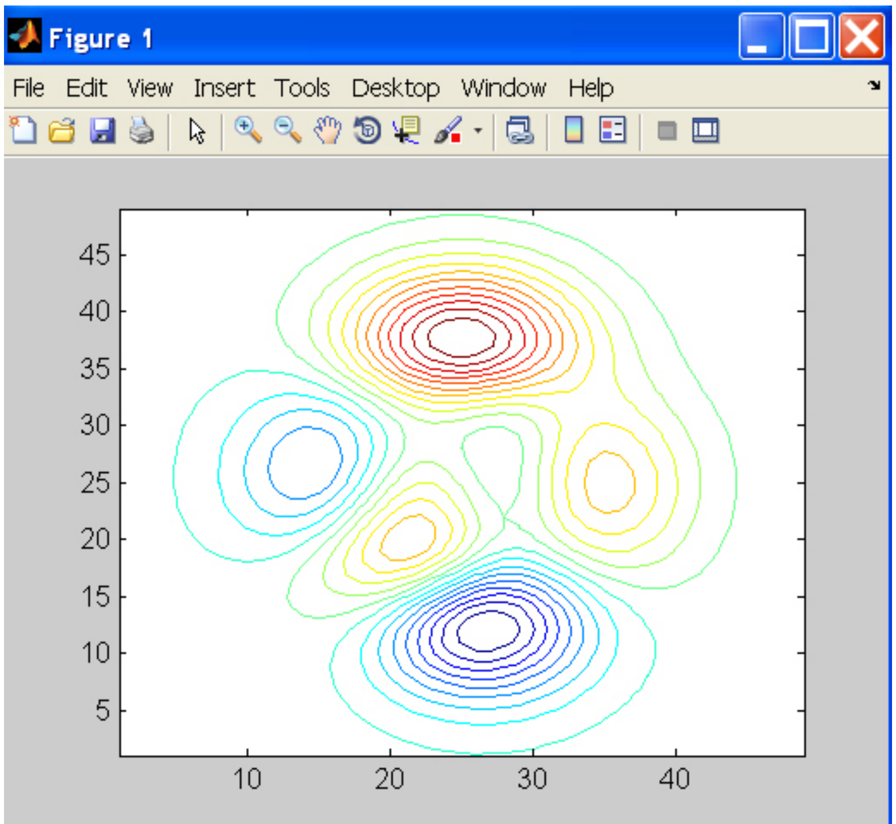
>> *shading flat*
>> *shading interp*
>> *shading faceted*
>> *grid off*
>> *axis off*
>> *colorbar*
>> *colormap('winter')*
>> *colormap('jet')*



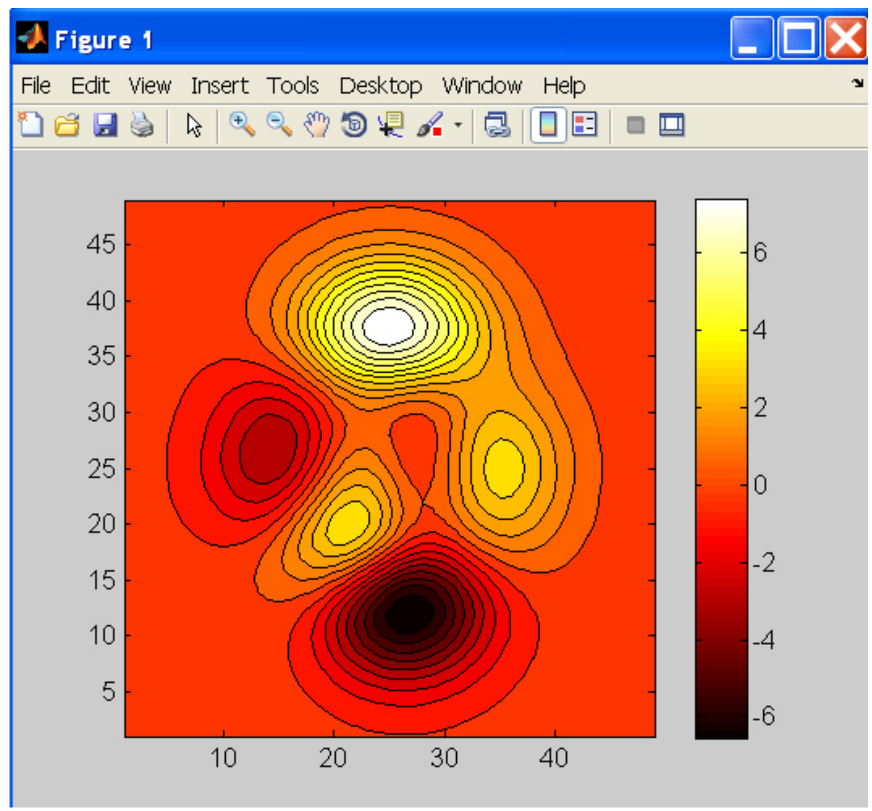
Plotting

Contour Plots

```
>> Z = peaks;
>> contour(Z, 20)    % contour plot of Z with 20 contours
```




```
>> contourf(Z, 20);  % with color fill
>> colormap('hot')   % map option
>> colorbar    % make color bar
```



Singular Value Decomposition

For an $m \times n$ matrix $\mathbf{A}$ of rank r there exists a factorization (Singular Value Decomposition = **SVD**) as follows:

$$A = U \Sigma V^T$$

$m \times m$ | $m \times n$ | V is $n \times n$

The columns of ***U*** are orthogonal eigenvectors of AA^T.

The columns of ***V*** are orthogonal eigenvectors of A^TA.

Eigenvalues $\lambda 1 \ldots \lambda r$ of AA^T are the eigenvalues of A^TA.

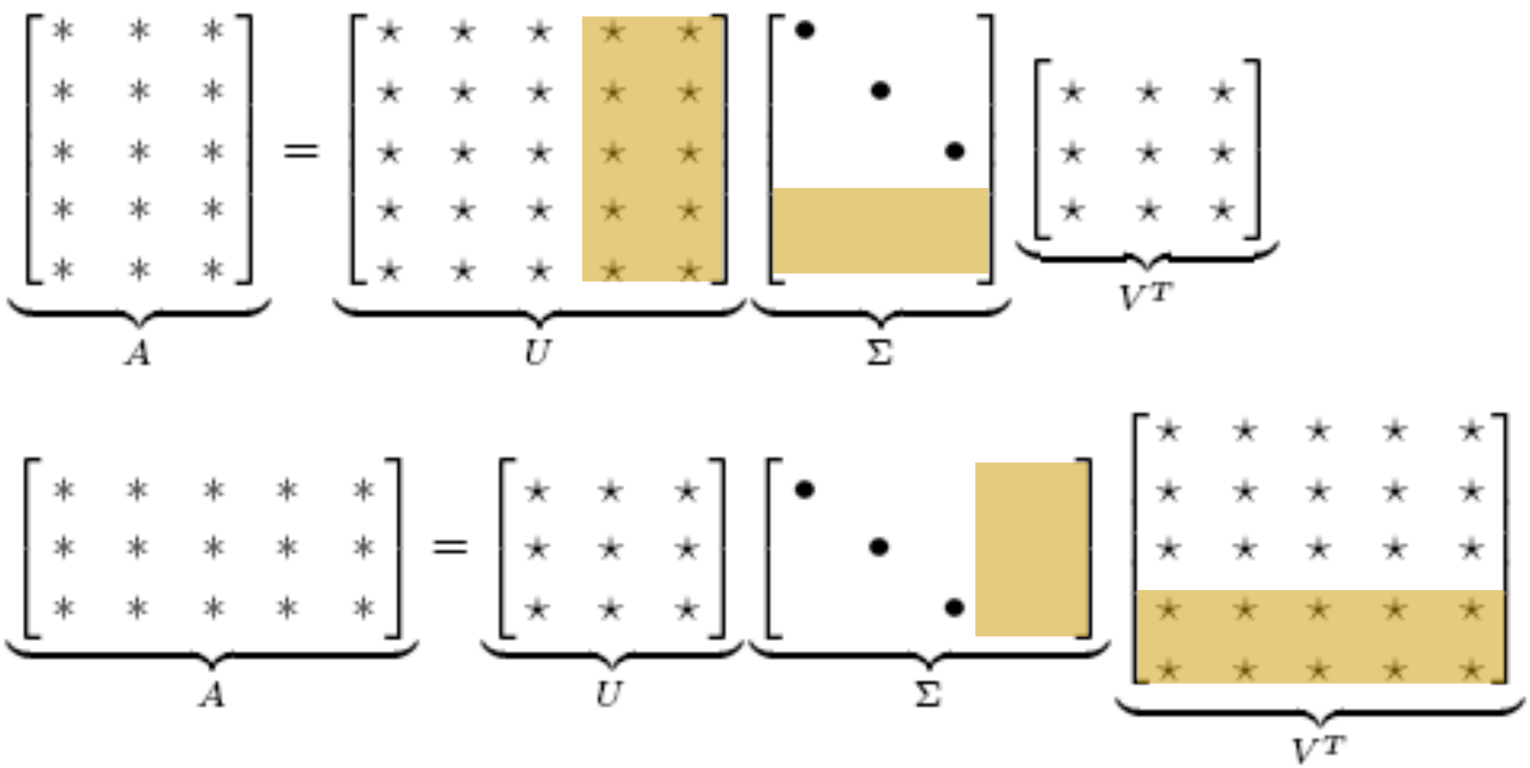
$$\sigma_i = \sqrt{\lambda_i}$$

$$\Sigma = diag(\sigma_1 \ldots \sigma_r)$$

Singular values.

Singular Value Decomposition

- Illustration of SVD dimensions and sparseness



Singular Value Decomposition

Let $$A = \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Thus m=3, n=2. Its SVD is

$$\begin{bmatrix} 0 & 2/\sqrt{6} & 1/\sqrt{3} \\ 1/\sqrt{2} & -1/\sqrt{6} & 1/\sqrt{3} \\ 1/\sqrt{2} & 1/\sqrt{6} & -1/\sqrt{3} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{3} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$$

Typically, the singular values arranged in decreasing order.