Probability and Statstics

Dr. Faisal Bukhari
Associate Professor
Department of Data Science
Faculty of Computing and Information Technology
University of the Punjab

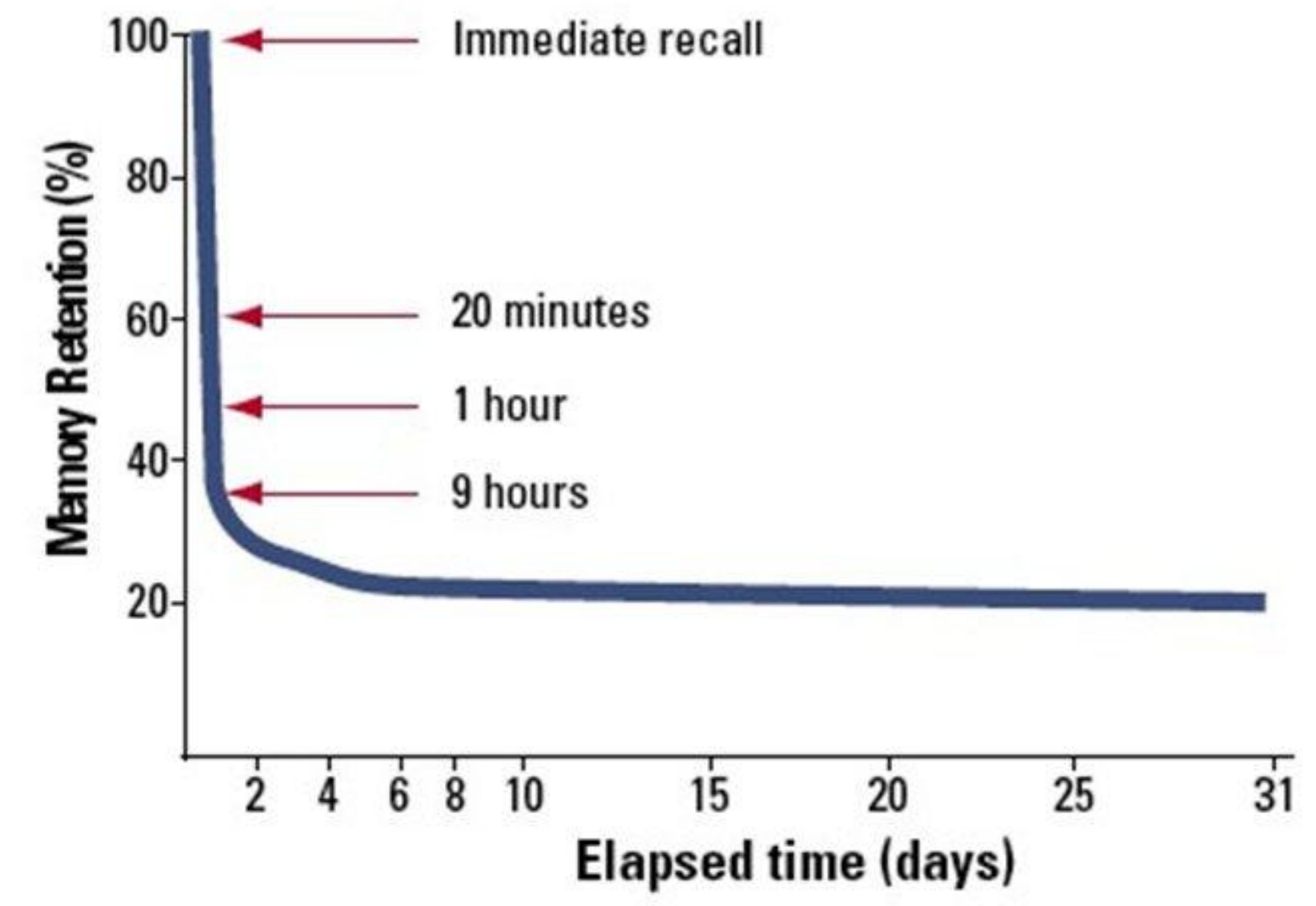
Reference

Readings for these lecture notes:

- ❑**Elementary Linear Algebra,Howard Anton and Chris Rorres**
- ❑**MATLAB Demystified, David McMahon**
- ❑**MATLAB® Primer, Seventh Edition, Timothy A. Davis Kermit Sigmon**
- ❑**http://tutorial45.com/matrix-division-matlab/**

These notes contain material from the above resources.

Forgetting curve



Introduction to MATLAB

MATLAB (Matrix Laboratory) is a high-level programming language and environment used for numerical computing.

- Developed by MathWorks.
- Widely used in engineering, science, and mathematics.
- Efficient for handling matrices and performing linear algebra operations.

MATLAB in Linear Algebra

MATLAB provides built-in support for:

- Matrix operations and manipulations.

- Solving systems of linear equations.

- Eigenvalues and eigenvectors.

- Singular Value Decomposition (SVD).

- Least squares solutions.

Basic MATLAB Matrix Operations

- Creating a Matrix: A = [1 2; 3 4]

- Matrix Addition: C = A + B

- Matrix Multiplication: C = A * B

- Transpose: A'

- Determinant: det(A)

- Inverse: inv(A)

Solving Systems of Equations

- MATLAB provides an efficient way to solve Ax = b using:
  x = A\b (left matrix division)

- More stable and efficient than computing inv(A)*b.

- Example:

```
A = [2 3; 4 5];
b = [1; 2];
x = A\b;
```

Eigenvalues and Eigenvectors

- Eigenvalues and eigenvectors satisfy $Ax = \lambda x$.

- MATLAB function:

```
[V, D] = eig(A)
```

- Example:

```
A = [2 1; 1 2];
[V, D] = eig(A);
```

- Useful in diagonalization and transformations.

Applications of MATLAB in Linear Algebra

MATLAB is used in various applications including:

- Machine Learning & AI

- Data Analysis & Visualization

- Image and Signal Processing

- Control Systems and Robotics

- Numerical Simulations and Optimization.

» lookfor inverse

INVHILB Inverse Hilbert matrix.

ACOS Inverse cosine.

ACOSH Inverse hyperbolic cosine.

ACOT Inverse cotangent.

ACOTH Inverse hyperbolic cotangent.

ACSC Inverse cosecant.

ACSCH Inverse hyperbolic cosecant.

ASEC Inverse secant.

ASECH Inverse hyperbolic secant.

ASIN Inverse sine.

ASINH Inverse hyperbolic sine.

ATAN Inverse tangent.

ATAN2 Four quadrant inverse tangent.

ATANH Inverse hyperbolic tangent.

ERFINV Inverse error function.

INV Matrix inverse.

PINV Pseudoinverse.

IFFT Inverse discrete Fourier transform.

IFFT2 Two-dimensional inverse discrete Fourier transform.

IFFTN N-dimensional inverse discrete Fourier transform.

IPERMUTE Inverse permute array dimensions.

From this list, we can see that the function being sought is named inv.

Inverse of matrix using Matlab

```
A = [1 2 3 4; 600 700 8002 2; 3 4 8 7; 56 3737 9 22]
B = inv(A)
B =

   -1.6816   -0.0003    0.9612   -0.0001
    0.0216    0.0000   -0.0132    0.0003
    0.1241    0.0001   -0.0709   -0.0000
    0.5666   -0.0000   -0.1806   -0.0001
```

Solution of system of equations

Solve the system of equation:

$x + y = 10$

$x - y = 2$

Matlab code:

```
A=[1 1; 1 -1]

B=[10 ; 2]

X=A\B

X =

   6
   4
```

Predicted team performances for T20 World Cup 2016

```
x = [0.3, 0.2, 0.2, 0.04, 0.02, 0.02, 0.21, 0.01];

labels = {'Pakistan', 'India', 'Australia', 'South Africa',  'Sri Lanka', 'New Zealand', 'West Indies', 'Bangladesh'};

figure

pie(x, labels)
```

MATLAB (Matrix Laboratory) [1]

- MATLAB **(matrix laboratory)** is a multi-paradigm numerical computing environment and fourth-generation programming language.

- Developed by MathWorks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and **interfacing** with programs written in other languages, including **C, C++, Java**, and **Fortran**.

Reference: http://en.wikipedia.org/wiki/MATLAB

Command Window and Basic Arithmetic [1]

The **Command Window** is found on the right-hand side of the MATLAB desktop. Commands are entered at the prompt with looks like two successive “greater than” signs:

>>

```
>> 433.12*15.7
    ans = 6.8000e+003
```

MATLAB spits out the answer to our query conveniently named **ans**. This is a variable or symbolic name that can be used to represent the value later.

```
>> x=5*6
     x = 30
```

Command Window and Basic Arithmetic [2]

Once a variable has been entered into the system, we can refer to it later. Suppose that we want to compute a new quantity that we'll call y, which is equal to x multiplied by 3.56. Then we type

```
>> y = x * 3.56
     y = 106.8000
```

Command Window and Basic Arithmetic [3]

- To write the multiplication ab, in MATLAB we type ab, in MATLAB we type **a * b**

- For division, the quantity **a ÷ b** is typed as **a / b (forward slash).** This type of division is referred to as **right division (forward slash).**

- MATLAB also allows another way to enter division, called **left division (backslash).** We can enter the quantity b by a typing the slash mark used for division in the opposite way, that is, we use a back slash instead of a forward slash **a \ b**

Command Window and Basic Arithmetic [4]

Exponentiation **a to the power b** is entered in the following way **a ^ b**

Finally, **addition** and **subtraction** are entered in the usual way

a + b

a – b

Referencing individual entries

Individual matrix and vector entries can be referenced with indices inside parentheses. For example, A(2,3) denotes the entry in the second row, third column of matrix A.

```
A = [1 2 3 ; 4 5 6 ; -1 7 9]
A(2,3)
```

Create a column vector, x, with:

```
x = [3 2 1]'
```

or equivalently:

```
x = [3 ; 2 ; 1]
```

Two-dimensional arrays:

Format: A = [row1 ; row2 ; ... ; rowN] an array with N rows.

- Entry of arrays begins with a left square bracket, **[** , and ends with a right square bracket, **]**.

- Within the brackets rows of numbers are separated by semicolons, **;** , each row must have the same number of elements and elements of the **rows are separated** by **spaces** or **commas**.

Example 1

A=[2 3 4; -5 4 2; 4 -3 5]

A =

2	**3**	**4**
-5	**4**	**2**
4	**-3**	**5**

Example 2

A2= [1 3 4

-5 54 2

4 -3 15

2 2 -9]

A2 =

1	**3**	**4**
-5	**54**	**2**
4	**-3**	**15**
2	**2**	**-9**

Backslash (\) or Left Matrix Division

Used to **solve systems of linear equations Ax = b**, where A is a matrix and b is a vector.

It computes **x = A\b**, which is equivalent to solving **Ax = b**.

MATLAB internally uses an **optimized method** (Gaussian elimination or LU decomposition) to solve the system.

Example of Backslash (\) or Left Matrix Division

Solve the following system of linear equations

$2x + 3y = 1$

$4x + 5y = 2$

Matrix form:

$Ax = b$

Where

A = $\begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix}$, b = $\begin{bmatrix} 1 \\ 2 \end{bmatrix}$

```
A = [2 3; 4 5];

b = [1; 2];

x = A\b;                    % Solves Ax = b

disp(x);
```

Forward Slash (/) or Right Matrix Division

Used for solving $xA = b$ (instead of $Ax = b$).

Computes $x = b/A$, which solves the system in terms of rows.

Solve the following system of linear equations

$$2x + 3y = 1$$

$$4x + 5y = 2$$

Matrix form:

$$Ax = b$$

Where

$A = \begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix}$, $b = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$

```
A = [2 3; 4 5];

b = [1 2]; % Row vector

x = b/A; % Solves xA = b

disp(x);
```

Performance Note

- Using A\b is preferred over inv(A)*b because it's numerically more stable and computationally efficient.

- Directly computing inv(A) can introduce rounding errors

Display command

The display command: disp(variable).

- To **display *A*** , enter ***disp(A),*** to display ***A*** and ***A2*** together enter ***disp(A); disp(A2)***

Example 3

disp(A); disp(A2)

2	3	4
-5	4	2
4	-3	5

1	3	4
-5	54	2
4	-3	15
2	2	-9

Revision of array elements:

- **Revision of array elements: Assignment: change a single element.**

- Use row and column position numbers as an argument of the array variable name.

- Note: the use of the **colon (wild card)** in arrays, **:** , means to **use all elements of a specified row or column**.

- The notation, **A(:,:)** represents **all elements of array A** .

- To change the row 2 column 3 element of A from 2 to 17, type **A(2,3)=17**

Example 4

A(2,3) = 17

A =

2	**3**	**4**
-5	**4**	**17**
4	**-3**	**5**

Recall

A =

2	**3**	**4**
-5	**4**	**2**
4	**-3**	**5**

Example 5

Write a Matlab command to replace **row 3** of *A* by **[1 23 8]**

```
A(3,:) = [1 23 8]
```

A =

2	3	4
-5	4	17
1	23	8

Recall

A =

2	3	4
-5	4	17
4	-3	5

Example 6

Write a Matlab command to replace column 2 of A matrix by **[4 6 -14]’**

A(:,2)=[4 6 -14]’

A =

2	**4**	**4**
-5	**6**	**17**
1	**-14**	**8**

Recall

A =

2	**3**	**4**
-5	**4**	**17**
1	**23**	**8**

Operations in MATLAB

Symbol	Operation	Meaning
+/-	addition/subtraction	elementwise addition or subtraction
*	multiplication	standard matrix multiplication
.*	multiplication	elementwise multiplication, e.g., [4 3].*[2 -2] = [8 -6]
/	division	B/A = (A'\B')', solution of X*A=B
. /	Division	. / Division elementwise division, e.g., [4 3] . / [2 -2] = [2 -1.5]
\	None	A\B, efficient solution of A*X=B
^	exponentiation	matrix exponentiation, A^3=A*A*A
. ^	exponentiation	elementwise exponentiation, e.g., [4 3] . ^ 3 = [64 27]

Augmenting arrays

Symbol	Action (meaning)
;	Begin new row of array, separate MATLAB commands on a line
:	Wildcard for all rows or all columns, separator for limits
,	Separator for elements or array parts

MATLAB Column Vector Concatenation Example

If we have two row vectors:

```
B = [1 2];
C = [3 4] ;
```

We can concatenate them vertically using the MATLAB command:

```
D = [B ; C]
```

Output:

```
D =
    1    2
    3    4
```

MATLAB Row Vector Concatenation Example

If we have two row vectors:

```
B = [1 2];
C = [3 4];
```

We can concatenate them horizontally using the MATLAB command:

```
D = [B C]
```

Output:

```
D =
    1    2    3    4
```

Hilbert matrix

- **hilb** Hilbert matrix.

- **hilb(N)** is the N by N matrix with elements **1/(i+j-1).**

Format command

- ❑ **format Set output format.**
- ❑ format with no inputs sets the output format to the default appropriate for the class of the variable. For float variables, the default is format **SHORT**.

format SHORT	Scaled fixed point format with 5 digits.
format LONG	Scaled fixed point format with 15 digits for double and 7 digits for single.
format SHORTE	Floating point format with 5 digits.
format LONGE	Floating point format with 15 digits for double and 7 digits for single.
format SHORTG	Best of fixed or floating point format with 5 digits.
format LONGG	Best of fixed or floating point format with 15 digits for double and 7 digits for single.
format SHORTENG	Engineering format that has at least 5 digits and a power that is a multiple of three
format LONGENG	Engineering format that has exactly 16 significant digits and a power that is a multiple of three.

format RAT Approximation by ratio of small integers. Numbers with a large numerator or large denominator are replaced by *.

format short, pi

ans =

3.1416

format long, pi

ans =

3.141592653589793

```
format rat; D = hilb(6)
```

D =

Columns 1 through 3

1	1/2	1/3
1/2	1/3	1/4
1/3	1/4	1/5
1/4	1/5	1/6
1/5	1/6	1/7
1/6	1/7	1/8

Columns 4 through 6

1/4	1/5	1/6
1/5	1/6	1/7
1/6	1/7	1/8
1/7	1/8	1/9
1/8	1/9	1/10
1/9	1/10	1/11

What will be the output of the following MATLAB command?

D = hilb(6);

format rat; E = D(2:4,5:6);

It stores elements from rows **2** through **4** and in columns **5** through **6** of ***D*** into ***E***.

E =

1/6	**1/7**
1/7	**1/8**
1/8	**1/9**

What will be the output of the following MATLAB command?

D = hilb(6);

format rat;

F = D(:, 4:6)

% It stores elements from all rows and in columns 4 through 6 of *D* into *F*

F =

1/4	**1/5**	**1/6**
1/5	**1/6**	**1/7**
1/6	**1/7**	**1/8**
1/7	**1/8**	**1/9**
1/8	**1/9**	**1/10**
1/9	**1/10**	**1/11**

What will be the output of the following MATLAB command?

```
D = hilb(6);
format rat;
F = D(:, 4:6)
```

% It %stores the elements in rows 2 through 4, all columns, of *D* into *G*.

G =

Columns 1 through 3

1/2	**1/3**	**1/4**
1/3	**1/4**	**1/5**
1/4	**1/5**	**1/6**

Columns 4 through 6

1/5	**1/6**	**1/7**
1/6	**1/7**	**1/8**
1/7	**1/8**	**1/9**

PICKING OFF PARTS OF ARRAYS: HOW TO SELECT SUB-ARRAYS

For the **array *D***, the command **D(*row i* : *row j* , *column k* : *column l*)**, picks off subarrays of array *D*.

Column-Major Order in MATLAB

MATLAB uses **column-major order**, meaning that when storing or accessing elements in a **matrix**, it **fills and reads** elements **column by column**, rather than row by row.

A = [**1** 5 9 13;
2 **6** 10 14;
3 7 **11** 15;
4 8 12 **16**];

Column-major order means that the elements are stored in memory in this sequence:

1 → 2 → 3 → 4 → 5 → **6** → 7 → 8 → 9 → 10 → **11** → 12 → 13 → 14 → 15 → **16**

The **linear indices** in MATLAB follow this order:

A(1,1) → A(2,1) → A(3,1) → A(4,1) → A(1,2) → A(2,2) → A(3,2) → A(4,2) → ...

Comparison: Row-Major vs. Column-Major Order

Order Type	Storage & Access Pattern
Row-Major Order (C, C++)	Elements are stored row by row.
Column-Major Order (MATLAB, Fortran)	Elements are stored column by column.

In class quiz

Create a 8×8 Hilbert matrix in Matlab and store it in H and store its diagnol elements in D efficiently or using only colon operator.

The **diagonal elements** are at positions **(1,1), (2,2), (3,3), ..., (8,8).**

Their linear indices in an 8×8 column-major matrix are:
1, 10, 19, 28, 37, 46, 55, 64

The pattern follows **1:9:end**, where:

1 is the **first element**.
9 is the **step size**.
end ensures it stops at the **last diagonal element**.

```matlab
% Generate an 8×8 Hilbert matrix
H = hilb(8);

% Extract the diagonal elements using the colon operator
D = H(1:9:end);

% Display the results
disp('Hilbert Matrix H:');
disp(H);

disp('Diagonal Elements D:');
disp(D);
```