COMPUTATION PHYSICS

PHYS 4361
DR. Maher O. El-Ghossain
Associate Professor
Creating Arrays

The simplest array (one-dimensional) is a row or a column of numbers. A more complex array (two-dimensional) is a collection of numbers arranged in rows and columns. One use of arrays is to store information and data, as in a table. In science and engineering, one-dimensional arrays frequently represent vectors, and two-dimensional arrays often represent matrices:

**Creating a One-Dimensional Array (Vector):**

A one-dimensional array is a list of numbers arranged in a row or a column. The position of point $A$ can be expressed in terms of a position vector

$$\mathbf{r}_A = 2\mathbf{i} + 4\mathbf{j} + 5\mathbf{k}$$

where $\mathbf{i}$, $\mathbf{j}$, and $\mathbf{k}$ are unit vectors in the direction of the $x$, $y$, and $z$ axes, respectively.

The numbers 2, 4, and 5 can be used to define a row or a column vector.

Creating a vector from a known list of numbers:

The vector is created by typing the elements (numbers) inside square brackets $[\ ]$.

```python
variable_name = [type vector elements]
```
Creating a vector with constant spacing by specifying the first term, the spacing, and the last term: `variable_name = [m:q:n]` or `variable_name = m:q:n`

```matlab
>> pntAH=[2, 4, 5]
pntAH =
    2    4    5
>> pntAV=[2
4
5]
pntAV =
    2
    4
    5
```

The coordinates of point \(A\) are assigned to a row vector called `pntAH`.

The coordinates of point \(A\) are assigned to a column vector called `pntAV`. (The Enter key is pressed after each element is typed.)

```matlab
>> x=[1:2:13]
x =
    1    3    5    7    9   11   13
>> y=[1.5:0.1:2.1]
y =
    1.5000   1.6000   1.7000   1.8000   1.9000   2.0000   2.1000
>> z=[-3:7]
z =
    -3    -2    -1     0     1     2     3     4     5     6
```

First element 1, spacing 2, last element 13.

First element 1.5, spacing 0.1, last element 2.1.

First element \(-3\), last term 7. If spacing is omitted, the default is 1.

```matlab
>> xa=[21:-3:6]
xa =
    21    18    15    12     9     6
```

First element 21, spacing \(-3\), last term 6.
Creating a vector with linear (equal) spacing by specifying the first and last terms, and the number of terms:

ó A vector with \( n \) elements that are linearly (equally) spaced in which the first element is \( x_i \) and the last element is \( x_f \) can be created by typing the \texttt{linspace} command (MATLAB determines the correct spacing): \texttt{command variable name = linspace(xi,xf,n)}
CREATING A TWO-DIMENSIONAL ARRAY (MATRIX)

- A two-dimensional array, also called a matrix, has numbers in rows and columns.
- Matrices play an important role in linear algebra and are used in science and engineering to describe many physical quantities.
- In a square matrix the number of rows and the number of columns is equal. For example, the matrix

  \[
  \begin{pmatrix}
  7 & 4 & 9 \\
  3 & 8 & 1 \\
  6 & 5 & 3 \\
  \end{pmatrix}
  \]

  is a 3 × 3 matrix.
A \( m \times n \) matrix has \( m \) rows and \( n \) columns, and \( m \) By \( n \) is called the size of the matrix.

variable_name = [1st row elements; 2nd row elements; 3rd row elements; ... ; last row elements]

\[
\begin{bmatrix}
5 & 35 & 43 \\
4 & 76 & 81 \\
21 & 32 & 40
\end{bmatrix}
\]

\[
\begin{bmatrix}
7 & 2 & 76 & 33 & 8 \\
1 & 98 & 6 & 25 & 6 \\
5 & 54 & 68 & 9 & 0
\end{bmatrix}
\]

\[
\begin{bmatrix}
3.0000 & 24.0000 & 0.5000 \\
16.0000 & 1.6330 & 14.0000
\end{bmatrix}
\]
The zeros, ones and, eye Commands

- The zeros(m,n), ones(m,n), and eye(n) commands can be used to create matrices with m rows and n columns in which all elements are the numbers 0 and 1 respectively. and eye(n) create square matrix with n rows and n columns in which the diagonal elements are equal to 1 and the rest of the elements are 0.

```
>> zr=zeros(3,4)
zr =
     0     0     0     0
     0     0     0     0
     0     0     0     0
>> ne =
ne =
     1     1     1     1
     1     1     1     1
     1     1     1     1
     1     1     1
>> idn=eye(5)
idn =
idn =
     1     0     0     0     0
     0     1     0     0     0
     0     0     1     0     0
     0     0     1     0     0
     0     0     0     1     1
```
NOTES ABOUT VARIABLES IN MATLAB

• All variables in MATLAB are arrays. A scalar is an array with one element, a vector is an array with one row or one column of elements, and a matrix is an array with elements in rows and columns.

• The variable (scalar, vector, or matrix) is defined by the input when the variable is assigned. There is no need to define the size of the array (single element for a scalar, a row or a column of elements for a vector, or a two-dimensional array of elements for a matrix) before the elements are assigned.

• Once a variable exists—as a scalar, vector, or matrix—it can be changed to any other size, or type, of variable. For example, a scalar can be changed to a vector or a matrix; a vector can be changed to a scalar, a vector of different length, or a matrix; and a matrix can be changed to have a different size, or be reduced to a vector or a scalar.
THE TRANSPOSE OPERATOR

The transpose operator, when applied to a vector, switches a row (column) vector to a column (row) vector.

\[
\begin{align*}
&>> \text{aa} = [3 \ 8 \ 1] \\
&\text{aa} = \\
&\begin{array}{c}
3 \\
8 \\
1
\end{array} \\
&>> \text{bb} = \text{aa}' \\
&\text{bb} = \\
&\begin{array}{c}
3 \\
8 \\
1
\end{array} \\
&>> \text{c} = [2 \ 55 \ 14 \ 8; \ 21 \ 5 \ 32 \ 11; \ 41 \ 64 \ 9 \ 1] \\
&\text{c} = \\
&\begin{array}{cccc}
2 & 55 & 14 & 8 \\
21 & 5 & 32 & 11 \\
41 & 64 & 9 & 1
\end{array} \\
&>> \text{d} = \text{c}' \\
&\text{d} = \\
&\begin{array}{ccc}
2 & 21 & 41 \\
55 & 5 & 64 \\
14 & 32 & 9 \\
8 & 11 & 1
\end{array}
\end{align*}
\]
ARRAY ADDRESSING

- Elements in an array (either vector or matrix) can be addressed individually or in subgroups.

- **Vector**
  - The address of an element in a vector is its position in the row (or column). For a vector named ve, ve(k) refers to the element in position k.
  - Ve = 35 46 78 23 5 14 81 3 55
  - then
  - ve(4) = 23, ve(7) = 81, and ve(1) = 35.
% Define a vector.
VCT = [35 46 78 23 5 14 81 3 55]

% Display the fourth element.
VCT(4)

ans =
    23

% Assign a new value to the sixth element.
VCT(6) = 273
VCT =
    35  46  78  23  5  273  81  3  55

% The whole vector is displayed.

% Use the vector elements in mathematical expressions.
VCT(2) + VCT(8)
ans =
    49

VCT(5) * VCT(8) + sqrt(VCT(7))
ans =
    134

>>
Matrix

- The address of an element in a matrix is its position, defined by the row number and the column number where it is located. For a matrix assigned to a variable $ma$, $ma(k,p)$ refers to the element in row $k$ and column $p$.

- $Ma = \begin{bmatrix}
3 & 1 & 6 & 5 \\
4 & 7 & 10 & 2 \\
13 & 9 & 0 & 8
\end{bmatrix}$

For example, if the matrix is: then $ma(1,1) = 3$ and $ma(2,3) = 10$. 
MAT = [3 11 6 5; 4 7 10 2; 13 9 0 8]  
Create a $3 \times 4$ matrix.

MAT =
   3   11    6    5
   4    7   10    2
  13    9    0    8

MAT(3,1) = 20
Assign a new value to the (3,1) element.

MAT =
   3   11    6    5
   4    7   10    2
  20    9    0    8

MAT(2,4) - MAT(1,2)
Use elements in a mathematical expression.

ans =
   -9
USING A COLON : IN ADDRESSING ARRAYS
A colon can be used to address a range of elements in a vector or a matrix.

For a vector:
- $va(:)$ Refers to all the elements of the vector $va$ (either a row or a column vector).
- $va(m:n)$ Refers to elements $m$ through $n$ of the vector $va$. example

```matlab
>> v=[4 15 8 12 34 2 50 23 11];
A vector v is created.
v =
  4   15   8  12  34   2  50  23  11
>> u=v(3:7)
A vector u is created from the elements 3 through 7 of vector v.
u =
  8  12  34   2  50
```

- For a matrix:
- $A(:,n)$ Refers to the elements in all the rows of column $n$ of the matrix $A$.
- $A(n,:)$ Refers to the elements in all the columns of row $n$ of the matrix $A$.
- $A(:,m:n)$ Refers to the elements in all the rows between columns $m$ and $n$ of the matrix $A$.
- $A(m:n,:)$ Refers to the elements in all the columns between rows $m$ and $n$ of the matrix $A$.
- $A(m:n,p:q)$ Refers to the elements in rows $m$ through $n$ and columns $p$ through $q$ of the matrix $A$. 
Define a matrix \( A \) with 5 rows and 6 columns.

\[
A = \begin{bmatrix}
1 & 3 & 5 & 7 & 9 & 11 \\
2 & 4 & 6 & 8 & 10 & 12 \\
3 & 6 & 9 & 12 & 15 & 18 \\
4 & 8 & 12 & 16 & 20 & 24 \\
5 & 10 & 15 & 20 & 25 & 30
\end{bmatrix}
\]

Define a column vector \( B \) from the elements in all of the rows of column 3 in matrix \( A \).

\[
B = \begin{bmatrix}
5 \\
6 \\
9 \\
12 \\
15
\end{bmatrix}
\]

Define a row vector \( C \) from the elements in all of the columns of row 2 in matrix \( A \).

\[
C = \begin{bmatrix}
2 & 4 & 6 & 8 & 10 & 12
\end{bmatrix}
\]

Define a matrix \( E \) from the elements in rows 2 through 4 and all the columns in matrix \( A \).

\[
E = \begin{bmatrix}
2 & 4 & 6 & 8 & 10 & 12 \\
3 & 6 & 9 & 12 & 15 & 18 \\
4 & 8 & 12 & 16 & 20 & 24
\end{bmatrix}
\]

Create a matrix \( F \) from the elements in rows 1 through 3 and columns 2 through 4 in matrix \( A \).

\[
F = \begin{bmatrix}
3 & 5 & 7 \\
4 & 6 & 8 \\
6 & 9 & 12
\end{bmatrix}
\]
ADDING ELEMENTS TO EXISTING VARIABLES

Adding elements to a vector:

```matlab
>> v=4:3:34
Create a vector v with 11 elements.
v =
   4   7  10  13  16  19  22  25  28  31  34
>> u=v([3, 5, 7:10])
Create a vector u from the 3rd, the 5th, and the 7th through 10th elements of v.
u =
   10  16  22  25  28  31
>> A=[10:-1:4; ones(1,7); 2:2:14; zeros(1,7)]
Create a 4×7
A =
   10   9   8   7   6   5   4
   1   1   1   1   1   1   1
   2   4   6   8  10  12  14
   0   0   0   0   0   0   0
>> B = A([1,3], [1,3,5:7])
Create a matrix B from the 1st and 3rd rows, and 1st, 3rd, and the 5th through 7th columns of A
B =
   10   8   6   5   4
   2   6  10  12  14```
DF = 1:4
Define vector DF with 4 elements.

DF =
1 2 3 4

DF(5:10) = 10:5:35
Adding 6 elements starting with the 5th.

DF =
1 2 3 4 10 15 20 25 30 35

AD = [5 7 2]
Define vector AD with 3 elements.

AD =
5 7 2

AD(8) = 4
Assign a value to the 8th element.

AD =
5 7 2 0 0 0 0 4
MATLAB assigns zeros to the 4th through 7th elements.

AR(5) = 24
Assign a value to the 5th element of a new vector.

AR =
0 0 0 0 0 0 24
MATLAB assigns zeros to the 1st through 4th elements.

RE = [3 8 1 24];
Define vector RE with 4 elements.

GT = 4:3:15;

KNH = [RE GT]
Define a new vector KNH by appending RE and GT

KNH =
3 8 1 24 4 7 10 13 16

KNV = [RE'; GT']
Create a new column vector KNV by appending RE' and GT'

KNV =
3
0
1
24
4
7
10
13
16
Adding elements to a matrix:
Rows and/or columns can be added to an existing matrix by assigning values to the new rows or columns.

```
>> E=[1 2 3 4; 5 6 7 8]
E =
  1  2  3  4
  5  6  7  8

>> E(3,:)=[10 14 18 22]
E =
  1  2  3  4
  5  6  7  8
 10 14 18 22

>> K=eye(3)
K =
  1  0  0
  0  1  0
  0  0  1

>> G=[E K]
G =
  1  2  3  4  1  0  0
  5  6  7  8  0  1  0
 10 14 18 22 0  0  1
```
Define a $2 \times 3$ matrix. 
\[ \mathbf{A} = \begin{bmatrix} 3 & 6 & 9 \\ 8 & 5 & 11 \end{bmatrix} \]

Assign a value to the $(4,5)$ element. 
\[ \mathbf{A}(4,5) = 17 \]
\[ \mathbf{A} = \begin{bmatrix} 3 & 6 & 9 & 0 & 0 \\ 8 & 5 & 11 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 17 \end{bmatrix} \]

MATLAB changes the matrix size to $4 \times 5$, and assigns zeros to the new elements.

Assign a value to the $(3,4)$ element of a new matrix. 
\[ \mathbf{B}(3,4) = 15 \]
\[ \mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 15 \end{bmatrix} \]

MATLAB creates a $3 \times 4$ matrix and assigns zeros to all the elements except $\mathbf{B}(3,4)$. 
DELETING ELEMENTS

An element, or a range of elements, of an existing variable can be deleted by reassigning nothing to these elements.

```matlab
>> kt=[2 8 40 65 3 55 23 15 75 80]
k =
    2 8 40 65 3 55 23 15 75 80
>> kt(6)=[ ]
k =
    2 8 40 65 3 23 15 75 80
```

Eliminate the 6th element.

```matlab
>> kt(3:6)=[ ]
k =
    2 8 15 75 80
```

Eliminate elements 3 through 6.

```
>> mtr=[5 78 4 24 9; 4 0 36 60 12; 56 13 5 89 3]
mtr =
```

The vector now has 9 elements.

Define a 3 × 5 matrix.

```
```

Define a vector with 10 elements.

Eliminate all the rows of columns 2 through 4.
variable_name = char('string 1','string 2','string 3')

For example:

>> Info = char('Student Name:','John Smith','Grade:','A+')

Info =
Student Name: John Smith
Grade: A+

A variable named Info is assigned four rows of strings, each with different length.

The function char creates an array with four rows with the same length as the longest row by adding empty spaces to the shorter lines.
MATLAB has many built-in functions for managing and handling arrays. Some of these are listed below:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>length(A)</code></td>
<td>Returns the number of elements in the vector A.</td>
<td><code>&gt;&gt; A=[5 9 2 4];</code>&lt;br&gt;<code>&gt;&gt; length(A)</code>&lt;br&gt;<code>ans = 4</code></td>
</tr>
<tr>
<td><code>size(A)</code></td>
<td>Returns a row vector ([m, n]), where (m) and (n) are the size (m \times n) of the array A.</td>
<td><code>&gt;&gt; A=[6 1 4 0 12; 5 19 6 8 2]</code>&lt;br&gt;<code>A =</code>&lt;br&gt;<code>6 1 4 0 12</code>&lt;br&gt;<code>5 19 6 8 2</code>&lt;br&gt;<code>&gt;&gt; size(A)</code>&lt;br&gt;<code>ans = 2 5</code></td>
</tr>
<tr>
<td><code>reshape(A, m, n)</code></td>
<td>Creates a (m) by (n) matrix from the elements of matrix A. The elements are taken column after column. Matrix A must have (m) times (n) elements.</td>
<td><code>&gt;&gt; A=[5 1 6; 8 0 2]</code>&lt;br&gt;<code>A =</code>&lt;br&gt;<code>5 1 6</code>&lt;br&gt;<code>8 0 2</code>&lt;br&gt;<code>&gt;&gt; B = reshape(A, 3, 2)</code>&lt;br&gt;<code>B =</code>&lt;br&gt;<code>5 0</code>&lt;br&gt;<code>0 6</code>&lt;br&gt;<code>1 2</code></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>-----------</td>
<td>------------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
</tbody>
</table>
| `diag(v)` | When \( v \) is a vector, creates a square matrix with the elements of \( v \) in the diagonal | \[
\text{>> } v=[7 \ 4 \ 2];
\]
\[
\text{>> } A=\text{diag}(v)
\]
\[
A =
\begin{bmatrix}
7 & 0 & 0 \\
0 & 4 & 0 \\
0 & 0 & 2 \\
\end{bmatrix}
\] |
| `diag(A)` | When \( A \) is a matrix, creates a vector from the diagonal elements of \( A \). | \[
\text{>> } A=[1 \ 2 \ 3; \ 4 \ 5 \ 6; \ 7 \ 8 \ 9]
\]
\[
A =
\begin{bmatrix}
1 & 2 & 3 \\
4 & 5 & 6 \\
7 & 8 & 9 \\
\end{bmatrix}
\]
\[
\text{>> vec=diag(A)}
\]
\[
vec =
\begin{bmatrix}
1 \\
5 \\
9 \\
\end{bmatrix}
\] |
Example 2-1: Create a matrix

Using the ones and zeros commands, create a 4 x 5 matrix in which the first two rows are 0s and the next two rows are 1s.

Solution

```
>> A(1:2,:)=zeros(2,5)
A =
     0     0     0     0     0
     0     0     0     0     0

>> A(3:4,:)=ones(2,5)
A =
     0     0     0     0     0
     0     0     0     0     0
     1     1     1     1     1
     1     1     1     1     1
```

First, create a 2 x 5 matrix with 0s.

Add rows 3 and 4 with 1s.

Create a 4 x 5 matrix from two 2 x 5 matrices.
Example 2-2: Create a matrix
Create a 6 x 6 matrix in which the middle two rows and the middle two columns are 1s, and the rest of the entries are 0s. Solution

```
>> AR=zeros(6,6)
AR =
0   0   0   0   0   0
0   0   0   0   0   0
0   0   0   0   0   0
0   0   0   0   0   0
0   0   0   0   0   0
0   0   0   0   0   0

>> AR(3:4,:)=ones(2,6)
AR =
0   0   0   0   0   0
0   0   0   0   0   0
1   1   1   1   1   1
1   1   1   1   1   1
0   0   0   0   0   0
0   0   0   0   0   0

>> AR(:,3:4)=ones(6,2)
AR =
0   0   1   1   0   0
0   0   1   1   0   0
1   1   1   1   1   1
1   1   1   1   1   1
0   0   1   1   0   0
0   0   1   1   0   0
```

First, create a 6 × 6 matrix with 0s.
Reassign the number 1 to the 3rd and 4th rows.
Reassign the number 1 to the 3rd and 4th columns.
EXAMPLE 2-3: Matrix manipulation
Given are a 5x6 matrix A, a 3x6 matrix B, and a 9-element vector v.
Create the three arrays in the Command Window, and then, by writing one command, replace the last four columns of the first and third rows of A with the first four columns of the first two rows of B, the last four columns of the fourth row of A with the elements 5 through 8 of v, and the last four columns of the fifth row of A with columns 3 through 5 of the third row of B.
Solution
A=
  2 5 8 11 14 17
  3 6 9 12 15 18
  4 7 10 13 16 19
  5 8 11 14 17 20
  6 9 12 15 18 21
B=
  5 10 15 20 25 30
  30 35 40 45 50 55
  55 60 65 70 75 80
v = 99 98 97 96 95 94 93 92 91
\[ A = \begin{bmatrix} 2 & 5 & 8 & 11 & 14 & 17 \\ 3 & 6 & 9 & 12 & 15 & 18 \\ 4 & 7 & 10 & 13 & 16 & 19 \\ 5 & 8 & 11 & 14 & 17 & 20 \\ 6 & 9 & 12 & 15 & 18 & 21 \end{bmatrix} \]

\[ B = \begin{bmatrix} 5 & 10 & 15 & 20 & 25 & 30 \\ 30 & 35 & 40 & 45 & 50 & 55 \\ 55 & 60 & 65 & 70 & 75 & 80 \end{bmatrix} \]

\[ v = [99; -1; 91] \]

\[ A([1 \ 3 \ 4 \ 5], [3 \ 6]) = [B([1 \ 2], [1 \ 4]); v([5 \ 8]); B([3 \ 2 \ 5])] \]

4 × 4 matrix made of columns 3 through 6 of rows 1, 3, 4, and 5.

4 × 4 matrix. The first two rows are columns 1 through 4 of rows 1 and 2 of matrix B. The third row consists of elements 5 through 8 of vector v. The fourth row consists of columns 2 through 5 of row 3 of matrix B.
A =

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>5</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>9</td>
<td>12</td>
<td>15</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>30</td>
<td>35</td>
<td>40</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>95</td>
<td>94</td>
<td>93</td>
<td>92</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>60</td>
<td>65</td>
<td>70</td>
<td>75</td>
<td></td>
</tr>
</tbody>
</table>

variable_name = char('string 1', 'string 2', 'string 3')

For example:

```
>> Info = char('Student Name:', 'John Smith', 'Grade:', 'A+')
Info =
Student Name: John Smith
Grade: A+
```

A variable named Info is assigned four rows of strings, each with different length.

The function char creates an array with four rows with the same length as the longest row by adding empty spaces to the shorter lines.
Mathematical Operations with Arrays

Addition and Subtraction

In general, if \( A \) and \( B \) are two arrays (for example, \( 2 \times 3 \) matrices),

\[
A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \end{bmatrix}
\]

then the matrix that is obtained by adding \( A \) and \( B \) is:

\[
\begin{bmatrix}
(A_{11} + B_{11}) & (A_{12} + B_{12}) & (A_{13} + B_{13}) \\
(A_{21} + B_{21}) & (A_{22} + B_{22}) & (A_{23} + B_{23})
\end{bmatrix}
\]
Examples are:

```matlab
>> VectA=[8 5 4]; VectB=[10 2 7];
Define two vectors.
>> VectC=VectA+VectB
Define a vector VectC that is equal to VectA + VectB.
VectC =
    18    7   11

>> A=[5 -3 8; 9 2 10]
Define two 2 x 3
A =
      5     3     8
     9     2    10

>> B=[10 7 4; -11 15 1]
Subtracting matrix B from matrix A.
B =
         10     7     4
        -11    15     1

>> A-B
ans =
     -5   -10     4
      20   -13     9

>> C=A+B
Define a matrix C that is equal to A + B.
C =
      15     4    12
     -2    17    11
```
**Array Multiplication**

\[
A = \begin{bmatrix}
A_{11} & A_{12} & A_{13} \\
A_{21} & A_{22} & A_{23} \\
A_{31} & A_{32} & A_{33} \\
A_{41} & A_{42} & A_{43}
\end{bmatrix}
\quad \text{and} \quad
B = \begin{bmatrix}
B_{11} & B_{12} \\
B_{21} & B_{22} \\
B_{31} & B_{32}
\end{bmatrix}
\]

then the matrix that is obtained with the operation \(A*B\) has dimensions \(4 \times 2\) with the elements:

\[
\begin{bmatrix}
(A_{11}B_{11} + A_{12}B_{21} + A_{13}B_{31}) & (A_{11}B_{12} + A_{12}B_{22} + A_{13}B_{32}) \\
(A_{21}B_{11} + A_{22}B_{21} + A_{23}B_{31}) & (A_{21}B_{12} + A_{22}B_{22} + A_{23}B_{32}) \\
(A_{31}B_{11} + A_{32}B_{21} + A_{33}B_{31}) & (A_{31}B_{12} + A_{32}B_{22} + A_{33}B_{32}) \\
(A_{41}B_{11} + A_{42}B_{21} + A_{43}B_{31}) & (A_{41}B_{12} + A_{42}B_{22} + A_{43}B_{32})
\end{bmatrix}
\]

A numerical example is:

\[
\begin{bmatrix}
1 & 4 & 3 \\
2 & 6 & 1 \\
5 & 2 & 8
\end{bmatrix}
\begin{bmatrix}
5 & 4 \\
1 & 3 \\
2 & 6
\end{bmatrix}
= \begin{bmatrix}
(1 \cdot 5 + 4 \cdot 1 + 3 \cdot 2) & (1 \cdot 4 + 4 \cdot 3 + 3 \cdot 6) \\
(2 \cdot 5 + 6 \cdot 1 + 1 \cdot 2) & (2 \cdot 4 + 6 \cdot 3 + 1 \cdot 6) \\
(5 \cdot 5 + 2 \cdot 1 + 8 \cdot 2) & (5 \cdot 4 + 2 \cdot 3 + 8 \cdot 6)
\end{bmatrix}
= \begin{bmatrix}
15 & 34 \\
18 & 32 \\
43 & 74
\end{bmatrix}
\]
Define a $4 \times 3$ matrix $A$:

$$A = \begin{bmatrix} 1 & 4 & 2 \\ 5 & 7 & 3 \\ 9 & 1 & 6 \\ 4 & 2 & 8 \end{bmatrix}$$

Define a $3 \times 2$ matrix $B$:

$$B = \begin{bmatrix} 6 & 1 \\ 2 & 5 \\ 7 & 3 \end{bmatrix}$$

Multiply matrix $A$ by matrix $B$ and assign the result to variable $C$.

$$C = A \times B = \begin{bmatrix} 28 & 27 \\ 65 & 49 \\ 98 & 32 \\ 84 & 38 \end{bmatrix}$$
\[ A_{11}x_1 + A_{12}x_2 + A_{13}x_3 = B_1 \]
\[ A_{21}x_1 + A_{22}x_2 + A_{23}x_3 = B_2 \]
\[ A_{31}x_1 + A_{32}x_2 + A_{33}x_3 = B_3 \]

can be written in a matrix form as

\[
\begin{bmatrix}
A_{11} & A_{12} & A_{13} \\
A_{21} & A_{22} & A_{23} \\
A_{31} & A_{32} & A_{33}
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} =
\begin{bmatrix}
B_1 \\
B_2 \\
B_3
\end{bmatrix}
\]

and in matrix notation as

\[ AX = B \quad \text{where} \quad A = \begin{bmatrix}
A_{11} & A_{12} & A_{13} \\
A_{21} & A_{22} & A_{23} \\
A_{31} & A_{32} & A_{33}
\end{bmatrix}, \quad X = \begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix}
B_1 \\
B_2 \\
B_3
\end{bmatrix}. \]
**Array Division**

Identity matrix:

\[
\begin{bmatrix}
7 & 3 & 8 \\
4 & 1 & 5 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
= 
\begin{bmatrix}
7 & 3 & 8 \\
4 & 1 & 5 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
= 
\begin{bmatrix}
7 & 4 & 5 \\
0 & 1 & 0 \\
1 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
6 & 2 & 9 \\
1 & 8 & 3 \\
0 & 0 & 1
\end{bmatrix}
= 
\begin{bmatrix}
6 & 2 & 9 \\
1 & 8 & 3 \\
7 & 4 & 5
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
= 
\begin{bmatrix}
6 & 2 & 9 \\
1 & 8 & 3 \\
7 & 4 & 5
\end{bmatrix}
\]

If a matrix \( A \) is square, it can be multiplied by the identity matrix, \( I \), from the left or from the right:

\( A I = I A = A \)
**Inverse of a matrix:**

The matrix $B$ is the inverse of the matrix $A$ if, when the two matrices are multiplied, the product is the identity matrix. Both matrices must be square and the multiplication order can be $BA$ or $AB$.

$$BA = AB = I$$

Obviously $B$ is the inverse of $A$, and $A$ is the inverse of $B$. For example:

$$\begin{bmatrix} 2 & 1 & 4 \\ 4 & 1 & 8 \\ 2 & -1 & 3 \end{bmatrix} \begin{bmatrix} 5.5 & -3.5 & 2 \\ 2 & -1 & 0 \\ -3 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 5.5 & -3.5 & 2 \\ 2 & -1 & 0 \\ -3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 4 \\ 4 & 1 & 8 \\ 2 & -1 & 3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$
Creating the matrix A.

\[
A = \begin{bmatrix}
2 & 1 & 4 \\
4 & 1 & 8 \\
2 & -1 & 3 \\
\end{bmatrix}
\]

Use the inv function to find the inverse of A and assign it to B.

\[
B = \begin{bmatrix}
5.5000 & -3.5000 & 2.0000 \\
2.0000 & -1.0000 & 0 \\
-3.0000 & 2.0000 & -1.0000 \\
\end{bmatrix}
\]

Multiplication of A and B gives the identity matrix.

\[
A \times B = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
\end{bmatrix}
\]

Use the power \(-1\) to find the inverse of A. Multiplying it by A gives the identity matrix.

\[
\begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
\end{bmatrix}
\]
Determinants:

A determinant is a function associated with square matrices.

\[ |A| = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21}, \text{ for example, } \begin{vmatrix} 6 & 5 \\ 3 & 9 \end{vmatrix} = 6 \cdot 9 - 5 \cdot 3 = 39 \]
Array division:
MATLAB has two types of array division, right division and left division.

**Left division, \( \backslash \):**

Left division is used to solve the matrix equation \( AX = B \). In this equation \( X \) and \( B \) are column vectors. This equation can be solved by multiplying, on the left, both sides by the inverse of \( A \):

\[
A^{-1}AX = A^{-1}B
\]

The left-hand side of this equation is \( X \) since

\[
A^{-1}AX = IX = X
\]

So the solution of \( AX = B \) is:

\[
X = A^{-1}B
\]

In MATLAB the last equation can be written by using the left division character:

\[
X = A \backslash B
\]
Right division, /:
The right division is used to solve the matrix equation $XC = D$. In this equation $X$ and $D$ are row vectors. This equation can be solved by multiplying, on the right, both sides by the inverse of $C$:

$$X \cdot CC^{-1} = D \cdot C^{-1}$$

which gives

$$X = D \cdot C^{-1}$$

In MATLAB the last equation can be written using the right division character:

$$X = D/C$$
Use matrix operations to solve the following system of linear equations.

\[
4x - 2y + 6z = 8 \\
2x + 8y + 2z = 4 \\
6x + 10y + 3z = 0
\]

Solution

Using the rules of linear algebra demonstrated earlier, the above system of equations can be written in the matrix form \( AX = B \) or in the form \( XC = D \):

\[
\begin{bmatrix}
4 & -2 & 6 \\
2 & 8 & 2 \\
6 & 10 & 3
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix} =
\begin{bmatrix}
8 \\
4 \\
0
\end{bmatrix}
\]

or

\[
\begin{bmatrix}
x & y & z
\end{bmatrix}
\begin{bmatrix}
4 & 2 & 6 \\
-2 & 8 & 10 \\
6 & 2 & 3
\end{bmatrix} =
\begin{bmatrix}
8 & 4 & 0
\end{bmatrix}
\]

Solutions for both forms are shown below.
% Solving the form \( AX = B \).

\[
\begin{bmatrix}
4 & -2 & 6 \\
2 & 8 & 2 \\
6 & 10 & 3
\end{bmatrix}
\begin{bmatrix}
8 \\
4 \\
0
\end{bmatrix}
\]

\[
X = A \backslash B
\]

\[
\begin{bmatrix}
-1.8049 \\
0.2927 \\
2.6341
\end{bmatrix}
\]

% Solving by using left division: \( X = A \backslash B \).

\[
\begin{bmatrix}
4 & 2 & 6 \\
-2 & 8 & 10 \\
6 & 2 & 3
\end{bmatrix}
\begin{bmatrix}
8 \\
4 \\
0
\end{bmatrix}
\]

\[
X = A^{-1}B
\]

\[
\begin{bmatrix}
-1.8049 \\
0.2927 \\
2.6341
\end{bmatrix}
\]

% Solving by using the inverse of \( A \): \( X = A^{-1}B \).

% Solving the form \( XC = D \).

\[
\begin{bmatrix}
4 & 2 & 6 \\
-2 & 8 & 10 \\
6 & 2 & 3
\end{bmatrix}
\begin{bmatrix}
9 \\
4 \\
0
\end{bmatrix}
\]

\[
X = D \div C
\]

\[
\begin{bmatrix}
-1.8049 & 0.2927 & 2.6341
\end{bmatrix}
\]

% Solving by using right division: \( X = D \div C \).

\[
\begin{bmatrix}
4 & 2 & 6 \\
-2 & 8 & 10 \\
6 & 2 & 3
\end{bmatrix}
\]

\[
X = \text{inv}(C)
\]

\[
\begin{bmatrix}
-1.8049 & 0.2927 & 2.6341
\end{bmatrix}
\]

% Solving by using the inverse of \( C \): \( X = D \cdot C^{-1} \).
ELEMENT-BY-ELEMENT OPERATIONS

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.*</td>
<td>Multiplication</td>
<td>./</td>
<td>Right division</td>
</tr>
<tr>
<td>^</td>
<td>Exponentiation</td>
<td>\</td>
<td>Left Division</td>
</tr>
</tbody>
</table>

Define a $2 \times 3$

Define a $2 \times 3$

Element-by-element multiplication of array A by B.

Element-by-element division of array A by B. The result is assigned to variable C.
Element-by-element exponentiation of array B. The result is an array in which each term is the corresponding term in B raised to the power of 3.

Trying to multiply A*B gives an error since A and B cannot be multiplied according to linear algebra rules. (The number of columns in A is not equal to the number of rows in B.)

Create a vector x with eight elements.

Vector x is used in element-by-element calculations of the elements of vector y.
USING ARRAYS IN MATLAB BUILT-IN MATH FUNCTIONS

```matlab
>> x=[0:pi/6:pi]
x =
   0  0.5236  1.0472  1.5708  2.0944  2.6180  3.1416
>> y=cos(x)
y =
   1.0000  0.8660  0.5000  0.0000 -0.5000 -0.8660 -1.0000
>>
>> d=[1 4 9; 16 25 36; 49 64 81] Creating a 3 x 3 array.
d =
   1  4  9
   16  25  36
   49  64  81
>> h=sqrt(d)
h =
   1  2  3
   4  5  6
   7  8  9
```

h is a 3 x 3 array in which each element is the square root of the corresponding element in array d.
## Built-in Functions for Analyzing Arrays

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mean(A)</code></td>
<td>If A is a vector, returns the mean value of the elements of the vector.</td>
<td><code>&gt;&gt; A=[5 9 2 4];</code>&lt;br&gt;<code>&gt;&gt; mean(A)</code>&lt;br&gt;<code>ans =</code>&lt;br&gt;<code>5</code></td>
</tr>
<tr>
<td><code>C=max(A)</code></td>
<td>If A is a vector, C is the largest element in A. If A is a matrix, C is a row vector containing the largest element of each column of A.</td>
<td><code>&gt;&gt; A=[5 9 2 4 11 6 11 1];</code>&lt;br&gt;<code>&gt;&gt; C=max(A)</code>&lt;br&gt;<code>C =</code>&lt;br&gt;<code>11</code></td>
</tr>
<tr>
<td><code>[d,n]=max(A)</code></td>
<td>If A is a vector, d is the largest element in A, and n is the position of the element (the first if several have the max value).</td>
<td><code>&gt;&gt; [d,n]=max(A)</code>&lt;br&gt;<code>d =</code>&lt;br&gt;<code>11</code>&lt;br&gt;<code>n =</code>&lt;br&gt;<code>5</code></td>
</tr>
<tr>
<td><code>min(A)</code></td>
<td>The same as <code>max(A)</code>, but for the smallest element.</td>
<td><code>&gt;&gt; A=[5 9 2 4];</code>&lt;br&gt;<code>&gt;&gt; min(A)</code>&lt;br&gt;<code>ans =</code>&lt;br&gt;<code>2</code></td>
</tr>
<tr>
<td><code>[d,n]=min(A)</code></td>
<td>The same as <code>[d,n]=max(A)</code>, but for the smallest element.</td>
<td></td>
</tr>
<tr>
<td><code>sum(A)</code></td>
<td>If A is a vector, returns the sum of the elements of the vector.</td>
<td><code>&gt;&gt; A=[5 9 2 4];</code>&lt;br&gt;<code>&gt;&gt; sum(A)</code>&lt;br&gt;<code>ans =</code>&lt;br&gt;<code>20</code></td>
</tr>
<tr>
<td><code>sort(A)</code></td>
<td>If A is a vector, arranges the elements of the vector in ascending order.</td>
<td><code>&gt;&gt; A=[5 9 2 4];</code>&lt;br&gt;<code>&gt;&gt; sort(A)</code>&lt;br&gt;<code>ans =</code>&lt;br&gt;<code>2 4 5 9</code></td>
</tr>
<tr>
<td><code>median(A)</code></td>
<td>If A is a vector, returns the median value of the elements of the vector.</td>
<td><code>&gt;&gt; A=[5 9 2 4];</code>&lt;br&gt;<code>&gt;&gt; median(A)</code>&lt;br&gt;<code>ans =</code>&lt;br&gt;<code>4.5000</code></td>
</tr>
</tbody>
</table>
### Generation of Random Numbers

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
</table>
| `rand`     | Generates a single random number between 0 and 1.                            | `>> rand  
an =  
0.2311` |
| `rand(1,n)`| Generates an n-element row vector of random numbers between 0 and 1.         | `>> a=rand(1,4)  
a =  
0.6068  0.4860  0.8913  0.7621` |
| `rand(n)`  | Generates an $n \times n$ matrix with random numbers between 0 and 1.       | `>> b=rand(3)  
b =  
0.4565  0.4447  0.9218  
0.0185  0.6154  0.7382  
0.8214  0.7919  0.1763` |
| `rand(m,n)`| Generates an $m \times n$ matrix with random numbers between 0 and 1.       | `>> c=rand(2,4)  
c =  
0.4057  0.9169  0.8936  0.3529  
0.9355  0.4103  0.0579  0.8132` |
| `randperm(n)` | Generates a row vector with $n$ elements that are random permutation of integers 1 through $n$. | `>> randperm(8)  
an =  
8 2 7 4 3 6 5 1` |
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
</table>
| `std(A)` | If A is a vector, returns the standard deviation of the elements of the vector. | >> A=[5 9 2 4];  
>> std(A)  
ans =  
2.9439 |
| `det(A)` | Returns the determinant of a square matrix A. | >> A=[2 4; 3 5];  
>> det(A)  
ans =  
-2 |
| `dot(a,b)` | Calculates the scalar (dot) product of two vectors a and b. The vectors can each be row or column vectors. | >> a=[1 2 3];  
>> b=[3 4 5];  
>> dot(a,b)  
ans =  
26 |
| `cross(a,b)` | Calculates the cross product of two vectors a and b, (axb). The two vectors must have each three elements. | >> a=[1 3 2];  
>> b=[2 4 1];  
>> cross(a,b)  
ans =  
-5 3 -2 |
| `inv(A)` | Returns the inverse of a square matrix A. | >> A=[2 -2 1; 3 2 -1; 2 -3 2];  
>> inv(A)  
ans =  
0.2000 0.2000 0  
-1.6000 0.4000 1.0000  
-2.6000 0.4000 2.0000 |
For example, a vector of 10 elements with random values between $-5$ and 10 can be created by $(a = -5, b = 10)$:

\[
(b - a) \times \text{rand} + a
\]

```matlab
>> v = 15*rand(1,10) - 5
```

\[
v = \\
\begin{array}{cccccc}
-1.8640 & 0.6973 & 6.7499 & 5.2127 & 1.9164 & 3.5174 \\
6.9132 & -4.1123 & 4.0430 & -4.2460 \\
\end{array}
\]
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>randi(imax)</td>
<td>Generates a single random number between 1 and imax.</td>
<td>a = 9</td>
</tr>
<tr>
<td>(imx is an integer)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>randi(imax, n)</td>
<td>Generates an $n \times n$ matrix with random integers between 1 and imax.</td>
<td>b =</td>
</tr>
<tr>
<td></td>
<td></td>
<td>14 3 8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 15 8</td>
</tr>
<tr>
<td>randi(imax, m, n)</td>
<td>Generates an $m \times n$ matrix with random integers between 1 and imax.</td>
<td>c =</td>
</tr>
<tr>
<td></td>
<td></td>
<td>11 2 2 13</td>
</tr>
</tbody>
</table>
```matlab
>> d=randi([50 90],3,4)

   d =
    57   82   71   75
    66   52   67   61
    84   66   76   67

>> d=randn(3,4)

   d =
   -0.4326   0.2877   1.1892   0.1746
  -1.6656  -1.1465  -0.0376  -0.1867
   0.1253   1.1909   0.3273   0.7258

>> v=4*randn(1,6)+50

   v =
    42.7785   57.4344   47.5819   50.4134   52.2527   50.4544

>> w=round(4*randn(1,6)+50)

   w =
    51    49    46    49    50    44
```
EXAMPLE 4-1

Equivalent force system (addition of vectors)

Three forces are applied to a bracket as shown. Determine the total (equivalent) force applied to the bracket.

Solution

A force is a vector (a physical quantity that has a magnitude and direction). In a Cartesian coordinate system a two-dimensional vector \( \mathbf{F} \) can be written as:

\[
\mathbf{F} = F_x \mathbf{i} + F_y \mathbf{j} = F \cos \theta \mathbf{i} + F \sin \theta \mathbf{j} = F (\cos \theta \mathbf{i} + \sin \theta \mathbf{j})
\]

\[
F = \sqrt{F_x^2 + F_y^2} \quad \text{and} \quad \tan \theta = \frac{F_y}{F_x}
\]
% Sample Problem 3-2 solution (script file)
clear
F1M=400; F2M=500; F3M=700;
Th1=-20; Th2=30; Th3=143;

Define variables with the magnitude of each vector.

Define variables with the angle of each vector.

F1=F1M*[cosd(Th1) sind(Th1)]
F2=F2M*[cosd(Th2) sind(Th2)]
F3=F3M*[cosd(Th3) sind(Th3)]

Define the three vectors.

Ftot=F1+F2+F3

Calculate the total force vector.

FtotM=sqrt(Ftot(1)^2+Ftot(2)^2)

Calculate the magnitude of the total force vector.

Th=atand(Ftot(2)/Ftot(1))

Calculate the angle of the total force vector.

\[
\begin{align*}
F1 & = 375.8770 \quad -136.8081 \\
F2 & = 433.0127 \quad 250.0000 \\
F3 & = -559.0449 \quad 421.2705 \\
Ftot & = 249.8449 \quad 534.4625 \\
FtotM & = 589.9768 \\
Th & = 64.9453
\end{align*}
\]

The components of \(F_1\).
The components of \(F_2\).
The components of \(F_3\).
The components of the total force.
The magnitude of the total force.
The direction of the total force in degrees.
Friction experiment (element-by-element calculations)

The coefficient of friction, $\mu$, can be determined in an experiment by measuring the force $F$ required to move a mass $m$. When $F$ is measured and $m$ is known, the coefficient of friction can be calculated by:

$$\mu = \frac{F}{(mg)} \quad (g = 9.81 \text{ m/s}^2).$$

<table>
<thead>
<tr>
<th>Test</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mass $m$ (kg)</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>10</td>
<td>20</td>
<td>50</td>
</tr>
<tr>
<td>Force $F$ (N)</td>
<td>12.5</td>
<td>23.5</td>
<td>30</td>
<td>61</td>
<td>117</td>
<td>294</td>
</tr>
</tbody>
</table>
Enter the values of \( m \) in a vector.
\[ m = [2 \ 4 \ 5 \ 10 \ 20 \ 50] \]

Enter the values of \( F \) in a vector.
\[ F = [12.5 \ 23.5 \ 30 \ 61 \ 117 \ 294] \]

\[ m_u = F ./ (m \times 9.81) \]

A value for \( m_u \) is calculated for each test, using element-by-element calculations.

\[
\begin{array}{cccccc}
0.6371 & 0.5989 & 0.6116 & 0.6218 & 0.5963 & 0.5994 \\
\end{array}
\]

The average of the elements in the vector \( m_u \) is determined by using the function `mean`.

\[ m_u_{\text{ave}} = \text{mean}(mu) \]

\[
\begin{array}{c}
m_u_{\text{ave}} = 0.6109 \\
\end{array}
\]
EXAMPLE 4-3

Electrical resistive network analysis (solving a system of linear equations)

The electrical circuit shown consists of resistors and voltage sources. Determine the current in each resistor using the mesh current method, which is based on Kirchhoff’s voltage law.

\[ V_1 = 20 \text{ V}, \quad V_2 = 12 \text{ V}, \quad V_3 = 40 \text{ V} \]
\[ R_1 = 18 \Omega, \quad R_2 = 10 \Omega, \quad R_3 = 16 \Omega \]
\[ R_4 = 6 \Omega, \quad R_5 = 15 \Omega, \quad R_6 = 8 \Omega \]
\[ R_7 = 12 \Omega, \quad R_8 = 14 \Omega \]

Solution

The equations for the four meshes in the current problem are:

\[ V_1 - R_1 i_1 - R_3 (i_1 - i_3) - R_2 (i_1 - i_2) = 0 \]
\[ -R_5 i_2 - R_2 (i_2 - i_1) - R_4 (i_2 - i_3) - R_7 (i_2 - i_4) = 0 \]
\[ -V_2 - R_6 (i_3 - i_4) - R_4 (i_3 - i_2) - R_3 (i_3 - i_1) = 0 \]
\[ V_3 - R_8 i_4 - R_7 (i_4 - i_2) - R_6 (i_4 - i_3) = 0 \]
The four equations can be rewritten in matrix form \([A][x] = [B]\):

\[
\begin{bmatrix}
-(R_1 + R_2 + R_3) & R_2 & R_3 & 0 \\
R_2 & -(R_2 + R_4 + R_5 + R_7) & R_4 & R_7 \\
R_3 & R_4 & -(R_3 + R_4 + R_6) & R_6 \\
0 & R_7 & R_6 & -(R_6 + R_7 + R_8)
\end{bmatrix}
\begin{bmatrix}
i_1 \\
i_2 \\
i_3 \\
i_4
\end{bmatrix}
= 
\begin{bmatrix}
-V_1 \\
0 \\
V_2 \\
-V_3
\end{bmatrix}
\]

\[V_1=20; \ V_2=12; \ V_3=40;\]
\[R_1=18; \ R_2=10; \ R_3=16; \ R_4=6;\]
\[R_5=15; \ R_6=8; \ R_7=12; \ R_8=14;\]
\[A=[-(R_1+R_2+R_3) \ \ R_2 \ \ R_3 \ \ 0 \\
R_2 \ -(R_2+R_4+R_5+R_7) \ \ R_4 \ \ R_7 \\
R_3 \ R_4 \ -(R_3+R_4+R_6) \ \ R_6 \\
0 \ R_7 \ R_6 \ -(R_6+R_7+R_8)]\]
\[B=[-V_1; \ 0; \ V_2; \ -V_3]\]
\[I=A\backslash B\]
The numerical value of the matrix A.

The numerical value of the vector B.

The solution.
EXAMPLE 4-5

Motion of two particles

A train and a car are approaching a road crossing. At time \( t = 0 \) the train is 400 ft south of the crossing traveling north at a constant speed of 54 mi/h. At the same time the car is 200 ft west of the crossing traveling east at a speed of 28 mi/h and accelerating at 4 ft/s\(^2\). Determine the positions of the train and the car, the distance between them, and the speed of the train relative to the car every second for the next 10 seconds.

Solution

The position of an object that moves along a straight line at a constant acceleration is given by \( s = s_0 + v_0 t + \frac{1}{2} a t^2 \) where \( s_0 \) and \( v_0 \) are the position and velocity at \( t = 0 \), and \( a \) is the acceleration. Applying this equation to the train and the car gives:

\[
\begin{align*}
    y &= -400 + v_{o_{\text{train}}} t \quad (\text{train}) \\
    x &= -200 + v_{o_{\text{car}}} t + \frac{1}{2} a_{\text{car}} t^2 \quad (\text{car})
\end{align*}
\]

The distance between the car and the train is:

\[
d = \sqrt{x^2 + y^2}.
\]
\[ v_{0\text{train}} = \frac{54 \times 5280}{3600}; \quad v_{0\text{car}} = \frac{28 \times 5280}{3600}; \quad a_{\text{car}} = 4; \]

Create variables for the initial velocities (in ft/s) and the acceleration.

\[ t = 0.10; \]

Create the vector \( t \).

\[ y = -400 + v_{0\text{train}} \cdot t; \]

Calculate the train and car positions.

\[ x = -200 + v_{0\text{car}} \cdot t + 0.5 \cdot a_{\text{car}} \cdot t^2; \]

\[ d = \sqrt{x^2 + y^2}; \]

Calculate the distance between the train and car.

\[ v_{\text{car}} = v_{0\text{car}} + a_{\text{car}} \cdot t; \]

Calculate the car's velocity.

\[ \text{speed}_{\text{trainRcar}} = \sqrt{v_{\text{car}}^2 + v_{0\text{train}}^2}; \]

Calculate the speed of the train relative to the car.

\[ \text{table} = [t' \ y' \ x' \ d' \ v_{\text{car}}' \ \text{speed}_{\text{trainRcar}}'] \]

Create a table (see note below).
<table>
<thead>
<tr>
<th>Time (s)</th>
<th>Train position (ft)</th>
<th>Car position (ft)</th>
<th>Car-train distance (ft)</th>
<th>Car speed (ft/s)</th>
<th>Train speed relative to the car (ft/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-400.0000</td>
<td>-200.0000</td>
<td>447.2136</td>
<td>41.0667</td>
<td>89.2139</td>
</tr>
<tr>
<td>1.0000</td>
<td>-320.8000</td>
<td>-156.9333</td>
<td>357.1284</td>
<td>45.0667</td>
<td>91.1243</td>
</tr>
<tr>
<td>2.0000</td>
<td>-241.6000</td>
<td>-109.8667</td>
<td>265.4077</td>
<td>49.0667</td>
<td>93.1675</td>
</tr>
<tr>
<td>3.0000</td>
<td>-162.4000</td>
<td>-58.8000</td>
<td>172.7171</td>
<td>53.0667</td>
<td>95.3347</td>
</tr>
<tr>
<td>4.0000</td>
<td>-83.2000</td>
<td>-3.7333</td>
<td>83.2837</td>
<td>57.0667</td>
<td>97.6178</td>
</tr>
<tr>
<td>5.0000</td>
<td>-4.0000</td>
<td>55.3333</td>
<td>55.4777</td>
<td>61.0667</td>
<td>100.0089</td>
</tr>
<tr>
<td>6.0000</td>
<td>75.2000</td>
<td>118.4000</td>
<td>140.2626</td>
<td>65.0667</td>
<td>102.5003</td>
</tr>
<tr>
<td>7.0000</td>
<td>154.4000</td>
<td>185.4667</td>
<td>241.3239</td>
<td>69.0667</td>
<td>105.0849</td>
</tr>
<tr>
<td>8.0000</td>
<td>233.6000</td>
<td>256.5333</td>
<td>346.9558</td>
<td>73.0667</td>
<td>107.7561</td>
</tr>
<tr>
<td>9.0000</td>
<td>312.8000</td>
<td>331.6000</td>
<td>455.8535</td>
<td>77.0667</td>
<td>110.5075</td>
</tr>
<tr>
<td>10.0000</td>
<td>392.0000</td>
<td>410.6667</td>
<td>567.7245</td>
<td>81.0667</td>
<td>113.3333</td>
</tr>
</tbody>
</table>