Dan B. Marghitu · Mihai Dupac
Nels H. Madsen

Statics with MATLAB®

Springer

Statics with MATLAB®

Dan B. Marghitu · Mihai Dupac
Nels H. Madsen

Statics with MATLAB®

Springer

Dan B. Marghitu
Mechanical Engineering
Auburn University
Auburn, AL
USA

Nels H. Madsen
Samuel Ginn College of Engineering
Auburn University
Auburn, AL
USA

Mihai Dupac
Talbot Campus, School of Design,
Engineering and Computing
Bournemouth University
Poole
UK

ISBN 978-1-4471-5109-8 ISBN 978-1-4471-5110-4 (eBook)
DOI 10.1007/978-1-4471-5110-4
Springer London Heidelberg New York Dordrecht

Library of Congress Control Number: 2013936003

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Engineering mechanics involves the development of mathematical models of the physical world. Statics, a branch of mechanics, addresses the forces acting on and in mechanical objects and systems in equilibrium, and the impact those forces have on the motion, or lack thereof, of those systems. The project deals with the understanding of the mechanical behavior of complex engineering structures and components. The tools of formulating the mathematical equations and the solution methods are discussed. An understanding of forces in and equilibrium of structures and components is most important for their design.

MATLAB is a modern tool that has transformed mathematical methods, because MATLAB not only provides numerical calculations but also facilitates analytical or symbolic calculations using the computer. The present project uses MATLAB as a tool to solve problems. The intent is to show the convenience of MATLAB for theory and applications in statics. This approach will significantly enhance the student's ability to use MATLAB both within statics and beyond. Using examples of problems the MATLAB syntax will be demonstrated. MATLAB is very useful in the process of deriving solutions for any problem in statics. The project will include a large number of problems that are solved using MATLAB. Specific functions dealing with statics topics are introduced and created. The programs will be available on a website accompanying the project.

The main distinction of the study from other projects and books is the use of symbolic MATLAB for both theory and applications. Special attention is given to the solutions of the problems that are solved analytically and numerically using MATLAB. The figures generated with MATLAB will reinforce visual learning for students as they study the programs.

This project is intended primarily for use in a one semester course in statics and could be used in a two semester sequence of courses in statics and dynamics. The project can be used for classroom instruction, for self-study, and in a distance learning environment. It would be appropriate for use as a text at the undergraduate level.

Chapter 1 is intended to give an introduction to vector mechanics. The reason for this chapter is that many scientific concepts used to describe the physical world, have attributes not only of size or magnitude, but also have associated with them the idea of a direction. Examples of such quantities include force, moment, and couple. This chapter provides a starting point for students wishing to develop the basic principle of mechanics. MATLAB is used to calculate the magnitudes of vectors, direction cosines, dot products, cross products, scalar triple products, vector triple products, and derivatives of vector functions. The examples presented begin with a symbolic development, followed by numerical evaluation and the generation of vector figures, all done within MATLAB.

Chapter 2 demonstrates the use of MATLAB in finding the moment of a vector about a point, the moment of a system of vectors, the moment of a couple about a point, the equivalence of systems of vectors, and the force vector and the moment of a force. The figures are depicted using graphical functions built in MATLAB. This chapter also provides an introduction to the basic principles of mechanics.

Chapter 3, centroids and center of mass, presents the principles and details of centroids (also known as the geometric center and connected to the first moment of area) and surface properties, their meaning and importance. All the presentation will be detailed (centroid of a set of points, centroid of a curve, surface or solid, Guldinus-Pappus theorems, parallel-axis theorem) and in some cases followed by examples using MATLAB. External functions can be introduced to calculate the centroids of complex figures. The concepts of the first moment are also useful in analyzing distributed forces.

Chapter 4 analyzes many of the equilibrium problems that are encountered in engineering applications. The equilibrium equations are stated and various types of supports are depicted. The unknown forces and moments acting on bodies are communicated using free-body diagrams and the equilibrium equations are determined. If an object is in equilibrium, the net moment about any point due to the forces and couples acting on the object is zero and the sum of the forces must also be zero. The calculation of moments is explained and the concept of equivalent systems of forces and moments is introduced. In engineering, the term structure can refer to any object that has the capacity to support and exert loads. This chapter studies structures composed of interconnected parts or links. The forces and couples acting on the structure as a total as well as on its individual members are determined. Trusses, which are composed of two-force members, are studied and then frames and machines are considered. MATLAB functions are applied to find and solve the algebraic static equations.

The objective of Chap. 5 is to provide an introduction to friction. Friction forces in engineering applications, have important effects both desirable and undesirable. The Coulomb law of friction is used to find the maximum friction forces that can be exerted by contacting surfaces and the friction forces exerted by sliding surfaces. Threaded connections are also analyzed. MATLAB is used to find friction forces in relation to the associated coefficients of static and kinetic friction.

In the last chapter work and potential energy are described. The work performed when a spring is stretched is stored in the spring as potential energy. Raising an object increases its gravitational potential energy. The principle of virtual work is presented in this chapter. Symbolical and numerical MATLAB are used to solve the examples in this chapter.

Contents

Chapter 1
Operation with Vectors

1.1 Introduction

Vectors are quantities that require the specification of magnitude, orientation, and sense. The characteristics of a vector are the magnitude, the orientation, and the sense. The magnitude of a vector is specified by a positive number and a unit having appropriate dimensions. No unit is stated if the dimensions are those of a pure number. The orientation of a vector is specified by the relationship between the vector and given reference lines and/or planes. The sense of a vector is specified by the order of two points on a line parallel to the vector. Orientation and sense together determine the direction of a vector. The line of action of a vector is a hypothetical infinite straight line collinear with the vector. Displacement, velocity, and force are examples of vectors quantities.

Scalars are mathematical quantities that can be fully defined by specifying their magnitude in suitable units of measure. Mass is a scalar quantity and can be expressed in kilograms, time is a scalar and can be expressed in seconds, and temperature is a scalar quantity that can be expressed in degrees.

To distinguish vectors from scalars it is customary to denote vectors by boldface letters Thus, the displacement vector from point A to point B could be denoted as $\mathbf{r}$ or $\mathbf{r}_{AB}$. The symbol $|\mathbf{r}| = r$ represents themagnitude (or module, norm, or absolute value) of the vector $\mathbf{r}$. In handwritten work a distinguishing mark is used for vectors, such as an arrow over the symbol, $\overrightarrow{r}$ or $\overrightarrow{AB}$, a line over the symbol, $\bar{r}$, or an underline, $\underline{r}$.

Vectors are most frequently depicted by straight arrows. A vector represented by a straight arrow has the direction indicated by the arrow. The displacement vector from point A to point B is depicted in Fig. 1.1 as a straight arrow. In some cases it is necessary to depict a vector whose direction is perpendicular to the surface in which the representation will be drawn. Under this circumstance the use of a portion of a circle with a direction arrow is useful. The orientation of the vector is perpendicular to the plane containing the circle and the sense of the vector is the same as the direction in which a right-handed screw moves when the axis of the screw is normal

D. B. Marghitu et al., *Statics with MATLAB®*,
DOI: 10.1007/978-1-4471-5110-4_1, © Springer-Verlag London 2013

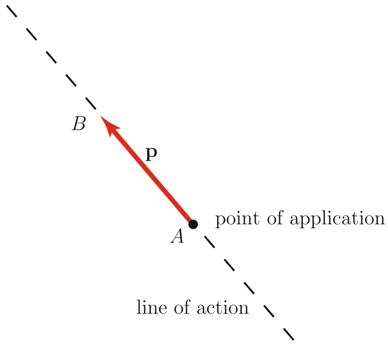


Fig. 1.1 Vector

to the plane in which the arrow is drawn and the screw is rotated as indicated by the arrow. A bound vector is a vector associated with a particular point A in space. The point A is the point of application of the vector, and the line passing through P and parallel to the vector is the line of action of the vector. The point of application may be represented as the tail, Fig. 1.1, or the head of the vector arrow. A free vector is not associated with any particular point in space. A transmissible (or sliding) vector is a vector that can be moved along its line of action without change of meaning. The operations of vector analysis deal only with the characteristics of vectors and apply, therefore, to bound, free, and transmissible vectors.

Two vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ are said to be equal to each other when they have the same characteristics and $\mathbf{v}_1 = \mathbf{v}_2$. Equality does not imply physical equivalence. For instance, two forces represented by equal vectors do not necessarily cause identical motions of a body on which they act.

The product of a vector $\mathbf{v}$ and a scalar s, $s\mathbf{v}$ or $\mathbf{v}s$, is a vector having the following characteristics:

1. Magnitude. $|s\mathbf{v}| \equiv |\mathbf{v}s| = |s||\mathbf{v}|$, where $|s| = s$ denotes the absolute value (or magnitude, or module) of the scalar s.
2. Orientation. $s\mathbf{v}$ is parallel to $\mathbf{v}$. If $s = 0$, no definite orientation is attributed to $s\mathbf{v}$.
3. Sense. If $s > 0$, the sense of $s\mathbf{v}$ is the same as that of $\mathbf{v}$. If $s < 0$, the sense of $s\mathbf{v}$ is opposite to that of $\mathbf{v}$. If $s = 0$, no definite sense is attributed to $s\mathbf{v}$.

A zero vector is a vector that does not have a definite direction and whose magnitude is equal to zero. The symbol used to denote a zero vector is $\mathbf{0}$.

A unit vector is a vector with magnitude equal to 1. Given a vector $\mathbf{v}$, a unit vector $\mathbf{u}$ having the same direction as $\mathbf{v}$ is obtained by forming the product of $\mathbf{v}$ with the reciprocal of the magnitude of $\mathbf{v}$

$$\mathbf{u} = \mathbf{v}\,\frac{1}{|\mathbf{v}|} = \frac{\mathbf{v}}{|\mathbf{v}|}.$$

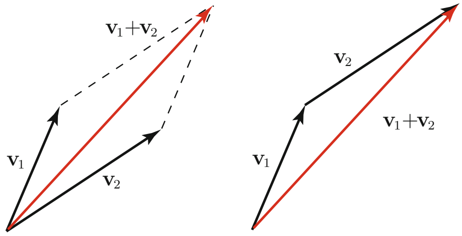


Fig. 1.2 Vector addition: parallelogram law

1.2 Vector Addition

The sum of a vector $\mathbf{v}_1$ and a vector $\mathbf{v}_2$: $\mathbf{v}_1 + \mathbf{v}_2$ or $\mathbf{v}_2 + \mathbf{v}_1$ is a vector whose characteristics can be found by either graphical or analytical processes. The vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ add according to the parallelogram law: the vector $\mathbf{v}_1 + \mathbf{v}_2$ is represented by the diagonal of a parallelogram formed by the graphical representation of the vectors, see Fig. 1.2. The vector $\mathbf{v}_1 + \mathbf{v}_2$ is called the resultant of $\mathbf{v}_1$ and $\mathbf{v}_2$. The vectors can be added by moving them successively to parallel positions so that the head of one vector connects to the tail of the next vector. The resultant is the vector whose tail connects to the tail of the first vector, and whose head connects to the head of the last vector. The sum $\mathbf{v}_1 + (-\mathbf{v}_2)$ is called the difference of $\mathbf{v}_1$ and $\mathbf{v}_2$ and is denoted by $\mathbf{v}_1 - \mathbf{v}_2$. The sum of n vectors $\mathbf{v}_i$, $i = 1, \ldots, n$,

$$\sum_{i=1}^{n} \mathbf{v}_i \text{ or } \mathbf{v}_1 + \mathbf{v}_2 + \cdots + \mathbf{v}_n$$

is called the resultant of the vectors $\mathbf{v}_i$, $i = 1, \ldots n$. Vector addition is:

1. commutative, that is, the characteristics of the resultant are independent of the order in which the vectors are added (commutativity law for addition)

$$\mathbf{v}_1 + \mathbf{v}_2 = \mathbf{v}_2 + \mathbf{v}_1.$$

2. associative, that is, the characteristics of the resultant are not affected by the manner in which the vectors are grouped (associativity law for addition)

$$\mathbf{v}_1 + (\mathbf{v}_2 + \mathbf{v}_3) = (\mathbf{v}_1 + \mathbf{v}_2) + \mathbf{v}_3.$$

3. distributive, that is, the vector addition obeys the following laws of distributivity

$$(s_1 + s_2)\mathbf{v} = s_1\mathbf{v} + s_2\mathbf{v} \quad \text{and} \quad s(\mathbf{v}_1 + \mathbf{v}_2) = s\mathbf{v}_1 + s\mathbf{v}_2,$$

or equivalent (for the general case)

$$\mathbf{v}\sum_{i=1}^{n} s_i = \sum_{i=1}^{n}(\mathbf{v}s_i) \quad \text{and} \quad s\sum_{i=1}^{n} \mathbf{v}_i = \sum_{i=1}^{n}(s\mathbf{v}_i).$$

Moreover, the characteristics of the resultant is not affected by the manner in which the vector is multiplied with scalars (associativity law for multiplication)

$$s_1 (s_2 \mathbf{v}) = (s_1 s_2) \mathbf{v}.$$

Every vector can be regarded as the sum of n vectors ($n = 2, 3, \ldots$) of which all but one can be selected arbitrarily.

1.3 Linear Independence

If $\mathbf{v}_i, i = 1, \ldots n$ are vectors and $s_i, i = 1, \ldots n$ are scalars, then a linear combination of the vectors with the scalars as coefficients is defined as $\sum_{i=1}^{n} s_i \mathbf{v}_i = s_1 \mathbf{v}_1 + \cdots + s_n \mathbf{v}_n$. A collection of non-zero vectors is said to be linearly independent if no vector in the set can be written as a linear combination of the remaining vectors in the set. The dimension of the space is equal to the maximum number of non-zero vectors that can be included in a linearly independent set of vectors. Thus, for a three-dimensional space the maximum number of non-zero vectors in a linearly independent collection is three. Given a set of three linearly independent vectors, any other vector can be constructed as a resultant of scalar multiplication of the three vectors. Such a set of vectors is called a basis set. A set of vectors which is not linearly independent is called linearly dependent.

1.4 Resolution of Vectors

Let $\mathbf{\imath}_1, \mathbf{\imath}_2, \mathbf{\imath}_3$ be three linearly independent unit vectors as a basis set $|\mathbf{\imath}_1| = |\mathbf{\imath}_2| = |\mathbf{\imath}_3| = 1$. For a given vector $\mathbf{v}$, Fig. 1.3, there exist three unique scalars v_1, v_1, v_3, such that $\mathbf{v}$ can be expressed as

$$\mathbf{v} = v_1 \mathbf{\imath}_1 + v_2 \mathbf{\imath}_2 + v_3 \mathbf{\imath}_3. \tag{1.1}$$

The opposite action of addition of vectors is the resolution of vectors. Thus, for the given vector $\mathbf{v}$ the vectors $v_1 \mathbf{\imath}_1$, $v_2 \mathbf{\imath}_2$, and $v_3 \mathbf{\imath}_3$ sum to the original vector. The vector $v_p \mathbf{\imath}_p$ is called the $\mathbf{\imath}_p$ component of $\mathbf{v}$ relative to the given basis set where $p = 1, 2, 3$. A vector is often replaced by its components since the components are equivalent to the original vector.

Frequently a vector will be given and its components relative to a particular basis set need to be calculated. A trivial example of this situation occurs when the vector to be resolved is the zero vector. Then each of its components are zero. Thus, under these circumstances every vector equation $\mathbf{v} = \mathbf{0}$, where $\mathbf{v} = v_1 \mathbf{\imath}_1 + v_2 \mathbf{\imath}_2 + v_3 \mathbf{\imath}_3$, is

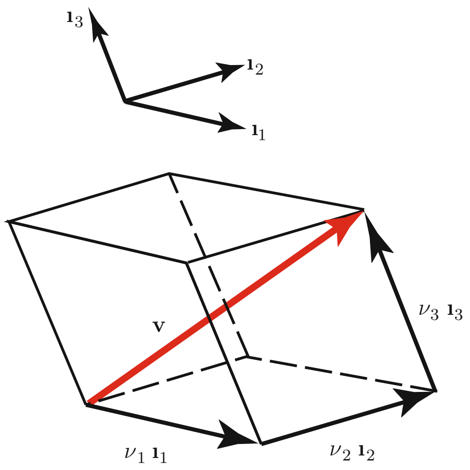


Fig. 1.3 Resolution of a vector **v**

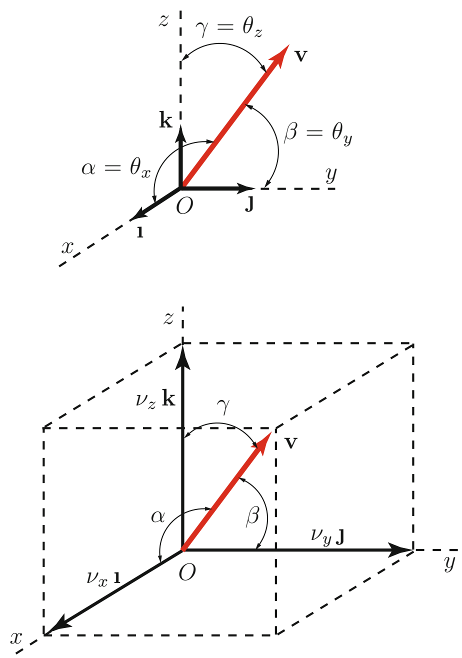


Fig. 1.4 Cartesian reference frame, orthogonal scalar components v_x, v_y, v_z, and direction cosines, α, β, γ

equivalent to three scalar equations $v_1 = 0$, $v_2 = 0$, $v_3 = 0$. Note that the zero vector **0** is not the number zero.

If the unit vectors $\mathbf{\imath}_1, \mathbf{\imath}_2, \mathbf{\imath}_3$ are mutually perpendicular they form a cartesian basis or a cartesian reference frame. For a cartesian reference frame the following notation is used: (Fig. 1.4)

$$\imath_1 \equiv \imath,\ \imath_2 \equiv \jmath,\ \imath_3 \equiv \mathbf{k} \quad \text{and} \quad \imath \perp \jmath,\ \imath \perp \mathbf{k},\ \jmath \perp \mathbf{k}.$$

The symbol $\perp$ denotes perpendicular. When a vector $\mathbf{v}$ is expressed in the form $\mathbf{v} = v_x\imath + v_y\jmath + v_z\mathbf{k}$ where $\imath, \jmath, \mathbf{k}$ are mutually perpendicular unit vectors (cartesian reference frame or orthogonal reference frame), the magnitude of $\mathbf{v}$ is given by

$$|\mathbf{v}| = \sqrt{v_x^2 + v_y^2 + v_z^2}. \tag{1.2}$$

The vectors $\mathbf{v}_x = v_x\imath$, $\mathbf{v}_y = v_y\jmath$, and $\mathbf{v}_z = v_z\mathbf{k}$ are the orthogonal or rectangular component vectors of the vector $\mathbf{v}$. The measures v_x, v_y, v_z are the orthogonal or rectangular scalar components of the vector $\mathbf{v}$.

The resolution of a vector into components frequently facilitate the valuation of a vector equation. If $\mathbf{v}_1 = v_{1x}\imath + v_{1y}\jmath + v_{1z}\mathbf{k}$ and $\mathbf{v}_2 = v_{2x}\imath + v_{2y}\jmath + v_{2z}\mathbf{k}$, then the sum of the vectors is

$$\mathbf{v}_1 + \mathbf{v}_2 = (v_{1x} + v_{2x})\,\imath + \left(v_{1y} + v_{2y}\right)\jmath + (v_{1z} + v_{2z})\,\mathbf{k}.$$

Similarly,

$$\mathbf{v}_1 - \mathbf{v}_2 = (v_{1x} - v_{2x})\,\imath + \left(v_{1y} - v_{2y}\right)\jmath + (v_{1z} - v_{2z})\,\mathbf{k}.$$

In the MATLAB® environment, a three-dimensional row vector `v_` is written as a list of variables `v_ = [ vx vy vz ]` or `v_ = [ vx, vy, vz ]` where `vx`, `vy`, and `vz` are the spatial coordinates of the vector `v`. The elements of a row are separated with blanks or commas. The list of elements are surrounded with square brackets. The first component of the vector `v` is `vx = v(1)`, the second component is `vy = v(2)`, and the third component is `vz = v(3)`. The semicolon `;` is used to separate the end of each row for a column vector. When a variable name is assigned to data, the data is immediately displayed, along with its name. The display of the data can be suppressed by using the semicolon, `;`, at the end of a statement.

Symbolic MATLAB Toolbox can perform symbolical calculation and a vector `v` can be expressed in MATLAB in a symbolical fashion. In MATLAB the `sym` command constructs symbolic variables and expressions. The commands:

```
vx = sym('vx','real');
vy = sym('vy','real');
vz = sym('vz','real');
```

create a symbolic variables `vx`, `vy`, and `vz` and also assume that the variables are real numbers. The symbolic variables can then be treated as mathematical variables. One can use the statement `syms` for generating a shortcut for constructing symbolic objects:

```
syms vx vy vz real
v_ = [ vx vy vz ];
```

where `v_` is a symbolic vector. The same symbolic vector can be created with:

```
v_ = sym('[vx vy vz]');
```

In MATLAB a vector is defined as a matrix with either one row or one column. To make distinction between row vectors and column vectors is essential, especially when operations with vectors are required. Many errors are caused by using a row vector instead a column vector, or vice versa. The command `zeros(m,n)` or `zeros([m n])` returns an `m`-by-`n` matrix of zeros. A zero row vector `[ 0 0 0 ]` is generated with `zeros(1,3)` and a zero column vector is generated with `zeros(3,1)`. The command `ones(m,n)` or `ones([m n])` returns an `m`-by-`n` matrix of ones. In MATLAB two vectors `u_` and `v_` of the same size (defined either as column vectors or row vectors) can be added together using the next command `u_ + v_`. Vectors addition in MATLAB must follow strict rules. The vectors should be either column vectors or row vectors in order to be added and should have the same dimension. It is not possible to add a row vector to a column vector. To subtract one vector from another of the same size, use a minus (−) sign. The subtraction applied to the vectors `u` and `v` can be written in MATLAB as `u_ − v_`. The magnitude of a numerical vector `v_` can be found using the next MATLAB command `norm(v_)`. The MATLAB command `norm(v_)` does not work if the components of the vector `v_` are given symbolically. The symbolic magnitude of a vector `v_` is calculated with:

```
mv = sqrt(v_(1)*v_(1)+v_(2)*v_(2)+v_(3)*v_(3));
```

where the MATLAB statement `sqrt(x)` is the square root of the elements of `x`. To create a unit vector in the direction of the vector `v_` the following relation is used:

```
uv_=v_/sqrt(v_(1)*v_(1)+v_(2)*v_(2)+v_(3)*v_(3))
```

Vector transposition is as easy as adding an apostrophe, `'`, (prime) to the name of the vector. Thus if `v_ = [vx vy vz]` then `v_'` is:

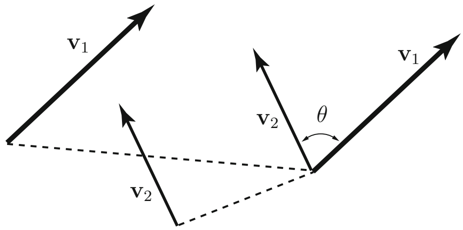
```
vx
vy
vz
```

1.5 Angle Between Two Vectors

The angle between two vectors can be determined by moving either or both vectors parallel to themselves (leaving the sense unaltered) until their initial points (tails) coincide, as shown in Fig. 1.5. This angle will always be in the range between 0° and 180° inclusive. The angle between $\mathbf{v}_1$ and $\mathbf{v}_2$ is the angle θ and is denoted by the symbols $(\mathbf{v}_1, \mathbf{v}_2)$ or $(\mathbf{v}_2, \mathbf{v}_1)$.

The direction of a vector $\mathbf{v} = v_x\mathbf{\imath} + v_y\mathbf{\jmath} + v_z\mathbf{k}$ and relative to a cartesian reference, $\mathbf{\imath}$, $\mathbf{\jmath}$, $\mathbf{k}$, is given by the cosines of the angles formed by the vector and the respective unit vectors. These are called direction cosines and are denoted as (Fig. 1.4)

Fig. 1.5 The angle θ between the vectors $\mathbf{v}_1$ and $\mathbf{v}_2$



$$\cos(\mathbf{v}, \imath) = \cos\alpha = \cos\theta_x = l,\ \cos(\mathbf{v}, \jmath) = \cos\beta = \cos\theta_y = m,\ \text{and}$$
$$\cos(\mathbf{v}, \mathbf{k}) = \cos\gamma = \cos\theta_z = n. \tag{1.3}$$

The following relations exist: $v_x = |\mathbf{v}|\cos\alpha$; $v_y = |\mathbf{v}|\cos\beta$; $v_z = |\mathbf{v}|\cos\gamma$. From these definitions, it follows that

$$\cos^2\alpha + \cos^2\beta + \cos^2\gamma = 1 \quad \text{or} \quad l^2 + m^2 + n^2 = 1. \tag{1.4}$$

Equation (1.4) is proved using the MATLAB commands:

```
syms vx vy vz
v_ = [vx vy vz];
mv=sqrt(v_(1)*v_(1)+v_(2)*v_(2)+v_(3)*v_(3));
l = vx/mv;
m = vy/mv;
n = vz/mv;
sdc=simplify(l^2+m^2+n^2);
fprintf...
 ('l^2+m^2+n^2 = %g \n',eval(sdc))
```

The MATLAB statement `simplify(x)` simplifies each element of the symbolic matrix `x`. Recall, the formula for the unit vector of the vector $\mathbf{v}$ is

$$\mathbf{u_v} = \frac{\mathbf{v}}{|\mathbf{v}|} = \frac{\mathbf{v}}{v} = \frac{v_x}{v}\imath + \frac{v_y}{v}\jmath + \frac{v_x}{v}\mathbf{k},$$

or written another way

$$\mathbf{u_v} = \cos\alpha\imath + \cos\beta\jmath + \cos\gamma\mathbf{k}. \tag{1.5}$$

1.6 Position Vector

The position vector of a point P relative to a point O is a vector $\mathbf{r}_{OP} = \overrightarrow{OP}$ having the following characteristics:

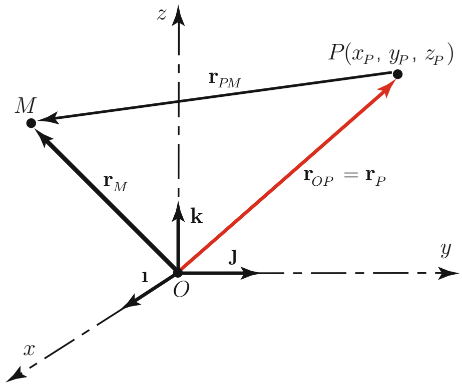


Fig. 1.6 Position vector

1. magnitude the length of line OP;
2. orientation parallel to line OP;
3. sense OP (from point O to point P).

The vector $\mathbf{r}_{OP}$ is shown as an arrow connecting O to P, as depicted in Fig. 1.6. The position of a point P relative to P is a zero vector. Let $\mathbf{\imath}$, $\mathbf{\jmath}$, $\mathbf{k}$ be mutually perpendicular unit vectors (cartesian reference frame) with the origin at O, as shown in Fig. 1.6. The axes of the cartesian reference frame are x, y, z. The unit vectors $\mathbf{\imath}$, $\mathbf{\jmath}$, $\mathbf{k}$ are parallel to x, y, z, and they have the senses of the positive x, y, z axes. The coordinates of the origin O are $x = y = z = 0$, i.e., $O(0,\ 0,\ 0)$. The coordinates of a point P are $x = x_P,\ y = y_P,\ z = z_P$, i.e., $P(x_P, y_P, z_P)$. The position vector of P relative to the origin O is

$$\mathbf{r}_{OP} = \mathbf{r}_P = \overrightarrow{OP} = x_P\mathbf{\imath} + y_P\mathbf{\jmath} + z_P\,\mathbf{k}. \tag{1.6}$$

The coordinates of a point M, $M \neq O$, are $(x_M,\ y_M,\ z_M)$. The position vector of the point M relative to a point P is

$$\mathbf{r}_{PM} = \overrightarrow{PM} = (x_M - x_P)\mathbf{\imath} + (y_M - y_P)\,\mathbf{\jmath} + (z_M - z_P)\,\mathbf{k}. \tag{1.7}$$

The distance d between P and M is given by

$$d = |\mathbf{r}_M - \mathbf{r}_P| = |\mathbf{r}_{PM}| = |\overrightarrow{PM}| = \sqrt{(x_M - x_P)^2 + (y_M - y_P)^2 + (z_M - z_P)^2}.$$

1.7 Scalar Product of Vectors

The scalar (dot) product of a vector $\mathbf{v}_1$ and a vector $\mathbf{v}_2$ is

$$\mathbf{v}_1 \cdot \mathbf{v}_1 = |\mathbf{v}_1|\,|\mathbf{v}_2|\,\cos(\mathbf{v}_1,\ \mathbf{v}_2). \tag{1.8}$$

For the scalar (dot) product the following rules apply:

1. for any vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ one can write the commutative law for scalar product

$$\mathbf{v}_1 \cdot \mathbf{v}_2 = \mathbf{v}_2 \cdot \mathbf{v}_1.$$

2. for any two vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ and any scalar s the following relation is written

$$(s\mathbf{v}_1)\cdot\mathbf{v}_2 = s(\mathbf{v}_1 \cdot \mathbf{v}_2) = \mathbf{v}_1\cdot(s\mathbf{v}_2) = s\mathbf{v}_1\cdot\mathbf{v}_2.$$

3. for any vectors $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$ the distributive law in the first argument is

$$(\mathbf{v}_1 + \mathbf{v}_2) \cdot \mathbf{v}_3 = \mathbf{v}_1 \cdot \mathbf{v}_3 + \mathbf{v}_2 \cdot \mathbf{v}_3,$$

and the distributive law in the second argument is

$$\mathbf{v}_1 \cdot (\mathbf{v}_2 + \mathbf{v}_3) = \mathbf{v}_1 \cdot \mathbf{v}_2 + \mathbf{v}_1 \cdot \mathbf{v}_3.$$

It can be shown that the dot product is distributive and the following relation can be written

$$s_a\,\mathbf{v}_1 \cdot (s_b\,\mathbf{v}_2 + s_c\,\mathbf{v}_3) = s_a\,s_b\,\mathbf{v}_1 \cdot \mathbf{v}_2 + s_a\,s_c\,\mathbf{v}_1 \cdot \mathbf{v}_3.$$

If

$$\mathbf{a} = a_x\boldsymbol{\imath} + a_y\boldsymbol{\jmath} + a_z\mathbf{k} \quad \text{and} \quad \mathbf{b} = b_x\boldsymbol{\imath} + b_y\boldsymbol{\jmath} + b_z\mathbf{k},$$

where $\boldsymbol{\imath}$, $\boldsymbol{\jmath}$, $\mathbf{k}$ are mutually perpendicular unit vectors, then

$$\mathbf{a} \cdot \mathbf{b} = a_x b_x + a_y b_y + a_z b_z\,. \tag{1.9}$$

The following relationships exist

$$\boldsymbol{\imath} \cdot \boldsymbol{\imath} = \boldsymbol{\jmath} \cdot \boldsymbol{\jmath} = \mathbf{k} \cdot \mathbf{k} = 1,$$
$$\boldsymbol{\imath} \cdot \boldsymbol{\jmath} = \boldsymbol{\jmath} \cdot \boldsymbol{\imath} = \boldsymbol{\jmath} \cdot \mathbf{k} = \mathbf{k} \cdot \boldsymbol{\jmath} = \mathbf{k} \cdot \boldsymbol{\imath} = \boldsymbol{\imath} \cdot \mathbf{k} = 0.$$

Every vector $\mathbf{v}$ can be expressed in the form

$$\mathbf{v} = \boldsymbol{\imath} \cdot \mathbf{v}\,\boldsymbol{\imath} + \boldsymbol{\jmath} \cdot \mathbf{v}\boldsymbol{\jmath} + \mathbf{k} \cdot \mathbf{v}\mathbf{k}. \tag{1.10}$$

Proof The vector $\mathbf{v}$ can always be expressed as

$$\mathbf{v} = v_x\boldsymbol{\imath} + v_y\boldsymbol{\jmath} + v_z\mathbf{k}.$$

Dot multiply both sides by $\boldsymbol{\imath}$

$$\boldsymbol{\imath} \cdot \mathbf{v} = v_x\boldsymbol{\imath} \cdot \boldsymbol{\imath} + v_y\boldsymbol{\imath} \cdot \boldsymbol{\jmath} + v_z\boldsymbol{\imath} \cdot \mathbf{k}.$$

But,

$$\mathbf{\imath}\cdot\mathbf{\imath}=1 \text{ and } \mathbf{\imath}\cdot\mathbf{\jmath}=\mathbf{\imath}\cdot\mathbf{k}=0.$$

Hence, $\mathbf{\imath}\cdot\mathbf{v} = v_x$. Similarly, $\mathbf{\jmath}\cdot\mathbf{v} = v_y$ and $\mathbf{k}\cdot\mathbf{v} = v_z$. The MATLAB command `dot(v_, u_)` calculates the scalar product (or vector dot product) of the vectors `v` and `u`. The dot product of two vectors `v_` and `u_` can be expressed as:

```
sum(v_.*u_)
```

The command `sum(x_)` with `x_` defined as a vector, returns the sum of its elements. The MATLAB command `.*`, named *array multiplication* is the element-by-element product of the associated arrays, i.e., `v_.*u_`, and the arrays must have the same size, unless one of them is a scalar. To indicate an array (element-by-element) operation, the standard operator is preceded with a period (dot). Thus `v_.*u_` is:

```
[ vx*ux, vy*uy, vz*uz ]
```

1.8 Vector Product of Vectors

The vector (cross) product of a vector $\mathbf{v}_1$ and a vector $\mathbf{v}_2$ is the vector (Fig. 1.7)

$$\mathbf{v}_1 \times \mathbf{v}_2 = |\mathbf{v}_1|\,|\mathbf{v}_2|\sin(\mathbf{v}_1,\ \mathbf{v}_2)\mathbf{n} \tag{1.11}$$

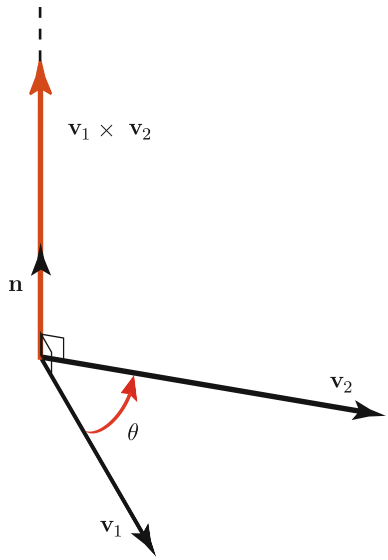


Fig. 1.7 Vector (*cross*) product of the vector **a** and the vector **b**

where $\mathbf{n}$ is a unit vector whose direction is the same as the direction of advance of a right-handed screw rotated from $\mathbf{v}_1$ toward $\mathbf{v}_2$, through the angle $(\mathbf{v}_1,\ \mathbf{v}_2)$, when the axis of the screw is perpendicular to both $\mathbf{v}_1$ and $\mathbf{v}_2$. The magnitude of $\mathbf{v}_1 \times \mathbf{v}_2$ is given by

$$|\mathbf{v}_1 \times \mathbf{v}_2| = |\mathbf{v}_1|\,|\mathbf{v}_2|\sin(\mathbf{v}_1,\ \mathbf{v}_2).$$

If $\mathbf{v}_1$ is parallel to $\mathbf{v}_2$, $\mathbf{v}_1||\mathbf{v}_2$, then $\mathbf{v}_1 \times \mathbf{v}_2 = \mathbf{0}$. The symbol $||$ denotes parallel. The relation $\mathbf{v}_1 \times \mathbf{v}_2 = \mathbf{0}$ implies only that the product $|\mathbf{v}_1|\,|\mathbf{v}_2|\sin(\mathbf{v}_1,\ \mathbf{v}_2)$ is equal to zero, and this is the case whenever $|\mathbf{v}_1| = 0$, or $|\mathbf{v}_2| = 0$, or $\sin(\mathbf{v}_1,\ \mathbf{v}_2) = 0$.

For any two vectors $\mathbf{v}_1$ and $\mathbf{v}_1$ and any real scalar s the following relation can be written

$$(s\mathbf{v}_1) \times \mathbf{v}_2 = s(\mathbf{v}_1 \times \mathbf{v}_2) = \mathbf{v}_1 \times (s\mathbf{v}_2) = s\mathbf{v}_1 \times \mathbf{v}_2.$$

The sense of the unit vector $\mathbf{n}$ which appears in the definition of $\mathbf{v}_1 \times \mathbf{v}_2$ depends on the order of the factors $\mathbf{v}_1$ and $\mathbf{v}_2$ in such a way that (cross product is not commutative)

$$\mathbf{v}_1 \times \mathbf{v}_2 = -\mathbf{v}_1 \times \mathbf{v}_2. \tag{1.12}$$

The cross product distributive law for the first argument can be written as

$$(\mathbf{v}_1 + \mathbf{v}_2) \times \mathbf{v}_3 = \mathbf{v}_1 \times \mathbf{v}_3 + \mathbf{v}_2 \times \mathbf{v}_3,$$

while the distributive law for the second argument is

$$\mathbf{v}_1 \times (\mathbf{v}_2 + \mathbf{v}_3) = \mathbf{v}_1 \times \mathbf{v}_2 + \mathbf{v}_1 \times \mathbf{v}_3.$$

Vector multiplication obeys the following law of distributivity (Varignon theorem)

$$\mathbf{a} \times \sum_{i=1}^{n} \mathbf{v}_i = \sum_{i=1}^{n} (\mathbf{a} \times \mathbf{v}_i).$$

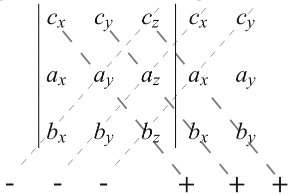
A set of mutually perpendicular unit vectors $\mathbf{\imath}$, $\mathbf{\jmath}$, $\mathbf{k}$ is called right-handed if $\mathbf{\imath} \times \mathbf{\jmath} = \mathbf{k}$. A set of mutually perpendicular unit vectors $\mathbf{\imath}$, $\mathbf{\jmath}$, $\mathbf{k}$ is called left-handed if $\mathbf{\imath} \times \mathbf{\jmath} = -\mathbf{k}$. If $\mathbf{a} = a_x\mathbf{\imath} + a_y\mathbf{\jmath} + a_z\mathbf{k}$, and $\mathbf{b} = b_x\mathbf{\imath} + b_y\mathbf{\jmath} + b_z\mathbf{k}$, where $\mathbf{\imath}$, $\mathbf{\jmath}$, $\mathbf{k}$ are right-handed mutually perpendicular unit vectors, then $\mathbf{a} \times \mathbf{b}$ can be expressed in the following determinant form

$$\mathbf{a} \times \mathbf{b} = \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix}. \tag{1.13}$$

The determinant can be expanded by minors of the elements of the first row

$$\begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix} = \mathbf{\imath} \begin{vmatrix} a_y & a_z \\ b_y & b_z \end{vmatrix} - \mathbf{\jmath} \begin{vmatrix} a_x & a_z \\ b_x & b_z \end{vmatrix} + \mathbf{k} \begin{vmatrix} a_x & a_y \\ b_x & b_y \end{vmatrix}$$
$$= \mathbf{\imath}(a_y\, b_z - a_z\, b_y) - \mathbf{\jmath}(a_x\, b_z - a_z\, b_x) + \mathbf{k}(a_x\, b_y - a_y\, b_x)$$
$$= (a_y\, b_z - a_z\, b_y)\mathbf{\imath} + (a_z\, b_x - a_x\, b_z)\mathbf{\jmath} + (a_x\, b_y - a_y\, b_x)\mathbf{k}. \tag{1.14}$$

As a general rule a third order determinant can be expanded by diagonal multiplication, i.e., repeating the first two columns on the right side of the determinant, and adding the signed diagonal products of the diagonal elements as



The determinant in Eq. (1.13) can be expanded using the general rule as

$$\begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix} = -\mathbf{k}\, a_y\, b_x - \mathbf{\imath}\, a_z\, b_y - \mathbf{\jmath}\, a_x\, b_z + \mathbf{\imath}\, a_y\, b_z + \mathbf{\jmath}\, a_z\, b_x + \mathbf{k}\, a_x b_y$$
$$= (a_y\, b_z - a_z\, b_y)\mathbf{\imath} + (a_z\, b_x - a_x\, b_z)\mathbf{\jmath} + (a_x\, b_y - a_y\, b_x)\mathbf{k}.$$

The MATLAB command `cross(a, b)` calculates the cross product of the vectors `a` and `b`.

1.9 Scalar Triple Product of Three Vectors

The scalar triple product of three vectors **a**, **b**, and **c** is defines as

$$[\mathbf{a}, \mathbf{b}, \mathbf{c}] \equiv \mathbf{a}\cdot(\mathbf{b} \times \mathbf{c}) = \mathbf{a} \cdot \mathbf{b} \times \mathbf{c}. \tag{1.15}$$

The MATLAB commands for the scalar triple product of three vectors `a_`, `b_`, and `c_` is:

```
syms ax ay az bx by bz cx cy cz real
a_=[ax ay az]; b_=[bx by bz]; c_=[cx cy cz];
% [a_,b_,c_] = a_.(b_ x c_)
abc = dot(a_, cross(b_, c_));
```

It does not matter whether the dot is placed between **a** and **b**, and the cross between **b** and **c**, or vice versa, that is,

$$[\mathbf{a}, \mathbf{b}, \mathbf{c}] = \mathbf{a} \cdot \mathbf{b} \times \mathbf{c} = \mathbf{a} \times \mathbf{b} \cdot \mathbf{c}. \tag{1.16}$$

The relation given by Eq. (1.16) is demonstrated using the MATLAB commands:

```
% [a_,b_,c_] = a_.(b_ x c_)
abxc = simplify(dot(a_, cross(b_, c_)));
% [a_,b_,c_] = (a_ x b_).c_
axbc = simplify(dot(cross(a_, b_), c_));
% a_.(b_ x c_)==(a_ x b_).c_
abxc == axbc
```

The MATLAB relational operator `==` or `eq` is used to compare each element of array for equality. The statement `LHS == RHS` or `eq(LHS, RHS)` compares each element of the array `LHS` for equality with the corresponding element of the array `RHS`, and returns an array with elements set to logical 1 (true) if `LHS` and `RHS` are equal, or logical 0 (false) where they are not equal.

A change in the order of the factors appearing in a scalar triple product at most changes the sign of the product, that is,

$$[\mathbf{b}, \mathbf{a}, \mathbf{c}] = -[\mathbf{a}, \mathbf{b}, \mathbf{c}] \quad \text{and} \quad [\mathbf{b}, \mathbf{c}, \mathbf{a}] = [\mathbf{a}, \mathbf{b}, \mathbf{c}].$$

If **a**, **b**, **c** are parallel to the same plane, or if any two of the vectors **a**, **b**, **c** are parallel to each other, then $[\mathbf{a}, \mathbf{b}, \mathbf{c}] = 0$.

The scalar triple product $[\mathbf{a}, \mathbf{b}, \mathbf{c}]$ can be expressed in the following determinant form

$$[\mathbf{a}, \mathbf{b}, \mathbf{c}] = \begin{vmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \end{vmatrix}. \tag{1.17}$$

In MATLAB the scalar triple product of three vectors `a_`, `b_`, and `c_` is expressed as:

```
det([a_; b_; c_])
```

where `det(x)` is the determinant of the square matrix `x`. To verify Eq. (1.17) the following MATLAB command is used:

```
det([a_; b_; c_]) == simplify(dot(a_, cross(b_, c_)))
```

1.10 Vector Triple Product of Three Vector

The vector triple product of three vectors **a**, **b**, **c** is the vector $\mathbf{a} \times (\mathbf{b} \times \mathbf{c})$. The parentheses are essential because $\mathbf{a} \times (\mathbf{b} \times \mathbf{c})$ is not, in general, equal to $(\mathbf{a} \times \mathbf{b}) \times \mathbf{c}$. For any three vectors **a**, **b**, and **c**

$$\mathbf{a} \times (\mathbf{b} \times \mathbf{c}) = (\mathbf{a} \cdot \mathbf{c})\mathbf{b} - (\mathbf{a} \cdot \mathbf{b})\mathbf{c}\,. \tag{1.18}$$

The previous relation given by Eq. (1.18) can be explained using the MATLAB statements:

```
% a_ x (b_ x c_)
axbxc = cross(a_, cross(b_, c_));
% (a_.c_)b_ - (a_.b_)c_
RHS = dot(a_, c_)*b_ - dot(a_, b_)*c_;
% a_ x (b_ x c_) - (a_.c_)b_ + (a_.b_)c_ = [0, 0, 0]
simplify(axbxc-RHS)
```

1.11 Derivative of a Vector Function

The derivative of a vector function is defined in exactly the same way as is the derivative of a scalar function. Thus

$$\frac{d}{dt}\mathbf{a} = \lim_{\Delta t \to 0} \frac{\mathbf{a}(t+\Delta t) - \mathbf{a}(t)}{\Delta t}.$$

The derivative of a vector has some of the properties of the derivative of a scalar function. The derivative of the sum of two vector functions **a** and **b** is

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{a}+\mathbf{b}) = \frac{\mathrm{d}\mathbf{a}}{\mathrm{d}t} + \frac{\mathrm{d}\mathbf{b}}{\mathrm{d}t}. \tag{1.19}$$

The components of the vectors `a` and `b` are functions of time, `t`, and are introduced in MATLAB with:

```
syms t real
ax = sym('ax(t)');
ay = sym('ay(t)');
az = sym('az(t)');
bx = sym('bx(t)');
by = sym('by(t)');
bz = sym('bz(t)');
a_ = [ax ay az];
b_ = [bx by bz];
```

To calculate symbolically the derivative of a vector using the MATLAB the command `diff(p,t)` is used, which gives the derivative of `p` with respect to `t`. The relation given by Eq. (1.19) can be demonstrated using the MATLAB command:

```
diff(a_+b_, t) == diff(a_, t) + diff(b_, t)
```

The time derivative of the product of a scalar function f and a vector function **a** is

$$\frac{\mathrm{d}(f\mathbf{a})}{\mathrm{d}t} = \frac{\mathrm{d}f}{\mathrm{d}t}\mathbf{a} + f\frac{\mathrm{d}\mathbf{a}}{\mathrm{d}t}. \tag{1.20}$$

Equation (1.20) is verified using the MATLAB command:

```
syms f real
diff(f*a_, t) == diff(f, t)*a_ + f*diff(a_, t)
```

Combining the previous results one can conclude

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{a}\cdot\mathbf{b}) = \frac{\mathrm{d}\mathbf{a}}{\mathrm{d}t}\cdot\mathbf{b}+\mathbf{a}\cdot\frac{\mathrm{d}\mathbf{b}}{\mathrm{d}t} \quad \text{and} \quad \frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{a}\times\mathbf{b}) = \frac{\mathrm{d}\mathbf{a}}{\mathrm{d}t}\times\mathbf{b}+\mathbf{a}\times\frac{d\mathbf{b}}{\mathrm{d}t}. \tag{1.21}$$

Equation (1.21) is demonstrated with the MATLAB commands:

```
diff(a_*b_.', t) == diff(a_, t)*b_.' + a_*diff(b_,t).'
diff(cross(a_, b_), t) == cross(diff(a_, t), b_) ...
                        + cross(a_, diff(b_, t))
```

where `p_.'` is the array transpose of `p_`.

The general derivative a vector is

$$\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = \frac{\mathrm{d}}{\mathrm{d}t}\left(v_x\,\boldsymbol{\imath}+v_y\,\boldsymbol{\jmath}+v_z\,\mathbf{k},\right) = \frac{\mathrm{d}v_x}{\mathrm{d}t}\,\boldsymbol{\imath}+v_x\frac{\mathrm{d}\boldsymbol{\imath}}{\mathrm{d}t}+\frac{\mathrm{d}v_y}{\mathrm{d}t}\,\boldsymbol{\jmath}+v_y\frac{\mathrm{d}\boldsymbol{\jmath}}{\mathrm{d}t}+\frac{\mathrm{d}v_z}{\mathrm{d}t}\,\mathbf{k}+v_z\frac{\mathrm{d}\mathbf{k}}{\mathrm{d}t},$$

and if the reference basis or reference frame $[\boldsymbol{\imath},\ \boldsymbol{\jmath}, \mathbf{k}]$ is unchanging then

$$\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = \frac{\mathrm{d}v_x}{dt}\,\boldsymbol{\imath}+\frac{\mathrm{d}v_y}{\mathrm{d}t}\,\boldsymbol{\jmath}+\frac{\mathrm{d}v_z}{\mathrm{d}t}\,\mathbf{k}.$$

If V is a function of position and is expressed in terms of a cartesian reference frame as $V = V(x,\ y,\ z)$, then the differential of $\mathrm{d}V$ is

$$\mathrm{d}V = \frac{\partial V}{\partial x}\,\mathrm{d}x+\frac{\partial V}{\partial y}\,\mathrm{d}y+\frac{\partial V}{\partial z}\,\mathrm{d}z. \tag{1.22}$$

Given the function $V = V(x,\ y,\ z)$ expressed in cartesian coordinates, the gradient of V is

$$\nabla V = \frac{\partial V}{\partial x}\,\boldsymbol{\imath}+\frac{\partial V}{\partial y}\,\boldsymbol{\jmath}+\frac{\partial V}{\partial z}\,\mathbf{k}. \tag{1.23}$$

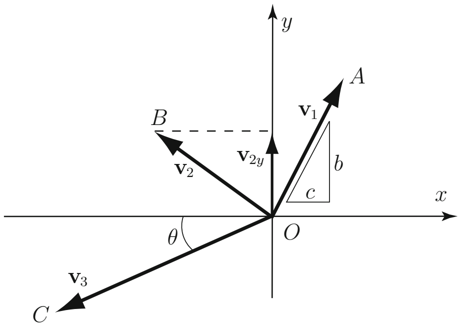
The gradient expressed in cartesian coordinates is

$$\nabla = \frac{\partial}{\partial x}\,\boldsymbol{\imath}+\frac{\partial}{\partial y}\,\boldsymbol{\jmath}+\frac{\partial}{\partial z}\,\mathbf{k}. \tag{1.24}$$

The curl of a vector $\mathbf{F} = F_x\,\boldsymbol{\imath}+F_y\,\boldsymbol{\jmath}+F_z\,\mathbf{k}$ in cartesian coordinates is

$$\nabla\times\mathbf{F} = \begin{vmatrix} \boldsymbol{\imath} & \boldsymbol{\jmath} & \mathbf{k} \\ \dfrac{\partial}{\partial x} & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial z} \\ F_x & F_y & F_z \end{vmatrix}. \tag{1.25}$$

Fig. 1.8 Example 1.1



1.12 Examples

Example 1.1
Find the magnitude and direction of the resultant $\mathbf{v} = \mathbf{v}_1 + \mathbf{v}_2 + \mathbf{v}_3$ of the vectors $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$ shown in the Fig. 1.8. The vector $\mathbf{v}_1 = \overrightarrow{OA}$ is contained in the *xy*-plane, has the slope $m = \dfrac{b}{c}$ and the magnitude v_1. The vector $\mathbf{v}_2 = \overrightarrow{OB}$ has the magnitude v_2 and the magnitude of its *y*-component is v_{2y}. The vector $\mathbf{v}_3 = \overrightarrow{OC}$ has the magnitude v_3 and makes the angle θ with the *x*-axis as shown in the Fig. 1.8. For the numerical application use: $v_1 = 1$ units, $v_2 = 1$ units, $v_{2y} = 0.5$ units, $v_3 = 1.5$ units, $b = 4$ units, $c = 3$ units, and $\theta = 30°$.

Solution
The MATLAB program will start with:

```
clear all
% clears all the objects in the MATLAB workspace and
% resets the default MuPAD symbolic engine
clc % clears the command window and homes the cursor
close all % closes all the open figure windows
```

The numerical data is introduced with the following notation:

```
b=4;          % units
c=3;          % units
v(1) =1;      % units magnitude of vector v1_
v(2) =1;      % units magnitude of vector v2_
vy(2)=0.5;    % units magnitude of vector v2y_
v(3) =1.5;    % units magnitude of vector v3_
theta=pi/6;   % rad   angle of v3_ with x-axis
```

The angle of the vector $\mathbf{v}_1$, in MATLAB `v1_`, with the *x*-axis is calculated with:

```
m=b/c;
alpha(1)=atan(m);
```

where `atan(m)` is the arctangent of `m` in radians. The results is displayed with:

```
fprintf('alpha1=%6.3f(rad)=%6.3f(deg)\n',...
  alpha(1),alpha(1)*180/pi)
```

and the results is:

```
alpha1= 0.927(rad)=53.130(deg)
```

The statement `fprintf(f,format,s)` writes data of array `s` to the file `f`. The `format` is a string in single quotation marks that describes the format of the output fields.

The angle of the vector $\mathbf{v}_2$, in MATLAB `v2_`, with the x-axis is calculated with:

```
phi=asin(vy(2)/v(2));
alpha(2)=pi-phi;
```

where `asin(x)` is the arcsine of `x`. The angle of the vector $\mathbf{v}_3$, in MATLAB `v3_`, with the x-axis is:

```
alpha(3)=pi+theta;
```

The numerical values for the angles `alpha(2)` and `alpha(3)` are:

```
alpha2= 2.618(rad)=150.000(deg)
alpha3= 3.665(rad)=210.000(deg)
```

The x and y components of the three vectors are calculated with:

```
for i = 1:3
    vx(i)=v(i)*cos(alpha(i));
    vy(i)=v(i)*sin(alpha(i));
end
```

The MATLAB commands:

```
for variable = expr, statement, ..., statement end
```

repeat statements a specific number of times. Now the vectors are introduced as:

```
v1_=[vx(1),vy(1),0];
v2_=[vx(2),vy(2),0];
v3_=[vx(3),vy(3),0];
```

and the numerical values are:

```
v1_=[ 0.600, 0.800,0](units)
v2_=[-0.866, 0.500,0](units)
v3_=[-1.299,-0.750,0](units)
```

The resultant of the vectors is:

```
v_ = v1_+v2_+v3_;
```

and numerically `v_=[-1.565, 0.550,0](units)`. The magnitude of `v_` is calculated with:

```
v = \index{norm}norm(v_);
```

where `norm` is vector norm. The magnitude of `v_` can be obtained also with:

```
v = sqrt(v_*v_.');
```

where `v_.'` is the transpose of the `v_` vector. The angle of `v_` with the *x*-axis is:

```
beta=atan(v_(2)/v_(1));
```

The results are:

```
v_=[-1.565, 0.550,0](units)
v=|v_|= 1.659(units)
beta=-0.338(rad)=-19.363(deg)
```

Next the vectors are plotted using MATLAB. The *x*-axis and *y*-axis are labeled using the commands:

```
xlabel('x'), ylabel('y')
```

The statement `axis(([xMIN xMAX yMIN yMAX])` sets scaling for the `x` and `y` axes on the current plot:

```
a = 2;
axis([-a a -a a])
```

To improve the graph a background grid lines were added with the command `grid on`. The command `hold on` locks up the plot and the axis properties and the next graphical commands add to the existing plot. The vectors are introduces with `quiver(x,y,u,v)` that represents the vectors as arrows with components `u,v` at the points `x,y`:

```
for i = 1:3
quiver(0,0,vx(i),vy(i),...
'Color','k','LineWidth',1.5)
end
quiver(0,0,v_(1),v_(2),...
'Color','r','LineWidth',2)
```

The labels for the vectors are introduced with:

```
text(vx(1),vy(1),'v_1',...
'fontsize',12,'fontweight','b')
text(vx(2),vy(2),'  v_2',...
'fontsize',12,'fontweight','b')
text(vx(3),vy(3),'v_3',...
'fontsize',12,'fontweight','b')
text(v_(1),v_(2),'v',...
```

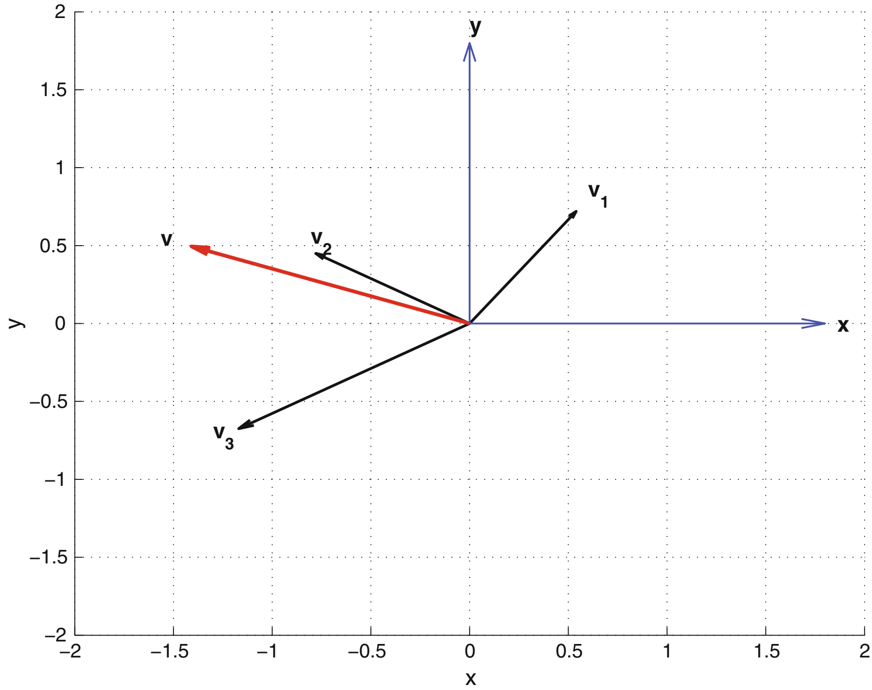


Fig. 1.9 Example 1.1: MATLAB graphical representation

```
'fontsize',12,'fontweight','b')
```

The cartesian axes are plotted with:

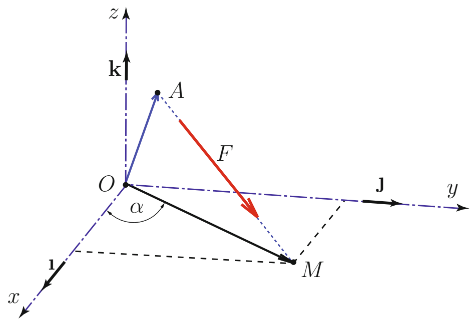
```
quiver(0,0,a,0,...
'Color','b','LineWidth',1.0)
text(a,0,'x',...
'fontsize',12,'fontweight','b')
quiver(0,0,0,a,...
'Color','b','LineWidth',1.0)
text(0,a,'y',...
'fontsize',12,'fontweight','b')
```

The MATLAB graphical representations for this example are shown in Fig. 1.9.

Example 1.2

For the Fig. 1.10 find the $F = 500\,\text{kN}$ force in vector format and then determine its direction cosines. The point M is in the xy-plane with $OM = 50\,\text{m}$. The angle between OM and the x-axis is $\alpha = 45°$. The coordinates of the point A are $x_A = 20\,\text{m}$, $y_A = 10\,\text{m}$, and $z_A = 30\,\text{m}$.

Fig. 1.10 Example 1.2



Solution
The input numerical data are introduced in MATLAB with:

```
F = 500; % kN
OM = 50; % m
xA = 20; % m
yA = 10; % m
zA = 30; % m
alpha = 45; % deg
```

The position vector of the point A is

$$\mathbf{r}_{OA} = \mathbf{r}_A = x_A\,\mathbf{\imath} + y_A\,\mathbf{\jmath} + z_A\,\mathbf{k},$$

and in MATLAB:

```
rA_ = [xA, yA, zA];
```

The components of the position vector of the point M are

$$x_M = OM\ \cos\alpha,\ y_M = OM\ \sin\alpha,\ z_M = 0,$$

and in MATLAB:

```
xM =  OM*cosd(alpha);
yM =  OM*sind(alpha);
zM =  0;
rM_ = [xM, yM, zM];
```

The position vector $\mathbf{r}_{AM}$ can be expressed as

$$\begin{aligned}\mathbf{r}_{AM} &= \mathbf{r}_M - \mathbf{r}_A\\ &= (x_M\,\mathbf{\imath} + y_M\,\mathbf{\jmath} + z_M\mathbf{k}) - (x_A\,\mathbf{\imath} + y_A\,\mathbf{\jmath} + z_A\mathbf{k})\\ &= (x_M - x_A)\ \mathbf{\imath} + (y_M - y_A)\ \mathbf{\jmath} + (z_M - z_A)\ \mathbf{k},\end{aligned}$$

and in MATLAB:

```
rAM_ = rM_-rA_;
```

The magnitude of the vector $\mathbf{r}_{AM}$ is

$$r_{AM} = \sqrt{(x_M - x_A)^2 + (y_M - y_A)^2 + (z_M - z_A)^2}.$$

The magnitude is computed in MATLAB as:

```
rAM = norm(rAM_);
```

The unit vector $\mathbf{u}_F$ of the force $\mathbf{F}$ is calculated with

$$\begin{aligned}
\mathbf{u}_F = \mathbf{u}_{AM} &= \frac{\mathbf{r}_{AM}}{r_{AM}} \\
&= \frac{(x_M - x_A)\,\mathbf{\imath} + (y_M - y_A)\,\mathbf{\jmath} + (z_M - z_A)\,\mathbf{k}}{\sqrt{(x_M - x_A)^2 + (y_M - y_A)^2 + (z_M - z_A)^2}} \\
&= \frac{x_M - x_A}{\sqrt{(x_M - x_A)^2 + (y_M - y_A)^2 + (z_M - z_A)^2}}\,\mathbf{\imath} \\
&\quad + \frac{y_M - y_A}{\sqrt{(x_M - x_A)^2 + (y_M - y_A)^2 + (z_M - z_A)^2}}\,\mathbf{\jmath} \\
&\quad + \frac{z_M - z_A}{\sqrt{(x_M - x_A)^2 + (y_M - y_A)^2 + (z_M - z_A)^2}}\,\mathbf{k}.
\end{aligned}$$

In MATLAB the unit vector is calculated as:

```
uAM_ = rAM_/rAM;
```

One can express the force $\mathbf{F}$, as a magnitude F multiplied by the unit vector $\mathbf{u}_F$ as

$$\mathbf{F} = F\,\mathbf{u}_F = F\,\frac{x_M - x_A}{r_{AM}}\,\mathbf{\imath} + F\,\frac{y_M - y_A}{r_{AM}}\,\mathbf{\jmath} + F\,\frac{z_M - z_A}{r_{AM}}\,\mathbf{k}.$$

The force $\mathbf{F}$ was calculated and printed in MATLAB using the statement:

```
F_=F*uAM_;
```

The direction cosines are calculated in MATLAB with:

```
thetax = acos(F_(1)/F); % alpha
thetay = acos(F_(2)/F); % beta
thetaz = acos(F_(3)/F); % gamma
```

The numerical results are obtained in MATLAB as:

```
rA_=[20.000 10.000 30.000] (m)
rM_=[35.355 35.355  0.000] (m)
```

```
rAM_=[15.355 25.355 -30.000] (m)
uAM_=[ 0.364  0.601 -0.711] (m)

F_=[182.046 300.601 -355.666] (kN)

thetax= 1.198(rad)=68.648(deg)
thetay= 0.926(rad)=53.044(deg)
thetaz= 2.362(rad)=135.343(deg)
```

Next the force F will be plotted using MATLAB. The x-axis, y-axis, and z-axis are labeled using the commands:

```
xlabel('x(m)'), ylabel('y(m)'), zlabel('z(m)')
```

The origin of the reference frame is identified with the statement:

```
text(0,0,0,' O','HorizontalAlignment','right')
```

The statement `axis(([xMIN xMAX yMIN yMAX zMIN zMAX])` sets scaling for the `x`, `y`, and `z` axes on the current plot:

```
sf=30;
axis([-sf sf -sf sf -sf sf])
```

The vectors are introduces with `quiver3(x,y,z,u,v,w)` that represents the vectors as arrows with components `u,v,w` at the points `x,y,z`. The vectors `rA_` and `rM_` are plotted with:

```
quiver3(0,0,0, xA,yA,zA,1,...
    'Color','b','LineWidth',1.5)
quiver3(0,0,0, xM,yM,zM,1,...
    'Color','k','LineWidth',1.5)
```

The labels for the points `A` and `M` are introduced with:

```
text(xA,yA,zA,' A',...
'fontsize',12,'fontweight','b')
text(xM,yM,zM,' M',...
'fontsize',12,'fontweight','b')
```

The vector `F_` is plotted at `M` with:

```
ff=0.1; % force scale factor
quiver3(xM,yM,zM,...
    ff*F_(1),ff*F_(2),ff*F_(3),1,...
    'Color','r','LineWidth',2)
```

The force scale factor `ff` is introduced because the magnitude of the force vector is greater then the magnitude of the position vectors. The line between `A` and `M` are plotted with:

```
line([xA xM],[yA yM],[zA zM],'LineStyle','--')
```

The cartesian axes with the corresponding labels are represented with:

```
quiver3(0,0,0,sf,0,0,1,'Color','k','LineWidth',1)
quiver3(0,0,0,0,sf,0,1,'Color','k','LineWidth',1)
quiver3(0,0,0,0,0,sf,1,'Color','k','LineWidth',1)

text(sf,0,0,'  x','fontsize',12,'fontweight','b')
text(0,sf,0,'  y','fontsize',12,'fontweight','b')
text(0,0,sf,'  z','fontsize',12,'fontweight','b')
```

The graphical representation for the vectors is given by: The MATLAB plots are shown in Fig. 1.11.

Example 1.3
The vector $\overrightarrow{OA} = \mathbf{r}_A$ has the magnitude a and makes the angle θ_x, θ_y, and θ_z with the cartesian axes as shown in the Fig. 1.12. The vector $\overrightarrow{OB} = \mathbf{r}_B$ has the magnitude b and its projection on the xy-plane is the vector $\overrightarrow{OC}$. The angle between the vectors $\overrightarrow{OB}$ and $\overrightarrow{OC}$ is λ and the angle between $\overrightarrow{OC}$ and the x-axis is ν. Find: (a) the resultant $\overrightarrow{R} = \overrightarrow{OA} + \overrightarrow{OB}$, its magnitude, and the direction angles of $\overrightarrow{R}$; (b) the cross product $\mathbf{r}_A \times \mathbf{r}_B$ and the angle between $\mathbf{r}_A$ and $\mathbf{r}_B$; (c) the projection of the vector $\mathbf{r}_A$ on the vector $\mathbf{r}_B$; (d) the scalar triple product $\mathbf{r}_C \cdot (\mathbf{r}_B \times \mathbf{r}_A)$.

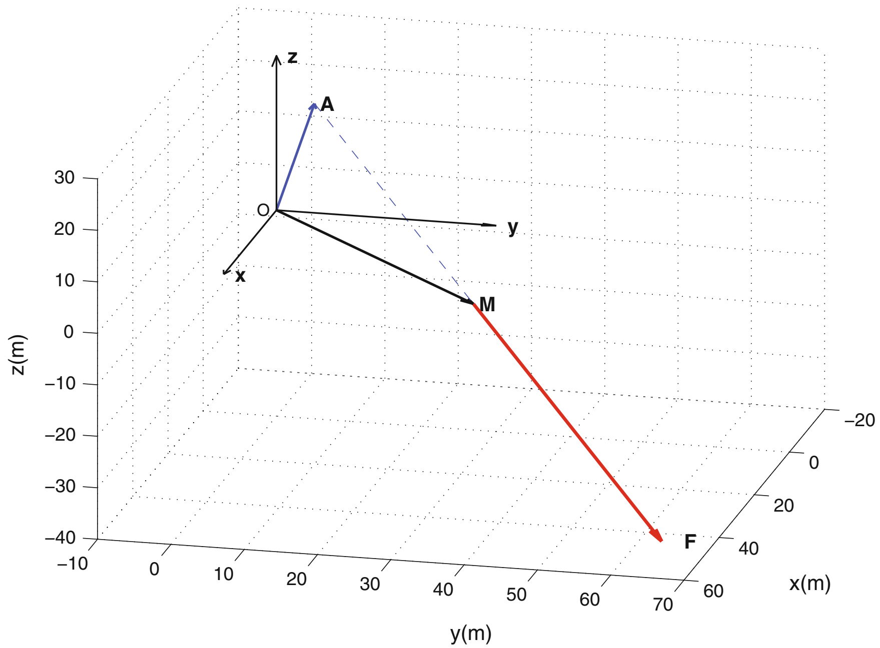


Fig. 1.11 Example 1.2: MATLAB graphical representation

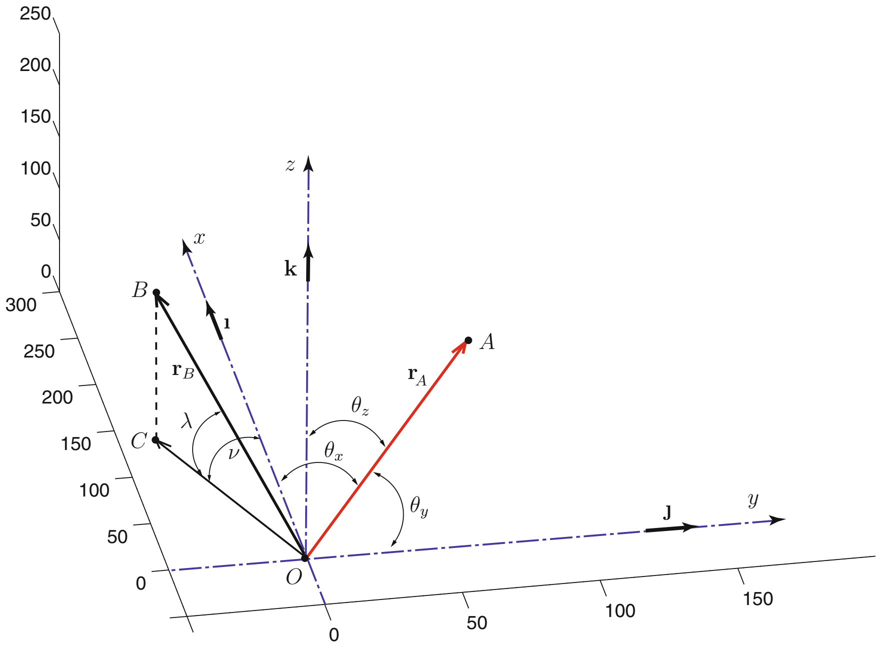


Fig. 1.12 Example 1.3

For the numerical application use: a =150 m, b =200 m, $\theta_x = 30°$, $\theta_y = 60°$, $\theta_z = 60°$, $\lambda = 45°$, and $\nu = 15°$.

Solution
The unit vectors of the cartesian reference frame are [$\mathbf{ı}$, $\mathbf{ȷ}$, $\mathbf{k}$]. The vector $\mathbf{r}_A$ is

$$\mathbf{r}_A = r_{Ax}\,\mathbf{ı} + r_{Ay}\,\mathbf{ȷ} + r_{Az}\,\mathbf{k} = r_A\,\left(\cos\theta_x\,\mathbf{ı} + \cos\theta_y\,\mathbf{ȷ} + \cos\theta_z\,\mathbf{k}\right)$$
$$= a\,\left(\cos\theta_x\,\mathbf{ı} + \cos\theta_y\,\mathbf{ȷ} + \cos\theta_z\,\mathbf{k}\right).$$

The vector $\mathbf{r}_B$ is

$$\mathbf{r}_B = r_{Bx}\,\mathbf{ı} + r_{By}\,\mathbf{ȷ} + r_{Bz}\,\mathbf{k}.$$

The z-component of the vector $\mathbf{r}_B$ is

$$r_{Bz} = r_B\,\sin\lambda = b\,\sin\lambda.$$

The x and y components of the vector $\mathbf{r}_B$ are

$$r_{Bx} = r_B\,\cos\lambda\,\cos\nu = b\,\cos\lambda\,\cos\nu,$$
$$r_{By} = -r_B\,\cos\lambda\,\sin\nu = -b\,\cos\lambda\,\sin\nu.$$

First the MATLAB command sym constructs the symbolic variables:

```
a = sym('a','real');
b = sym('b','real');
thetax = sym('thetax','real');
thetay = sym('thetay','real');
thetaz = sym('thetaz','real');
lambda = sym('lambda','real');
nu = sym('nu','real');
```

The command sym('a','real') also assume that a is a real number. The shortcut for constructing symbolic objects is

```
syms a b thetax thetay thetaz lambda nu
```

The position vectors rA_ and rB_ are can be written as:

```
rAx=a*cos(thetax); % x-component
rAy=a*cos(thetay); % y-component
rAz=a*cos(thetaz); % z-component
rA_=[rAx, rAy, rAz];  % rA_ vector

rBz=b*sin(lambda);
rBx=b*cos(lambda)*cos(nu);
rBy=-b*cos(lambda)*sin(nu);
rB_=[rBx, rBy, rBz]; % rB_ vector
```

The projection of the vector rB_ on the *xy*-plane is the vector:

```
rC_=[rBx, rBy, 0];   % rC_ vector
```

(a) The resultant vector **R** is

$$\mathbf{R} = \mathbf{r}_A + \mathbf{r}_B = (r_{Ax} + r_{Bx})\,\mathbf{\imath} + \left(r_{Ay} + r_{By}\right)\mathbf{\jmath} + (r_{Az} + r_{Bz})\,\mathbf{k},$$

and with MATLAB the resultant is:

```
R_=rA_+rB_;
```

The symbolical components of the resultant R_ are printed with:

```
fprintf('Rx = %s \n',char(R_(1)))
fprintf('Ry = %s \n',char(R_(2)))
fprintf('Rz = %s \n',char(R_(3)))
```

The statement fprintf(f,format,s) writes data of array s to the file f. The format is a string in single quotation marks that describes the format of the output fields. Percent sign followed by the s, conversion character, is used for strings. The MATLAB results are displayed as:

```
Rx = a*cos(thetax) + b*cos(lambda)*cos(nu)
```

```
Ry = a*cos(thetay) - b*cos(lambda)*sin(nu)
Rz = a*cos(thetaz) + b*sin(lambda)
```

To calculate the numerical values a list is created with the symbolical variable:

```
lists={a,b,thetax,thetay,thetaz,lambda,nu};
```

A new list with the numerical values for `lists` is introduced:

```
listn={150,200,pi/6,pi/3,pi/3,pi/4,15*pi/180};
% a -> 150
% b -> 200
% thetax -> pi/6
% thetay -> pi/3
% thetaz -> pi/3
% lambda -> pi/4
% nu -> 15*pi/180
```

To calculate numerically the vectors `rA_`, `rB_`, `rC_`, and `R_`, the symbolic variables need to be substituted with the input numerical data.

The statement `subs(expr,lists,listn)` replaces `lists` with `listn` in the symbolic expression `expr`. The numerical values for `rA_`, `rB_`, `rC_`, and `R_` are:

```
rAn_=subs(rA_,lists,listn);
rBn_=subs(rB_,lists,listn);
rCn_=subs(rC_,lists,listn);
Rn_=subs(R_,lists,listn);
```

The numerical values for the vectors are printed with:

```
fprintf('rA_ = [%6.3f %6.3f %6.3f] (m)\n', rAn_)
fprintf('rB_ = [%6.3f %6.3f %6.3f] (m)\n', rBn_)
fprintf('rC_ = [%6.3f %6.3f %6.3f] (m)\n', rCn_)
fprintf('R_ = [%6.3f %6.3f %6.3f] (m)\n', Rn_)
```

and the results are:

```
rA_ = [129.904 75.000 75.000] (m)
rB_ = [136.603 -36.603 141.421] (m)
rC_ = [136.603 -36.603  0.000] (m)
R_ = [266.506 38.397 216.421] (m)
```

The direction angles of **R** are calculated with

$$\alpha_R = \arccos\frac{R_x}{R},\ \beta_R = \arccos\frac{R_y}{R},\ \gamma_R = \arccos\frac{R_z}{R},$$

and in MATLAB:

```
uR_ = Rn_/sqrt(dot(Rn_,Rn_));
```

```
alpha = acosd(uR_(1));
beta  = acosd(uR_(2));
gamma = acosd(uR_(3));
```

The function `acosd` calculates the inverse cosine and the result is in degrees. The general function is `acos(x)`, the arccosine of the element `x`. The numerical results are:

```
uR_ = R_/|R| = [ 0.771  0.111  0.626]
alpha = 39.514 (deg)
beta  = 83.618 (deg)
gamma = 51.209 (deg)
```

(b) The vector (cross) product of the vector $\mathbf{r}_A$ and the vector $\mathbf{r}_B$ is the vector

$$\mathbf{r}_A \times \mathbf{r}_B = \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ r_{Ax} & r_{Ay} & r_{Az} \\ r_{Bx} & r_{By} & r_{Bz} \end{vmatrix},$$

or in MATLAB:

```
AxB_ = cross(rAn_, rBn_);
```

The numerical result is:

```
rA_ x rB_ = [1.34e+04 -8.13e+03 -1.5e+04] (m)
```

The angle, θ, between $\mathbf{r}_A$ and $\mathbf{r}_B$ is calculated from the relation

$$\mathbf{r}_A \cdot \mathbf{r}_B = r_A \, r_B \, \cos\theta,$$

and in MATLAB:

```
mrA=sqrt(dot(rAn_,rAn_));
mrB=sqrt(dot(rBn_,rBn_));

costheta = dot(rAn_, rBn_)/(mrA*mrB);
```

The numerical result is `theta = 40.231 (deg).`

(c) The projection of the vector $\mathbf{r}_A$ on the vector $\mathbf{r}_B$ is calculated from the relation

$$\mathrm{pr}_{\mathbf{r}_A}|_{\mathbf{r}_B} = \mathbf{r}_A \cdot \frac{\mathbf{r}_B}{r_B}.$$

The MATLAB commands are:

```
uRB_ = rBn_/mrB;
prrArB = dot(rAn_, uRB_);
```

and the MATLAB result is:

```
projection of rA_ on rB_ = rA_.rB_/|rB|
pr of rA_ on rB_ = 128.033 (m)
```

(d) The scalar triple product, $\mathbf{r}_C \cdot (\mathbf{r}_B \times \mathbf{r}_A)$, is calculated in MATLAB with:

```
dot(rCn_,cross(rBn_,rAn_))
```

or

```
CAB=[rCn(1),rCn(2),rCn(3);
     rBn(1),rBn(2),rBn(3);
     rAn(1),rAn(2),rAn(3)];
```

and the numerical result is `rC.(rB x rA) = -2.12e+06`.

Next the vectors are plotted using MATLAB. The numerical vectors are introduces with `quiver3(x,y,z,u,v,w)` that represents the vectors as arrows with components `u,v,w` at the points `x,y,z`:

```
quiver3(0,0,0, rAn_(1),rAn_(2),rAn_(3),1,...
    'Color','r','LineWidth',1.5)
quiver3(0,0,0, rBn_(1),rBn_(2),rBn_(3),1,...
    'Color','k','LineWidth',1.5)
quiver3(0,0,0, rCn_(1),rCn_(2),rCn_(3),1,...
    'Color','k','LineWidth',1)
quiver3(0,0,0, Rn_(1),Rn_(2),Rn_(3),1,...
    'Color','b','LineWidth',2)
```

The cartesian axes are plotted with:

```
quiver3(0,0,0,sf,0,0,1,'Color','k','LineWidth',1)
quiver3(0,0,0,0,sf,0,1,'Color','k','LineWidth',1)
quiver3(0,0,0,0,0,sf,1,'Color','k','LineWidth',1)
```

The MATLAB plots are shown in Fig. 1.13.

1.13 Problems

1.1 The force $\mathbf{F}$ shown in the Fig. 1.14 has the vector components $\mathbf{F}_x$, $\mathbf{F}_y$, and $\mathbf{F}_z$ withe the magnitudes F_x, F_y, and F_z respectively. Find the direction angles θ_x, θ_y and θ_z made by the vectorial force $\mathbf{F}$ with the positive x, y, and z axes. For the numerical application use $F_x = 140$ units, $F_y = 170$ units, and $F_z = 190$ units.

1.2 The forces $\mathbf{F}_1$ and $\mathbf{F}_2$ are applied as shown in the Fig. 1.15. The force $\mathbf{F}_2$ has the magnitude F_2 and makes the angle β with the horizontal axis and the force $\mathbf{F}_1$ has the magnitude F_1. The angle between the segment AB and the force F_1 is φ and the angle between BA and the horizontal axis is denoted by θ. Determine

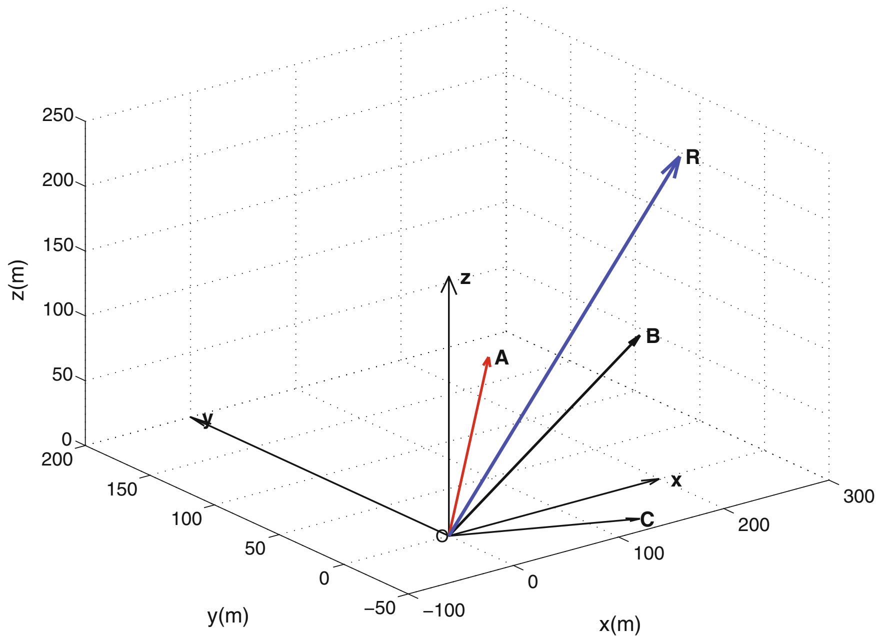
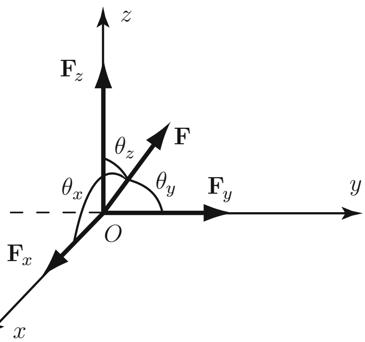


Fig. 1.13 Example 1.3: MATLAB graphical representation

Fig. 1.14 Problem 1.1



the resultant $\mathbf{F} = \mathbf{F}_1 + \mathbf{F}_2$. For the numerical application use $F_1 = 30$ units, $F_2 = 45$ units, $\theta = 105°$, $\varphi = 110°$, and $\beta = 30°$.

1.3 The following vectors are given: $\mathbf{v}_1 = 2\imath - 3\jmath - 7\mathbf{k}$, $\mathbf{v}_2 = 5\imath + 4\mathbf{k}$, and $\mathbf{v}_3 = 2\imath + 9\jmath + 10\mathbf{k}$. Find $(\mathbf{v}_1 \times \mathbf{v}_2) \times \mathbf{v}_3$ and $(\mathbf{v}_1 \times \mathbf{v}_2) \cdot \mathbf{v}_3$.

1.4 Find the angle between the vectors $\mathbf{v}_1 = 7\imath - 8\jmath + 3\mathbf{k}$ and $\mathbf{v}_2 = 2\imath + 5\jmath + 8\mathbf{k}$. Find the expressions $\mathbf{v}_1 \times \mathbf{v}_2$ and $\mathbf{v}_1 \cdot \mathbf{v}_2$.

1.5 The following vectors are given $\mathbf{v}_1 = 3\imath + 5\jmath + 7\mathbf{k}$, $\mathbf{v}_2 = 2\imath + 4\jmath + 5\mathbf{k}$, and $\mathbf{v}_3 = -4\imath - 4\mathbf{k}$. Find the vector triple product of $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$.

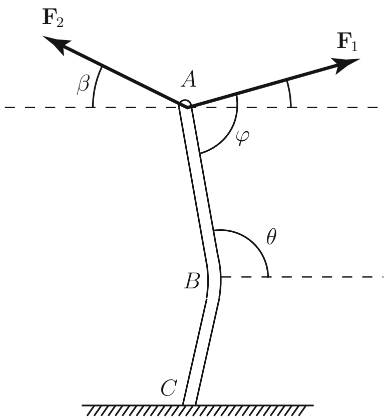


Fig. 1.15 Problem 1.2

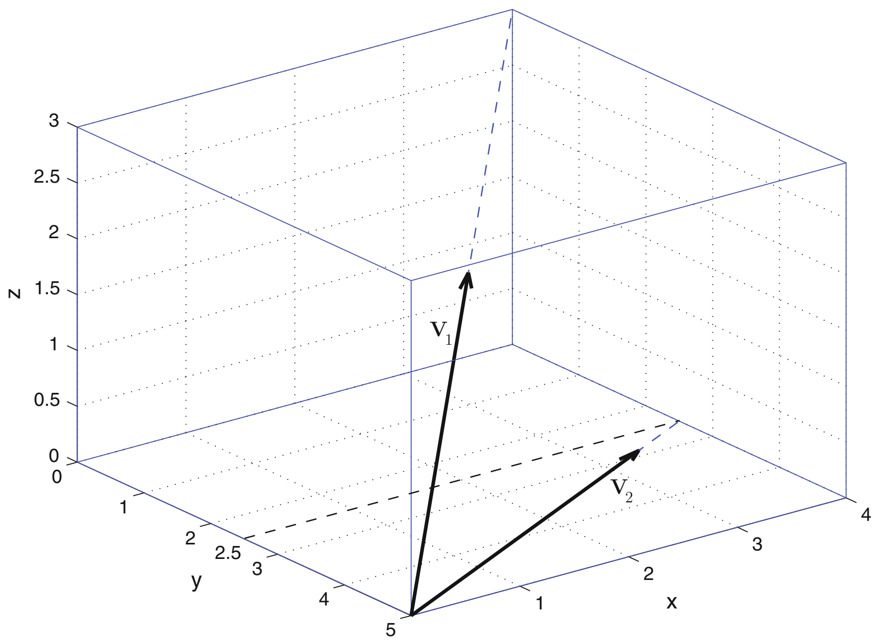


Fig. 1.16 Problem 1.6

1.6 Figure 1.16 represents the vectors $\mathbf{V}_1$ and $\mathbf{V}_2$ acting on a prism. The magnitudes of the vectors are $V_1 = V_2 = 4$ units. (a) Find the resultant and the direction cosines of the resultant. (b) Determine the angle between the vectors $\mathbf{V}_1$ and $\mathbf{V}_3$. (c) Find the projection of the vector $\mathbf{V}_1$ on the resultant vector.

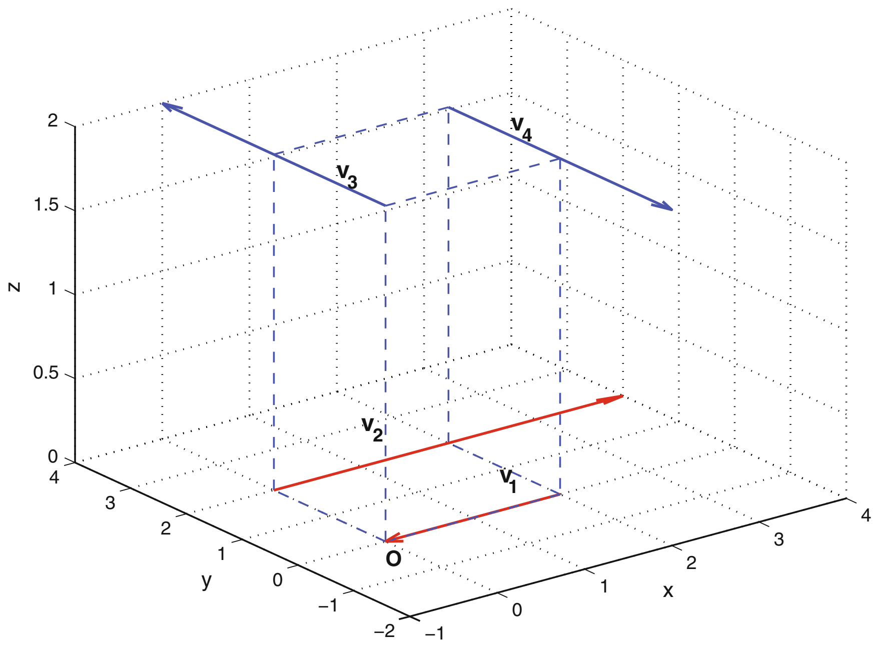


Fig. 1.17 Problem 1.7

1.7 Figure 1.17 represents the vectors $\mathbf{v}_1 = -V\mathbf{\imath}$, $\mathbf{v}_2 = 2\,V\mathbf{\imath}$, $\mathbf{v}_3 = 2\,V\mathbf{\jmath}$, and $\mathbf{v}_4 = -2\,V\mathbf{\jmath}$, where $V = 2$ units. Determine: (a) the resultant $\mathbf{v} = \mathbf{v}_1 + \mathbf{v}_2 + \mathbf{v}_3 + \mathbf{v}_4$; (b) the angle between the vectors $\mathbf{v}_1$ and $\mathbf{v}_3$; (c) the projection of the vector $\mathbf{v}_4$ on the resultant vector; (d) $\mathbf{v}_2 \cdot \mathbf{v}$; $\mathbf{v}_1 \times \mathbf{v}_2$; and $\mathbf{v}_2 \times \mathbf{v}_4$.

1.8 The magnitude of the vectors, shown in Fig. 1.18, are $F_1 = 2$ units, $F_2 = 2.5$ units, $F_3 = 3$ units, and $F_4 = 3.5$ units. (a) Find the resultant $\mathbf{F} = \mathbf{F}_1 + \mathbf{F}_2 + \mathbf{F}_3 + \mathbf{F}_4$. (b) Determine the angle between the vectors $\mathbf{F}_1$ and $\mathbf{F}_3$. (c) Find the projection of the vector $\mathbf{F}_2$ on the vector $\mathbf{F}_4$. (d) Calculate $\mathbf{F}_2 \cdot \mathbf{F}_4$; $\mathbf{F}_1 \cdot (\mathbf{F}_3 \times \mathbf{F}_1)$; $(\mathbf{F}_2 \times \mathbf{F}_3) \cdot \mathbf{F}_1$; and $[\mathbf{F}_1,\ \mathbf{F}_2,\ \mathbf{F}_3]$.

1.14 Programs

1.14.1 Program 1.1

```
% example 1.1
clear all
% clears all the objects in the MATLAB workspace and
% resets the default MuPAD symbolic engine
```

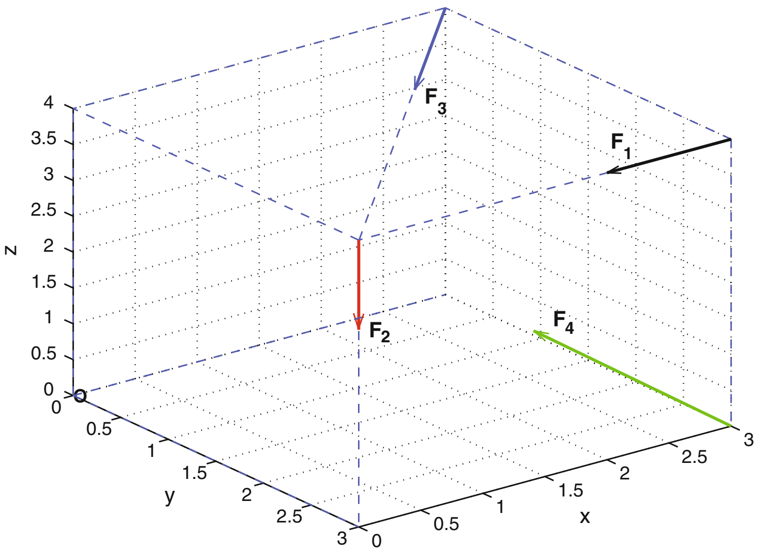


Fig. 1.18 Problem 1.8

```
clc % clears the command window and homes the cursor
close all % closes all the open figure windows

b=4;          % units
c=3;          % units
v(1) =1;      % units magnitude of vector v1_
v(2) =1;      % units magnitude of vector v2_
vy(2)=0.5;    % units magnitude of vector v2y_
v(3) =1.5;    % units magnitude of vector v3_
theta=pi/6;   % rad   angle of v3_ with x-axis

% angle of v1_ with x-axis
m=b/c;
alpha(1)=atan(m);
% atan(m) is arctangent of m in rad
fprintf('alpha1=%6.3f(rad)=%6.3f(deg)\n',...
  alpha(1),alpha(1)*180/pi)

% angle of v2_ with x-axis
phi=asin(vy(2)/v(2));
% asin(x) is arcsine of x
alpha(2)=pi-phi;
fprintf('alpha2=%6.3f(rad)=%6.3f(deg)\n',...
   alpha(2),alpha(2)*180/pi)
```

```
% angle of v3_ with x-axis
alpha(3)=pi+theta;
fprintf('alpha3=%6.3f(rad)=%6.3f(deg)\n',...
   alpha(3),alpha(3)*180/pi)

% for repeat statements a specific number of times
% for variable = expr, statement, ..., statement END
for i = 1:3
    vx(i)=v(i)*cos(alpha(i));
    vy(i)=v(i)*sin(alpha(i));
end
v1_=[vx(1),vy(1),0];
v2_=[vx(2),vy(2),0];
v3_=[vx(3),vy(3),0];

fprintf...
('v1_=[%6.3f,%6.3f,%d](units)\n',v1_)
fprintf...
('v2_=[%6.3f,%6.3f,%d](units)\n',v2_)
 fprintf...
('v3_=[%6.3f,%6.3f,%d](units)\n',v3_)
fprintf('\n')

v_ = v1_+v2_+v3_;

v = sqrt(v_*v_.');
% v_.' transpose of v_
v = norm(v_);
% norm is vector norm
fprintf...
('resultant v_ = v1_+v2_+v3 \n')
fprintf...
('v_=[%6.3f,%6.3f,%d](units)\n',v_)
fprintf...
('v=|v_|=%6.3f(units)\n',v)
beta=atan(v_(2)/v_(1));
fprintf...
('angle of v_ with x-axis\n')
fprintf...
('beta=%6.3f(rad)=%6.3f(deg)\n',...
  beta,beta*180/pi)

% graphic
xlabel('x'), ylabel('y')
```

```
a = 2;
axis([-a a -a a])
grid on
hold on

% quiver(x,y,u,v) plots vectors as arrows with
% components (u,v) at the points (x,y)

for i = 1:3
quiver(0,0,vx(i),vy(i),...
'Color','k','LineWidth',1.5)
end
quiver(0,0,v_(1),v_(2),...
'Color','r','LineWidth',2)

text(vx(1),vy(1),'v_1',...
'fontsize',12,'fontweight','b')
text(vx(2),vy(2),'  v_2',...
'fontsize',12,'fontweight','b')
text(vx(3),vy(3),'v_3',...
'fontsize',12,'fontweight','b')
text(v_(1),v_(2),'v',...
'fontsize',12,'fontweight','b')

quiver(0,0,a,0,...
'Color','b','LineWidth',1.0)
text(a,0,'x',...
'fontsize',12,'fontweight','b')
quiver(0,0,0,a,...
'Color','b','LineWidth',1.0)
text(0,a,'y',...
'fontsize',12,'fontweight','b')

% end of program
```

1.14.2 Program 1.2

```
% example 1.2
clear all; clc; close all

F = 500; % kN
OM = 50; % m
xA = 20; % m
```

```
yA = 10; % m
zA = 30; % m
alpha = 45; % deg

rA_ = [xA, yA, zA];

xM =  OM*cosd(alpha);
yM =  OM*sind(alpha);
zM =  0;
rM_ = [xM, yM, zM];

rAM_ = rM_-rA_;
rAM = norm(rAM_);

uAM_ = rAM_/rAM;

F_ = F*uAM_;
thetax = acos(F_(1)/F); % alpha
thetay = acos(F_(2)/F); % beta
thetaz = acos(F_(3)/F); % gamma

fprintf('rA_=[%6.3f %6.3f %6.3f] (m)\n',rA_)
fprintf('rM_=[%6.3f %6.3f %6.3f] (m)\n',rM_)
fprintf('rAM_=[%6.3f %6.3f %6.3f] (m)\n',rAM_)
fprintf('uAM_=[%6.3f %6.3f %6.3f] (m)\n',uAM_)
fprintf('\n')
fprintf('F_=[%6.3f %6.3f %6.3f] (kN)\n',F_)
fprintf('\n')
fprintf...
('thetax=%6.3f(rad)=%6.3f(deg)\n',...
  thetax,thetax*180/pi)
fprintf...
('thetay=%6.3f(rad)=%6.3f(deg)\n',...
  thetay,thetay*180/pi)
fprintf...
('thetaz=%6.3f(rad)=%6.3f(deg)\n',...
  thetaz,thetaz*180/pi)

% graphic

xlabel('x(m)'), ylabel('y(m)'), zlabel('z(m)')

text(0,0,0,' O','HorizontalAlignment','right')

sf=30;
```

```
axis([-sf sf -sf sf -sf sf])
% axis(([xMIN xMAX yMIN yMAX zMIN zMAX])
% set scaling for the x,y,z axes

grid on
% grid on adds major grid lines

hold on
% hold on locks up the current plot
% and all axis properties so that
% following graphing commands add
% to the existing graph

 axis auto
% axis auto returns the axis scaling to
% its default automatic mode

% quiver3(x,y,z,u,v,w) represents vectors as arrows
% with components(u,v,w) at the points (x,y,z)

quiver3(0,0,0, xA,yA,zA,1,...
    'Color','b','LineWidth',1.5)
quiver3(0,0,0, xM,yM,zM,1,...
    'Color','k','LineWidth',1.5)

text(xA,yA,zA,' A',...
'fontsize',12,'fontweight','b')
text(xM,yM,zM,' M',...
'fontsize',12,'fontweight','b')

ff=0.1; % force scale factor
quiver3(xM,yM,zM,...
    ff*F_(1),ff*F_(2),ff*F_(3),1,...
    'Color','r','LineWidth',2)

text(...
    xM+ff*F_(1),...
    yM+ff*F_(2),...
    zM+ff*F_(3),'    F',...
'fontsize',12,'fontweight','b')

line([xA xM],[yA yM],[zA zM],'LineStyle','--')

% cartesian axes
quiver3(0,0,0,sf,0,0,1,'Color','k','LineWidth',1)
```

```
quiver3(0,0,0,0,sf,0,1,'Color','k','LineWidth',1)
quiver3(0,0,0,0,0,sf,1,'Color','k','LineWidth',1)

text(sf,0,0,'  x','fontsize',12,'fontweight','b')
text(0,sf,0,'  y','fontsize',12,'fontweight','b')
text(0,0,sf,'  z','fontsize',12,'fontweight','b')

% end of program
```

1.14.3 Program 1.3

```
% example 1.3
clear all; clc; close all

% symbolic input data
a = sym('a','real');
b = sym('b','real');
thetax = sym('thetax','real');
thetay = sym('thetay','real');
thetaz = sym('thetaz','real');
lambda = sym('lambda','real');
nu = sym('nu','real');

rAx=a*cos(thetax); % x-component
rAy=a*cos(thetay); % y-component
rAz=a*cos(thetaz); % z-component
rA_=[rAx, rAy, rAz];  % rA_ vector

rBz=b*sin(lambda);
rBx=b*cos(lambda)*cos(nu);
rBy=-b*cos(lambda)*sin(nu);
rB_=[rBx, rBy, rBz]; % rB_ vector

rC_=[rBx, rBy, 0];   % rC_ vector

% a)
R_=rA_+rB_;

% symbolical components of R_
fprintf('Rx = %s \n',char(R_(1)))
fprintf('Ry = %s \n',char(R_(2)))
fprintf('Rz = %s \n',char(R_(3)))
fprintf('\n')
```

```
lists={a,b,thetax,thetay,thetaz,lambda,nu};
% numbers for lists
listn={150,200,pi/6,pi/3,pi/3,pi/4,15*pi/180};
% a -> 150
% b -> 200
% thetax -> pi/6
% thetay -> pi/3
% thetaz -> pi/3
% lambda -> pi/4
% nu -> 15*pi/180

rAn_=subs(rA_,lists,listn);
rBn_=subs(rB_,lists,listn);
rCn_=subs(rC_,lists,listn);
Rn_=subs(R_,lists,listn);

fprintf('rA_ = [%6.3f %6.3f %6.3f] (m)\n', rAn_)
fprintf('rB_ = [%6.3f %6.3f %6.3f] (m)\n', rBn_)
fprintf('rC_ = [%6.3f %6.3f %6.3f] (m)\n', rCn_)
fprintf('R_ = [%6.3f %6.3f %6.3f] (m)\n', Rn_)
fprintf('\n')

uR_ = Rn_/sqrt(dot(Rn_,Rn_));
alpha = acosd(uR_(1));
beta  = acosd(uR_(2));
gamma = acosd(uR_(3));

fprintf...
('uR_ = R_/|R| = [%6.3f %6.3f %6.3f]\n', uR_)
fprintf('alpha = %6.3f (deg)\n', alpha)
fprintf('beta  = %6.3f (deg)\n', beta)
fprintf('gamma = %6.3f (deg)\n', gamma)
fprintf('\n')

% b)
AxB_ = cross(rAn_, rBn_);
fprintf ...
('rA_ x rB_ = [%6.3g %6.3g %6.3g] (m)\n', AxB_)

mrA=sqrt(dot(rAn_,rAn_));
mrB=sqrt(dot(rBn_,rBn_));

costheta = dot(rAn_, rBn_)/(mrA*mrB);
fprintf('rA_.rB_ = |rA||rB| cos(theta) \n')
fprintf('theta = %6.3f (deg)\n', acosd(costheta))
```

```
% acos(phi) is the arccosine of the elements of phi
% acosd(phi) is the inverse cosine, expressed in degrees,
% of the elements of phi

% c)
fprintf('\n')
fprintf ...
('projection of rA_ on rB_ = rA_.rB_/|rB|\n')
uRB_ = rBn_/mrB;
prrArB = dot(rAn_, uRB_);
fprintf('pr of rA_ on rB_ = %6.3f (m)\n',prrArB)

% d)
fprintf('\n')
fprintf('rC_.(rB_ x rA_) = %6.3g \n',...
  dot(rCn_,cross(rBn_,rAn_)))

CAB=[rCn_(1),rCn_(2),rCn_(3);
     rBn_(1),rBn_(2),rBn_(3);
     rAn_(1),rAn_(2),rAn_(3)];

fprintf('[rC_;rB_;rA_] = %6.3g \n',det(CAB))

% graphic

xlabel('x(m)'), ylabel('y(m)'), zlabel('z(m)')

text(0,0,0,' O','HorizontalAlignment','right')

sf=200;
axis([-sf sf -sf sf -sf sf])
% axis(([xMIN xMAX yMIN yMAX zMIN zMAX])
% set scaling for the x,y,z axes

grid on
% grid on adds major grid lines

hold on
% hold on locks up the current plot
% and all axis properties so that
% following graphing commands add
% to the existing graph
```

```
 axis auto
% axis auto returns the axis scaling to
% its default automatic mode

% quiver3(x,y,z,u,v,w) represents vectors as arrows
% with components(u,v,w) at the points (x,y,z)

quiver3(0,0,0, rAn_(1),rAn_(2),rAn_(3),1,...
    'Color','r','LineWidth',1.5)
text(rAn_(1),rAn_(2),rAn_(3),' A',...
'fontsize',12,'fontweight','b')

quiver3(0,0,0, rBn_(1),rBn_(2),rBn_(3),1,...
    'Color','k','LineWidth',1.5)
text(rBn_(1),rBn_(2),rBn_(3),' B',...
'fontsize',12,'fontweight','b')

% cartesian axes
quiver3(0,0,0,sf,0,0,1,'Color','k','LineWidth',1)
quiver3(0,0,0,0,sf,0,1,'Color','k','LineWidth',1)
quiver3(0,0,0,0,0,sf,1,'Color','k','LineWidth',1)

text(sf,0,0,'  x','fontsize',12,'fontweight','b')
text(0,sf,0,'  y','fontsize',12,'fontweight','b')
text(0,0,sf,'  z','fontsize',12,'fontweight','b')

% end of program
```

References

1. P. Appell, *Traité de mécanique rationnelle* (Gauthier-Villars, Paris, 1955)
2. M. Atanasiu, *Mechanics* (EDP, Bucharest, 1973)
3. H. Baruh, *Analytical Dynamics* (WCB/McGraw-Hill, Boston, 1999)
4. F.P. Beer, E.R. Johnston, *Vector Mechanics for Engineers: Statics and Dynamics 5/e* (McGraw-Hill Publishing Company, New York, 1988)
5. F.P. Beer, E.R. Johnston, D.F. Mazurek, *Vector Mechanics for Engineers: Statics, 10/e* (McGraw-Hill Publishing Company, New York, 2012)
6. A.M. Bedford, W. Fowler, K.M. Liechti, *Statics and Mechanics of Materials* (Prentice Hall, Inc., Upper Saddle River, 2002)
7. A.M. Bedford, W. Fowler, *Engineering Mechanics: Statics, 5/e* (Prentice Hall, Inc., Upper Saddle River, 2007)
8. A.P. Boresi, R.J. Schmidt, *Engineering Mechanics: Statics* (PWS Publishing Company, Boston, 2000)
9. M.I. Buculei, *Mechanics* (University of Craiova Press, Craiova, 1974)
10. M.I. Buculei, D. Bagnaru, G. Nanu, D.B. Marghitu, *Analysis of Mechanisms with Bars* (Scrisul romanesc, Craiova, 1986)

11. I. Stroe et al., *Analytical Mechanics Problems* (University Politehnica of Bucharest, Bucharest, 1997)
12. V. Ceausu, N. Enescu, F. Ceausu, *Mechanics Problems* (Printech, Bucharest, 1999)
13. S.J. Chapman, *MATLAB Programming for Engineers* (Thomson Learning, Pacific Grove, CA, 2002)
14. S. Attaway, *MATLAB: A Practical Introduction to Programming and Problem Solving* (Butterworth-Heinemann, Elsevier, Amsterdam, 2012)
15. D.M. Etter, D.C. Kuncicky, *Introduction to MATLAB for Engineers and Scientists* (Prentice Hall, Inc., Upper Saddle River, 1996)
16. C. Iacob, *Theoretical Mechanics* (EDP, Bucharest, 1980)
17. J.H. Ginsberg, *Advanced Engineering Dynamics* (Cambridge University Press, Cambridge, 1995)
18. D.T. Greenwood, *Principles of Dynamics* (Prentice-Hall, Inc., Englewood Cliffs, 1998)
19. L.E. Goodman, W.H. Warner, *Statics* (Dover Publications Inc., New York, 2001)
20. R.C. Hibbeler, *Engineering Mechanics: Statics and Dynamics 13/e* (Prentice-Hall, Inc., Upper Saddle River, 2013)
21. T.R. Kane, *Analytical Elements of Mechanics*, vol. 1 (Academic Press, New York, 1959)
22. T.R. Kane, *Analytical Elements of Mechanics*, vol. 2 (Academic Press, New York, 1961)
23. T.R. Kane, D.A. Levinson, *Dynamics* (McGraw-Hill, New York, 1985)
24. R. Maeder, *Programming in Mathematica* (Addison-Wesley Publishing Company, Redwood City, 1990)
25. N.H. Madsen, *Statics and Dynamics, class notes*, http://www.eng.auburn.edu/users/nmadsen/
26. D.B. Marghitu, *Mechanical Engineer's Handbook* (Academic Press, San Diego, 2001)
27. D.B. Marghitu, M.J. Crocker, *Analytical Elements of Mechanisms* (Cambridge University Press, Cambridge, 2001)
28. D.B. Marghitu, *Kinematic Chains and Machine Component Design* (Elsevier, Amsterdam, 2005)
29. D.B. Marghitu, *Mechanisms and Robots Analysis with MATLAB* (Springer, London, 2009)
30. D.B. Marghitu, M. Dupac, *Advanced Dynamics: Analytical and Numerical Calculations with MATLAB* (Springer, New York, 2012)
31. D.B. Marghitu, *Statics and Dynamics, class notes*, http://www.eng.auburn.edu/users/marghitu/
32. D.J. McGill, W.W. King, *Engineering Mechanics: Statics and an Introduction to Dynamics* (PWS Publishing Company, Boston, 1995)
33. J.L. Meriam, L.G. Kraige, *Engineering Mechanics: Statics, 7/e* (Wiley, New York, 2011)
34. R.L. Mott, *Machine Elements in Mechanical Design* (Prentice Hall, Upper Saddle River, 1999)
35. R.L. Norton, *Machine Design* (Prentice-Hall, Upper Saddle River, 1996)
36. M. Plesha, G. Gray, F. Costanzo, *Engineering Mechanics: Statics, 2/e* (McGraw-Hill Publishing Company, New York, 2012)
37. M. Radoi, E. Deciu, *Mechanics* (EDP, Bucharest, 1981)
38. W.F. Riley, L.D. Sturges, *Engineering Mechanics: Statics, 2/e* (Wiley, New York, 1995)
39. A. Ruina, R. Pratap, *Introduction to Statics and Dynamics* (Oxford University Press, Oxford, 2002)
40. A. Ripianu, P. Popescu, B. Balan, *Technical Mechanics* (EDP, Bucharest, 1979)
41. I.H. Shames, *Engineering Mechanics Statics, 4/e* (Prentice Hall, Inc., New Delhi, 1996)
42. S.D. Sheppard, B.H. Tongue, *Statics: Analysis and Design of Systems in Equilibrium* (Wiley, New York, 2005)
43. D. Smith, *Engineering Computation with MATLAB* (Pearson Education, Upper Saddle River, 2008)
44. R.W. Soutas-Little, D.J. Inman, *Engineering Mechanics: Statics and Dynamics* (Prentice-Hall, Upper Saddle River, 1999)
45. R.W. Soutas-Little, D.J. Inman, D. Balint, *Engineering Mechanics: Statics* (Cengage Learning, New Delhi, 2007)
46. S. Staicu, *Theoretical Mechanics* (EDP, Bucharest, 1998)
47. A. Stan, M. Grumazescu, *Mechanics Problems* (EDP, Bucharest, 1973)

48. J. Sticklen, M.T. Eskil, *An Introduction to Technical Problem Solving with MATLAB* (Great Lakes Press, Wildwood, 2006)
49. A. Stoenescu, G. Silas, *Theoretical Mechanics* (ET, Bucharest, 1957)
50. J.H. Jackson, H.G. Wirtz, *Schaum's Outline of Theory and Problems of Statics and Strength of Materials* (McGraw-Hill Publishing Company, New York, 1983)
51. The MathWorks: https://www.mathworks.com/
52. Statics eBook : https://ecourses.ou.edu/
53. R. Voinea, D. Voiculescu, V. Ceausu, *Mechanics* (EDP, Bucharest, 1983)
54. V. Valcovici, S. Balan, R. Voinea, *Theoretical Mechanics* (ET, Bucharest, 1959)
55. K.J. Waldron, G.L. Kinzel, *Kinematics, Dynamics, and Design of Machinery* (Wiley, New York, 1999)
56. H.B. Wilson, L.H. Turcotte, D. Halpern, *Advanced Mathematics and Mechanics Applications Using MATLAB* (Chapman & Hall/CRC, Boca Raton, 2003)
57. J.H. Williams Jr., *Fundamentals of Applied Dynamics* (Wiley, New York, 1996)
58. S. Wolfram, *Mathematica* (Wolfram Media/Cambridge University Press, Cambridge, 1999)

Chapter 2
Moments, Couples, Equipollent Systems

2.1 Moment of a Vector About a Point

The moment of a vector $\mathbf{v}$, whose line of action passes through a point B, about a point A is the vector

$$\mathbf{M}_A^{\mathbf{v}} = \mathbf{r}_{AB} \times \mathbf{v}, \tag{2.1}$$

where $\mathbf{r}_{AB}$ is the position vector of B relative to A, and B is any point of line of action, Δ, of the vector $\mathbf{v}$ (Fig. 2.1). The moment vector $\mathbf{M}_A^{\mathbf{v}} = \mathbf{0}$ if and only if the line of action of $\mathbf{v}$ passes through A or $\mathbf{v} = \mathbf{0}$. The magnitude of $\mathbf{M}_A^{\mathbf{v}}$ is

$$|\mathbf{M}_A^{\mathbf{v}}| = M_A^{\mathbf{v}} = |\mathbf{r}_{AB}|\,|\mathbf{v}|\,\sin\theta = r_{AB}\,v\,\sin\theta,$$

where θ is the angle between $\mathbf{r}_{AB}$ and $\mathbf{v}$. The perpendicular distance from A to the line of action of $\mathbf{v}$ is

$$d = |\mathbf{r}_{AB}|\,\sin\theta = r_{AB}\,\sin\theta.$$

The moment vector is zero if the vectors $\mathbf{v}$ and $\mathbf{r}_{AB}$ are parallel. The magnitude of the $\mathbf{M}_A^{\mathbf{v}}$ is

$$|\mathbf{M}_A^{\mathbf{v}}| = M_A^{\mathbf{v}} = |\mathbf{v}|\,d = v\,d.$$

The moment vector $\mathbf{M}_A^{\mathbf{v}}$ is perpendicular to both $\mathbf{r}_{AB}$ and $\mathbf{v}$: $\mathbf{M}_A^{\mathbf{v}} \perp \mathbf{r}_{AB}$ and $\mathbf{M}_A^{\mathbf{v}} \perp \mathbf{v}$. If the moment vector is non-zero then it is perpendicular to the plane defined by the distinct directions of $\mathbf{r}_{AB}$ and $\mathbf{v}$. The moment given by Eq. (2.1) does not depend upon the choice of point on the line of action of $\mathbf{v}$. Instead of using the point B, the point B', $B' \in \Delta$ (Fig. 2.1), can be used. The position vector of B relative to A is $\mathbf{r}_{AB} = \mathbf{r}_{AB'} + \mathbf{r}_{B'B}$ where the vector $\mathbf{r}_{B'B}$ is parallel to $\mathbf{v}$, $\mathbf{r}_{B'B}||\mathbf{v}$. Therefore,

D. B. Marghitu et al., *Statics with MATLAB®*,
DOI: 10.1007/978-1-4471-5110-4_2,

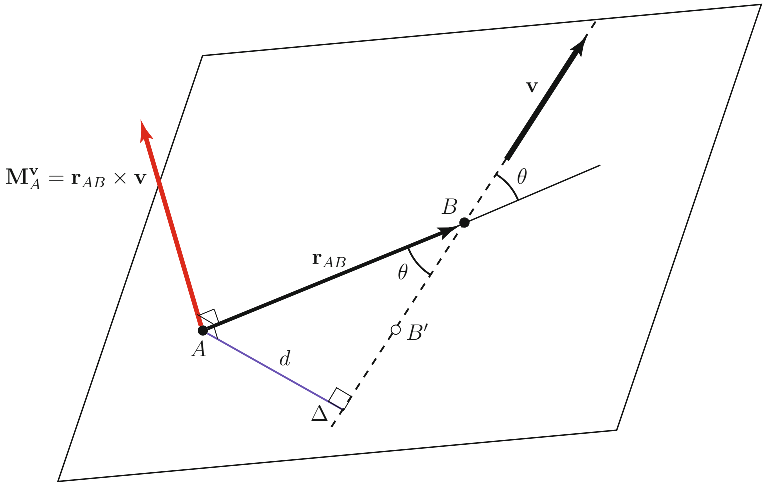


Fig. 2.1 Moment of a vector **v** about a point *A*

$$\mathbf{M}_A^{\mathbf{v}} = \mathbf{r}_{AB} \times \mathbf{v} = (\mathbf{r}_{AB'} + \mathbf{r}_{B'B}) \times \mathbf{v} = \mathbf{r}_{AB'} \times \mathbf{v} + \mathbf{r}_{B'B} \times \mathbf{v} = \mathbf{r}_{AB'} \times \mathbf{v}, \quad (2.2)$$

because $\mathbf{r}_{B'B} \times \mathbf{v} = \mathbf{0}$. The moment about a point is a vector in a particular direction. This moment vector is a sliding vector along that direction.

Next, using MATLAB®, the validity of Eq. (2.2) is shown. Three points A, B, and C are defined by three symbolic position vectors `rA_`, `rB_`, and `rC_`:

```
syms xA yA zA xB yB zB xC yC zC real
rA_ = [xA yA zA];
rB_ = [xB yB zB];
rC_ = [xC yC zC];
rBC_ = rC_ - rB_;
```

The vector **v** is selected as $\mathbf{v} = \mathbf{r}_C - \mathbf{r}_B$, or in MATLAB:

```
v_ = rC_ - rB_;
```

The line of action of the vector **v** is defined as should be by the line segment BC. A generic point B' (in MATLAB `Bp`) divides the line segment joining two given points B and C in a given ratio. The position vector of the point `Bp` is `rBp_`:

```
syms k real % k is a given real number
rBp_ = rB_ + k*(rC_-rB_);
```

The moment of the vector **v** with respect to A is calculated as $\mathbf{r}_{AB} \times \mathbf{v}$, $\mathbf{r}_{AB'} \times \mathbf{v}$, and $\mathbf{r}_{AC} \times \mathbf{v}$, or with MATLAB:

```
MB_ = cross(rB_-rA_, v_);      % rAB_ x v_
MBp_ = cross(rBp_-rA_, v_);    % rABp_ x v_
MC_ = cross(rC_-rA_, v_);      % rAC_ x v_
```

To prove that $\mathbf{M}_A^{\mathbf{v}} = \mathbf{r}_{AB} \times \mathbf{v} = \mathbf{r}_{AB'} \times \mathbf{v} = \mathbf{r}_{AC} \times \mathbf{v}$ the following MATLAB commands are used:

```
% rAB_ x v_ = rABp_ x v_ = rAC_ x v_
fprintf('1=TRUE 0=FALSE\n')
T1=expand(MB_) == expand(MBp_);
fprintf('rAB_ x v_ == rABp_ x v_ => [%d %d %d]\n',T1)
T2=expand(MB_) == expand(MC_);
fprintf('rAB_ x v_ == rAC_ x v_ => [%d %d %d]\n',T2)
```

If `T1 = [1 1 1]` then `rAB_ x v_ == rABp_ x v_` is true and if `T2 = [1 1 1]` then `rAB_ x v_ == rAC_ x v_` is true.

As an example consider the vectors $\mathbf{r}_{AB}$, $\mathbf{r}_{AB'}$, $\mathbf{r}_{AC}$, $\mathbf{v}$ and $\mathbf{M}_A^{\mathbf{v}}$ where the following numerical data are used:$x_A = y_A = z_A = 0$, $x_B = 1$, $y_B = 2$, $z_B = 0$, $x_C = 3$, $y_C = 3$, $z_C = 0$, and $k = 0.75$. The numerical values for the vectors `rA_`, `rB_`, `rC_`, `rBp_`, `v_`, `MB_`, `MBp_`, and `MC_` are calculated in MATLAB with:

```
% A = O origin
slist={xA,yA,zA,...
       xB,yB,zB,...
       xC,yC,zC,k};
nlist={0,0,0,1,2,0,3,3,0,.75};

rB_ = subs(rB_,slist,nlist);
rC_ = subs(rC_,slist,nlist);
rBp_ = subs(rBp_,slist,nlist);
V_ = subs(v_,slist,nlist);

MB_ = subs(MB_,slist,nlist);
MBp_ = subs(MBp_,slist,nlist);
MC_ = subs(MC_,slist,nlist);
```

The numerical values are:

```
rB_ = [1.0 2.0 0]
rC_ = [3.0 3.0 0]
rBp_ = [2.5 2.8 0]
V_ = [2.0 1.0 0]

MB_ = [0 0 -3]
MBp_ = [0 0 -3]
MC_ = [0 0 -3]
```

The MATLAB commands for the current axes and for the Cartesian reference with the origin at A are:

```
a=3;
axis([0 a  0 a  -a a])
```

```
grid on, hold on
% Cartesian axes A=O origin
quiver3(0,0,0,a-.5,0,0,1, ...
    'Color','k','LineWidth',1)
text('Interpreter','latex','String','  $x$',...
    'Position',[a-.5,0,0],'FontSize',12)
quiver3(0,0,0,0,a-.5,0,1, ...
    'Color','k','LineWidth',1)
text('Interpreter','latex','String','  $y$',...
    'Position',[0,a-.5,0],'FontSize',12)
quiver3(0,0,0,0,0,a-.5,1, ...
    'Color','k','LineWidth',1)
text('Interpreter','latex','String','  $z$',...
    'Position',[0,0,a-.5],'FontSize',12)
```

The fonts for the labels x, y, and z are LaTex fonts. The vectors rB_, rC_, rBp_, V_, and the line BC are plotted with:

```
quiver3(0,0,0, rB_(1),rB_(2),rB_(3),1,...
    'Color','k','LineWidth',1)
quiver3(0,0,0, rC_(1),rC_(2),rC_(3),1,...
    'Color','k','LineWidth',1)
quiver3(0,0,0, rBp_(1),rBp_(2),rBp_(3),1,...
    'Color','k','LineWidth',1)
quiver3(rB_(1),rB_(2),rB_(3), V_(1),V_(2),V_(3),1,...
    'Color','g','LineWidth',1)
line...
 ([rB_(1) rC_(1)],[rB_(2) rC_(2)],[rB_(3) rC_(3)],...
    'LineStyle','--','LineWidth',2)
```

The vectors MB_, MBp_, and MC_ are plotted with:

```
quiver3(0,0,0, MB_(1),MB_(2),MB_(3),1,...
    'Color','r','LineWidth',2)
quiver3(0,0,0, MBp_(1),MBp_(2),MBp_(3),1,...
    'Color','g','LineWidth',2)
quiver3(0,0,0, MC_(1),MC_(2),MC_(3),1,...
    'Color','r','LineWidth',2)
```

The labels for the vectors are printed with

```
text('Interpreter','latex','String','  $A=O$',...
text('Interpreter','latex','String','  $A=O$',...
    'Position',[0,0,0],'FontSize',12)
text('Interpreter','latex','String','  $B$',...
    'Position',[rB_(1),rB_(2),rB_(3)],'FontSize',12)
text('Interpreter','latex','String',...
```

```
'$B^\prime$','Position',[rBp_(1),rBp_(2),rBp_(3)],...
'FontSize',12)
text('Interpreter','latex','String','  $C$',...
    'Position',[rC_(1),rC_(2),rC_(3)],'FontSize',12)
text('Interpreter','latex','String',...
'  ${\bf M}_A^{\bf v}$','Position',...
[MB_(1),MB_(2),MB_(3)+.5],'FontSize',12)
```

The MATLAB representation of the vectors is shown in Fig. 2.2.

Moment of a Vector About a Line

The moment $\mathbf{M}_{\Omega}^{\mathbf{v}}$ of a vector $\mathbf{v}$ about a line Ω is the Ω resolute (Ω component) of the moment $\mathbf{v}$ about any point on Ω as shown in Fig. 2.3a. The moment of the vector $\mathbf{v}$ about the line Ω is

$$\mathbf{M}_{\Omega}^{\mathbf{v}} = \mathbf{n}\cdot\mathbf{M}_A^{\mathbf{v}}\,\mathbf{n} = \mathbf{n}\cdot(\mathbf{r}\times\mathbf{v})\,\mathbf{n} = [\mathbf{n},\mathbf{r},\mathbf{v}]\,\mathbf{n},$$

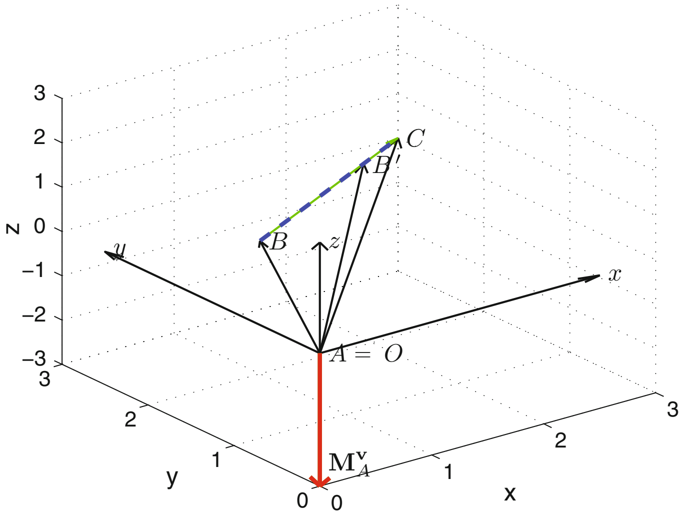


Fig. 2.2 Moment of $\mathbf{v} = \mathbf{r}_{BC}$ about A: $\mathbf{M}_A^{\mathbf{v}} = \mathbf{r}_{AB}\times\mathbf{v} = \mathbf{r}_{AB'}\times\mathbf{v} = \mathbf{r}_{AC}\times\mathbf{v}$

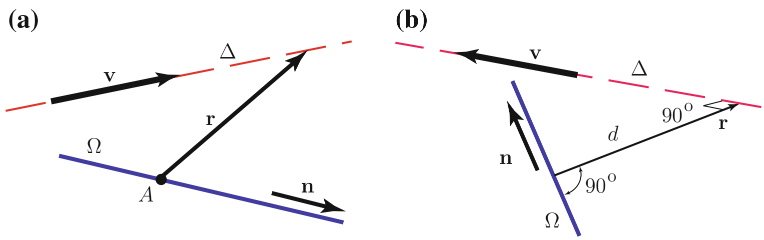


Fig. 2.3 Moment of a vector $\mathbf{v}$ about a line Ω; the line of action of $\mathbf{v}$ does not intersect the line Ω

where $\mathbf{n}$ is a unit vector parallel to Ω, and $\mathbf{r}$ is the position vector of a point on the line of action of $\mathbf{v}$ relative to a point on Ω. The magnitude of $\mathbf{M}_{\Omega}^{\mathbf{v}}$ is given by

$$\left|\mathbf{M}_{\Omega}^{\mathbf{v}}\right| = M_{\Omega}^{\mathbf{v}} = |[\mathbf{n}, \mathbf{r}, \mathbf{v}]|.$$

The moment of a vector about a line is a free vector. If a line Ω is parallel to the line of action Δ of a vector $\mathbf{v}$, then $[\mathbf{n}, \mathbf{r}, \mathbf{v}]\mathbf{n} = \mathbf{0}$ and $\mathbf{M}_{\Omega}^{\mathbf{v}} = \mathbf{0}$. If a line Ω intersects the line of action Δ of $\mathbf{v}$, then $\mathbf{r}$ can be chosen in such a way that $\mathbf{r} = \mathbf{0}$ and $\mathbf{M}_{\Omega}^{\mathbf{v}} = \mathbf{0}$. If a line Ω is perpendicular to the line of action Δ of a vector $\mathbf{v}$, and d is the shortest distance between these two lines, Fig. 2.3b, then

$$\left|\mathbf{M}_{\Omega}^{\mathbf{v}}\right| = |[\mathbf{n}, \mathbf{r}, \mathbf{v}]| = |\mathbf{n}\cdot(\mathbf{r} \times \mathbf{v})| = |\mathbf{n}\cdot(|\mathbf{r}||\mathbf{v}| \sin(\mathbf{r}, \mathbf{v})\mathbf{n})| = |\mathbf{r}||\mathbf{v}| = d|\mathbf{v}|.$$

Moment of a System of Vectors
The moment of a system $\{S\}$ of vectors $\mathbf{v}_i$, $\{S\} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\} = \{\mathbf{v}_i\}_{i=1,2,\ldots,n}$ about a point A is

$$\mathbf{M}_{A}^{\{S\}} = \sum_{i=1}^{n} \mathbf{M}_{A}^{\mathbf{v}_i}.$$

The moment of a system $\{S\}$ of vectors $\mathbf{v}_i$, $\{S\} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\} = \{\mathbf{v}_i\}_{i=1,2,\ldots,n}$ about a line Ω is

$$\mathbf{M}_{\Omega}^{\{S\}} = \sum_{i=1}^{n} \mathbf{M}_{\Omega}^{\mathbf{v}_i}.$$

The moments $\mathbf{M}_{A}^{\{S\}}$ and $\mathbf{M}_{P}^{\{S\}}$ of a system $\{S\}$, $\{S\} = \{\mathbf{v}_i\}_{i=1,2,\ldots,n}$, of vectors, $\mathbf{v}_i$, about two points A and P, are related to each other as follows,

$$\mathbf{M}_{A}^{\{S\}} = \mathbf{M}_{P}^{\{S\}} + \mathbf{r}_{AP} \times \mathbf{R}, \tag{2.3}$$

where $\mathbf{r}_{AP}$ is the position vector of P relative to A, and $\mathbf{R}$ is the resultant of $\{S\}$.

Proof Let B_i a point on the line of action of the vector $\mathbf{v}_i$, $\mathbf{r}_{ABi}$ and $\mathbf{r}_{PBi}$ the position vectors of B_i relative to A and P, Fig. 2.4. Thus,

$$\begin{aligned}\mathbf{M}_{A}^{\{S\}} &= \sum_{i=1}^{n} \mathbf{M}_{A}^{\mathbf{v}_i} = \sum_{i=1}^{n} \mathbf{r}_{ABi} \times \mathbf{v}_i \\ &= \sum_{i=1}^{n} (\mathbf{r}_{AP} + \mathbf{r}_{PBi}) \times \mathbf{v}_i = \sum_{i=1}^{n} (\mathbf{r}_{AP} \times \mathbf{v}_i + \mathbf{r}_{PBi} \times \mathbf{v}_i)\end{aligned}$$

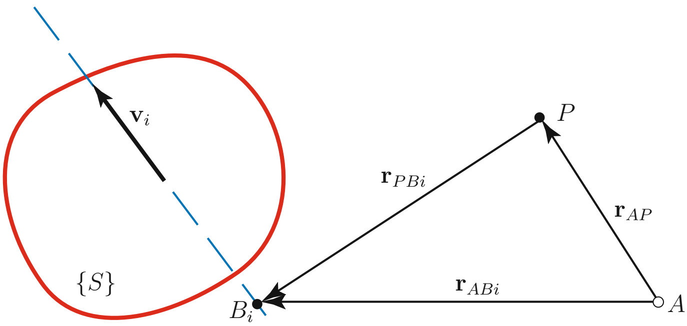


Fig. 2.4 Moments of a system of vectors, $\mathbf{v}_i$ about two points A and P

$$= \sum_{i=1}^{n} \mathbf{r}_{AP} \times \mathbf{v}_i + \sum_{i=1}^{n} \mathbf{r}_{PBi} \times \mathbf{v}_i$$

$$= \mathbf{r}_{AP} \times \sum_{i=1}^{n} \mathbf{v}_i + \sum_{i=1}^{n} \mathbf{r}_{PBi} \times \mathbf{v}_i = \mathbf{r}_{AP} \times \mathbf{R} + \sum_{i=1}^{n} \mathbf{M}_P^{\mathbf{v}_i} = \mathbf{r}_{AP} \times \mathbf{R} + \mathbf{M}_P^{\{S\}}.$$

The proof of Eq. (2.3) for a system of three vectors `v1_`, `v2_`, and `v3_` is given by the following MATLAB commands:

```
% vectors vi_ i=1,2,3
v1_ = sym('[v1x v1y v1z]');
v2_ = sym('[v2x v2y v2z]');
v3_ = sym('[v3x v3y v3z]');

% application points Bi of vi
rB1_ = sym('[xB1 yB1 zB1]');
rB2_ = sym('[xB2 yB2 zB2]');
rB3_ = sym('[xB3 yB3 zB3]');

% any two points A and P
% any two points A and P
rA_ = sym('[xA yA zA]');
rP_ = sym('[xP yP zP]');

rAP_ = rP_-rA_;

rPB1_ = rB1_-rP_;
rPB2_ = rB2_-rP_;
rPB3_ = rB3_-rP_;
```

```
rAB1_ = rAP_+rPB1_;
rAB2_ = rAP_+rPB2_;
rAB3_ = rAP_+rPB3_;

R_ = v1_+v2_+v3_;

% MA_ = sum(ABi_ x vi_) i=1,2,3
MA_ = cross(rAB1_,v1_)+...
      cross(rAB2_,v2_)+...
      cross(rAB3_,v3_);

% MP_ = sum(PBi_ x vi_) i=1,2,3
MP_ = cross(rB1_-rP_,v1_)+...
      cross(rB2_-rP_,v2_)+...
      cross(rB3_-rP_,v3_);

% MA_ = AP_ x R_ + MP_
T1=expand(MA_) == ....
expand(cross(rP_-rA_,R_)+MP_);
fprintf('MA_ == AP_ x R_ + MP_ => [%d %d %d]\n',T1)
fprintf('1=TRUE 0=FALSE\n')
```

The scalar product of the moments $\mathbf{M}_A^{\{S\}}$ and $\mathbf{M}_P^{\{S\}}$, about any points A and P, with the resultant $\mathbf{R}$ of $\{S\}$ are constant

$$\mathbf{M}_A^{\{S\}} \cdot \mathbf{R} = \mathbf{M}_P^{\{S\}} \cdot \mathbf{R}. \tag{2.4}$$

The scalar product $\mathbf{M}_A^{\{S\}} \cdot \mathbf{R}$ is an invariant of the system $\{S\}$. Taking into account the previous MATLAB program the proof for Eq. (2.1) is given below:

```
T2 = expand(MA_*R_.') == expand(MP_*R_.');
fprintf('MA_*R_ == MP_*R_ => %d \n',T2)
```

This invariant is the scalar invariant or the second invariant of the system of vectors. The resultant vector of the system is the vector invariant of that system or the first invariant. The resultant moment $\mathbf{M}_O^{\{S\}}$ with respect to a point O is is not an invariant of the system. The resolution of the moment vector into two components is

$$\mathbf{M}_O^{\{S\}} = \mathbf{M}_R + \mathbf{M}_N,$$

where $\mathbf{M}_R$ is the component along the resultant $\mathbf{R}$ direction and $\mathbf{M}_N$ is perpendicular to the resultant direction. The magnitude of the component along the resultant direction is

$$M_R = \mathbf{M}_O^{\{S\}} \cdot \mathbf{u}_R = \mathbf{M}_O^{\{S\}} \cdot \frac{\mathbf{R}}{R} = \frac{\mathbf{M}_O^{\{S\}} \cdot \mathbf{R}}{R}.$$

The projection of the resultant moment on the resultant of the system, M_R, is an invariant of the system. For the minimum value of the component M_R a corresponding minimum moment, $\mathbf{M}_{min}$, can be defined. The minimum moment is obtained when the normal component is zero, $\mathbf{M}_N = \mathbf{0}$. The minimum moment $\mathbf{M}_{min}$ is given by

$$\mathbf{M}_{min} = \frac{\mathbf{R} \cdot \mathbf{M}_O^{\{S\}}}{\mathbf{R} \cdot \mathbf{R}} \mathbf{R}.$$

If the resultant $\mathbf{R}$ of a system $\{S\}$ of vectors is not equal to zero, $\mathbf{R} \neq \mathbf{0}$, the points about which $\{S\}$ has a minimum moment $\mathbf{M}_{min}$ are on a line called *central axis*, (CA), of $\{S\}$, which is parallel to $\mathbf{R}$ and passes through a point P. The position vector $\mathbf{r}$ of point P relative to an arbitrarily selected reference point O is given by

$$\mathbf{r} = \frac{\mathbf{R} \times \mathbf{M}_O^{\{S\}}}{\mathbf{R} \cdot \mathbf{R}}.$$

The equation of the central axis is obtained from the following program:

```
% resultant force
R_  = sym('[Rx Ry Rz]');
% resultant moment
MO_ = sym('[MOx MOy MOz]');

rA_ = sym('[xA yA zA]');
% O(0,0,0) is the origin
% MA_ = MO_ + AO_ x R_
MA_ = MO_ + cross(-rA_,R_);

% colinearity condition between R_ and MO_
% MA_ = lambda*R_
syms lambda real
eq_ = MA_ - lambda*R_;

% solve for lambda
eqx=solve(eq_(1),'lambda');
eqy=solve(eq_(2),'lambda');
eqz=solve(eq_(3),'lambda');
```

and it results:

```
equation for central axis
(MOx - Rz*yA + Ry*zA)/Rx=
(MOy + Rz*xA - Rx*zA)/Ry=
(MOz - Ry*xA + Rx*yA)/Rz.
```

2.2 Couples

A *couple* is a system of vectors whose resultant is equal to zero and whose moment about some point is not equal to zero. A couple consisting of only two vectors is called a *simple couple*. The vectors of a simple couple have equal magnitudes, parallel lines of action, and opposite senses. The term "couple" can be used to denote the simple couple. In many textbooks the use of the term couple is restricted to the situation in which the contributing vectors are forces and the moment of a couple about a point is called the *torque* of the couple, and is usually denoted by **M** or **T**. The moment of a couple about one point is equal to the moment of the couple about any other point. The moment of a couple is independent of the specific point. The moment of a couple is a free vector.

The torques are vectors and the magnitude of a torque of a simple couple is given by

$$|\mathbf{M}| = d\,|\mathbf{v}| = d\,v,$$

where d is the distance between the lines of action of the two vectors comprising the couple, and **v** is one of these vectors.

Proof In Fig. 2.5, the moment **M** is the sum of the moments of **v** and $-$**v** about any point. The moments about point A are

$$\mathbf{M} = \mathbf{M}_A^{\mathbf{v}} + \mathbf{M}_A^{-\mathbf{v}} = \mathbf{r} \times \mathbf{v} + \mathbf{0}.$$

Hence,

$$|\mathbf{M}| = |\mathbf{r} \times \mathbf{v}| = |\mathbf{r}||\mathbf{v}|\sin(\mathbf{r}, \mathbf{v}) = d|\mathbf{v}|.$$

The direction of the moment of a simple couple can be determined by inspection: **M** is perpendicular to the plane determined by the lines of action of the two vectors comprising the couple, and the sense of **M** is the same as that of $\mathbf{r} \times \mathbf{v}$. The moment of a couple about a line Ω is equal to the Ω resolute of the torque of the couple.

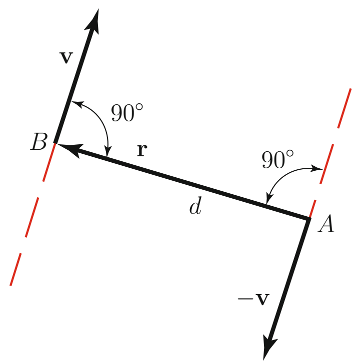


Fig. 2.5 Couple of the vectors **v** and $-$**v**, simple couple

2.3 Force Vectors

Force is a vector quantity, having both magnitude and direction. Force is commonly explained in terms of Newton's three laws of motion in *Principia Mathematica*, 1687. Newton's first principle: a body that is at rest or moving at a uniform rate in a straight line will remain in that state until some force is applied to it. Newton's second law of motion: a particle acted on by forces whose resultant is not zero will move in such a way that the time rate of change of its momentum will at any instant be proportional to the resultant force. Newton's third law: when one body exerts a force on another body, the second body exerts an equal force in magnitude, opposite in direction, and collinear, on the first body. This is the principle of action and reaction. The vector representation of forces implies that they are concentrated either at a single point or along a single line.

Force is measured in newtons (N); a force of 1 N will accelerate a mass of one kilogram at a rate of one meter per second. The newton is a unit of the International System (SI) used for measuring force. Using the English system, the force is measured in pounds (lb).

The force vector **F** can be expressed in terms of a cartesian reference frame, with the unit vectors $\mathbf{\imath}$, $\mathbf{\jmath}$, and $\mathbf{k}$, Fig. 2.6a

$$\mathbf{F} = F_x\mathbf{\imath} + F_y\mathbf{\jmath} + F_z\mathbf{k}. \tag{2.5}$$

The components of the force in the x, y, and z directions are F_x, F_y, and F_z. The resultant of two forces: $\mathbf{F}_1 = F_{1x}\mathbf{\imath} + F_{1y}\mathbf{\jmath} + F_{1z}\mathbf{k}$ and $\mathbf{F}_2 = F_{2x}\mathbf{\imath} + F_{2y}\mathbf{\jmath} + F_{2z}\mathbf{k}$ is the vector sum of those forces

$$\mathbf{R} = \mathbf{F}_1 + \mathbf{F}_2 = (F_{1x} + F_{2x})\mathbf{\imath} + (F_{1y} + F_{2y})\mathbf{\jmath} + (F_{1z} + F_{2z})\mathbf{k}. \tag{2.6}$$

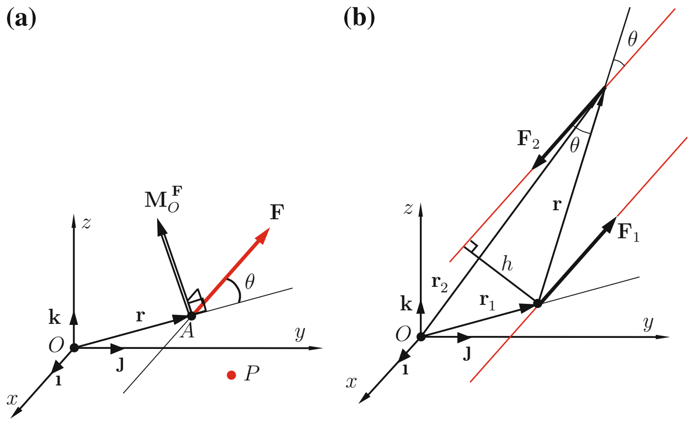


Fig. 2.6 **a** Moment of a force about (with respect to) a point and **b** couple of two forces

A moment is defined as the moment of a force about (with respect to) a point. The moment of the force $\mathbf{F}$ about the point O is the cross product vector

$$\begin{aligned}\mathbf{M}_O^{\mathbf{F}} &= \mathbf{r} \times \mathbf{F} \\ &= \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ r_x & r_y & r_z \\ F_x & F_y & F_z \end{vmatrix} \\ &= (r_y F_z - r_z F_y)\mathbf{\imath} + (r_z F_x - r_x F_z)\mathbf{\jmath} + (r_x F_y - r_y F_x)\mathbf{k}, \end{aligned} \tag{2.7}$$

where $\mathbf{r} = r_x\mathbf{\imath} + r_y\mathbf{\jmath} + r_z\mathbf{k}$ is a position vector directed from the point about which the moment is taken (O in this case) to any point A on the line of action of the force, see Fig. 2.6a. If the coordinates of O are $x_O,\ y_O,\ z_O$ and the coordinates of A are $x_A,\ y_A,\ z_A$, then $\mathbf{r} = \mathbf{r}_{OA} = (x_A - x_O)\mathbf{\imath} + (y_A - y_O)\mathbf{\jmath} + (z_A - z_O)\mathbf{k}$ and the the moment of the force $\mathbf{F}$ about the point O is

$$\mathbf{M}_O^{\mathbf{F}} = \mathbf{r}_{OA} \times \mathbf{F} = \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ x_A - x_O & y_A - y_O & z_A - z_O \\ F_x & F_y & F_z \end{vmatrix}.$$

The magnitude of $\mathbf{M}_O^{\mathbf{F}}$ is

$$|\mathbf{M}_O^{\mathbf{F}}| = M_O^{\mathbf{F}} = r\,F\,|\sin\theta|,$$

where $\theta = \angle(\mathbf{r},\ \mathbf{F})$ is the angle between vectors $\mathbf{r}$ and $\mathbf{F}$, and $r = |\mathbf{r}|$ and $F = |\mathbf{F}|$ are the magnitudes of the vectors. The line of action of $\mathbf{M}_O^{\mathbf{F}}$ is perpendicular to the plane containing $\mathbf{r}$ and $\mathbf{F}$ ($\mathbf{M}_O^{\mathbf{F}} \perp \mathbf{r}\ \&\ \mathbf{M}_O^{\mathbf{F}} \perp \mathbf{F}$) and the sense is given by the right-hand rule. The moment of the force $\mathbf{F}$ about another point P is

$$\mathbf{M}_P^{\mathbf{F}} = \mathbf{r}_{PA} \times \mathbf{F} = \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ x_A - x_P & y_A - y_P & z_A - z_P \\ F_x & F_y & F_z \end{vmatrix},$$

where $x_P,\ y_P,\ z_P$ are the coordinates of the point P.

The system of two forces, $\mathbf{F}_1$ and $\mathbf{F}_2$, which have equal magnitudes $|\mathbf{F}_1| = |\mathbf{F}_2|$, opposite senses $\mathbf{F}_1 = -\mathbf{F}_2$, and parallel directions ($\mathbf{F}_1||\mathbf{F}_2$) is a couple. The resultant force of a couple is zero $\mathbf{R} = \mathbf{F}_1 + \mathbf{F}_2 = \mathbf{0}$. The resultant moment $\mathbf{M} \neq \mathbf{0}$ about an arbitrary point is

$$\mathbf{M} = \mathbf{r}_1 \times \mathbf{F}_1 + \mathbf{r}_2 \times \mathbf{F}_2,$$

or

$$\mathbf{M} = \mathbf{r}_1 \times (-\mathbf{F}_2) + \mathbf{r}_2 \times \mathbf{F}_2 = (\mathbf{r}_2 - \mathbf{r}_1) \times \mathbf{F}_2 = \mathbf{r} \times \mathbf{F}_2, \tag{2.8}$$

where $\mathbf{r} = \mathbf{r}_2 - \mathbf{r}_1$ is a vector from any point on the line of action of $\mathbf{F}_1$ to any point of the line of action of $\mathbf{F}_2$. The direction of the torque of the couple is perpendicular to the plane of the couple and the magnitude is given by, Fig. 2.6b

$$|\mathbf{M}| = M = r\,F_2\,|\sin\theta| = h\,F_2, \tag{2.9}$$

where $h = r\,|\sin\theta|$ is the perpendicular distance between the lines of action. The resultant moment of a couple is independent of the point with respect to which moments are taken.

2.4 Equipollent Force Systems

Two systems $\{S\}$ and $\{S'\}$ of vectors are *equipollent* if and only if

1. the resultant of $\{S\}$, $\mathbf{R}$, is equal to the resultant of $\{S'\}$, $\mathbf{R}'$

$$\mathbf{R} = \mathbf{R}'.$$

2. there exists at least one point about which $\{S\}$ and $\{S'\}$ have equal moments

$$\text{exists } P: \quad \mathbf{M}_P^{\{S\}} = \mathbf{M}_P^{\{S'\}}.$$

Figures 2.7a and b show two forces acting on a rod. The two systems of forces are equipollent. The effects on the rod by the two systems are different tension and compression. Here the equipollence is not a physical equivalence.

Transitivity relation If $\{S\}$ is equipollent to $\{S'\}$, and $\{S'\}$ is equipollent to $\{S''\}$, then $\{S\}$ is equipollent to $\{S''\}$.

Every system $\{S\}$ of bound vectors with the resultant $\mathbf{R}$ is equipollent with a system consisting of a couple C and a single vector $\mathbf{v}$ whose line of action passes through a point O. The torque $\mathbf{M}$ of C depends on the choice of the point $\mathbf{M} = \mathbf{M}_O^{\{S\}}$. The vector $\mathbf{v}$ is independent of the choice of base point, $\mathbf{v} = \mathbf{R}$. A couple C can be equipollent with any system of couples, the sum of whose torque is equal to the torque of C. When a system of vectors consists of a couple of torque $\mathbf{M}$ and a single resultant vector parallel to $\mathbf{M}$, it is called a *wrench*. Any system is equipollent to either a null force and null couple, or a single force, or a single couple, or a wrench.

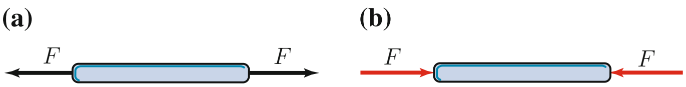


Fig. 2.7 Rod subjected to the action of a pair of forces

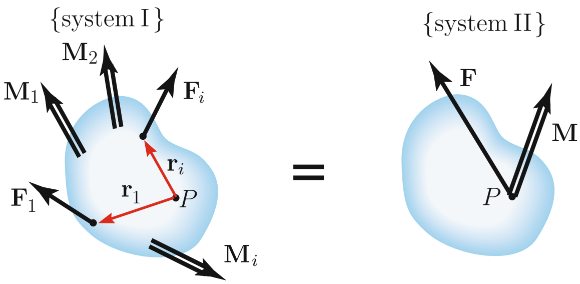


Fig. 2.8 Equipollent systems

To simplify the analysis of forces and moments acting on a given system, the system can be equipollent by a less complicated system. The forces and moments acting on the system can be equipollent with a total force and a total moment system.

Figure 2.8 shows an arbitrary system of forces and moments, {system I}, and a point P. This system is equipollent with a system, {system II}, consisting of a single force $\mathbf{F}$ acting at P and a single couple of torque $\mathbf{M}$. The conditions for equipollence are

$$\sum \mathbf{F}^{\{\text{system II}\}} = \sum \mathbf{F}^{\{\text{system I}\}} \implies \mathbf{F} = \sum \mathbf{F}^{\{\text{system I}\}},$$

and

$$\sum \mathbf{M}_P^{\{\text{system II}\}} = \sum \mathbf{M}_P^{\{\text{system I}\}} \implies \mathbf{M} = \sum \mathbf{M}_P^{\{\text{system I}\}}.$$

These conditions are satisfied if $\mathbf{F}$ equals the sum of the forces in {system I}, and $\mathbf{M}$ equals the sum of the moments about P in {system I}. Thus, no matter how complicated a system of forces and moments may be, it can be represented by a single force acting at a given point and a single couple. Three particular cases occur frequently in practice.

Force Equipollent with a Force and a Couple

A force $\mathbf{F}_I$ acting at a point I {system I} in Fig. 2.9 is equipollent with a force $\mathbf{F}$ acting at a different point P and a couple of torque $\mathbf{M}$, {system II}.The moment of {system I} about point P is $\mathbf{r}_{PI} \times \mathbf{F}_I$, where $\mathbf{r}_{PI}$ is the vector from P to I. The conditions for equipollence are

$$\sum \mathbf{F}^{\{\text{system II}\}} = \sum \mathbf{F}^{\{\text{system I}\}} \implies \mathbf{F} = \mathbf{F}_I,$$

and

$$\sum \mathbf{M}_P^{\{\text{system II}\}} = \sum \mathbf{M}_P^{\{\text{system I}\}} \implies \mathbf{M} = \mathbf{M}_P^{\mathbf{F}_I} = \mathbf{r}_{PI} \times \mathbf{F}_I.$$

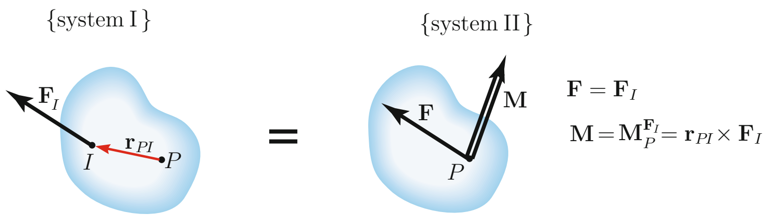


Fig. 2.9 Force $\mathbf{F}_I$ acting on {system I} and equipollent system {system II}

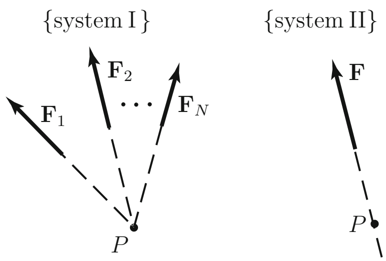


Fig. 2.10 System of concurrent forces and equipollent system

The systems are equipollent if the force $\mathbf{F}$ equals the force $\mathbf{F}_I$ and the couple of torque $\mathbf{M}_P^{\mathbf{F}_I}$ equals the moment of $\mathbf{F}_I$ about P.

Concurrent Forces Equipollent with a Single Force
A system of concurrent forces whose lines of action intersect at a point P {system I} in Fig. 2.10 is equipollent with a single force whose line of action intersects P, {system II}.
The sums of the forces in the two systems are equal if

$$\mathbf{F} = \mathbf{F}_1 + \mathbf{F}_2 + \cdots + \mathbf{F}_n.$$

The sum of the moments about P equals zero for each system, so the systems are equipollent if the force $\mathbf{F}$ equals the sum of the forces in {system I}.

Parallel Forces Equipollent with a Force
A system of parallel forces whose sum is not zero is equipollent with a single force $\mathbf{F}$ shown in Fig. 2.11.

System Equipollent with a Wrench
In general any system of forces and moments is equipollent with a single force acting at a given point and a single couple. Figure 2.12 shows an arbitrary force $\mathbf{F}$ acting at a point I and an arbitrary couple of torque $\mathbf{M}$, {system I}. This system is equipollent with a simpler one where the force $\mathbf{F}$ is acting at a different point P and the component of $\mathbf{M}$ is parallel to $\mathbf{F}$. A coordinate system is chosen so that $\mathbf{F}$ is along the y axis

$$\mathbf{F} = F\mathbf{J},$$

and $\mathbf{M}$ is contained in the xy plane

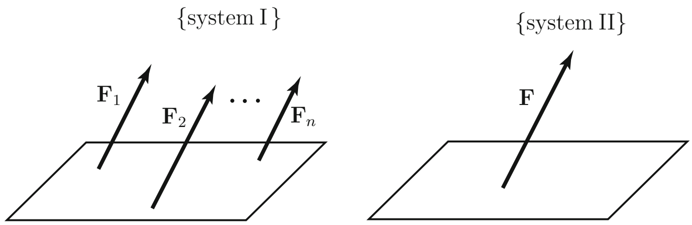


Fig. 2.11 System of parallel forces and equipollent system

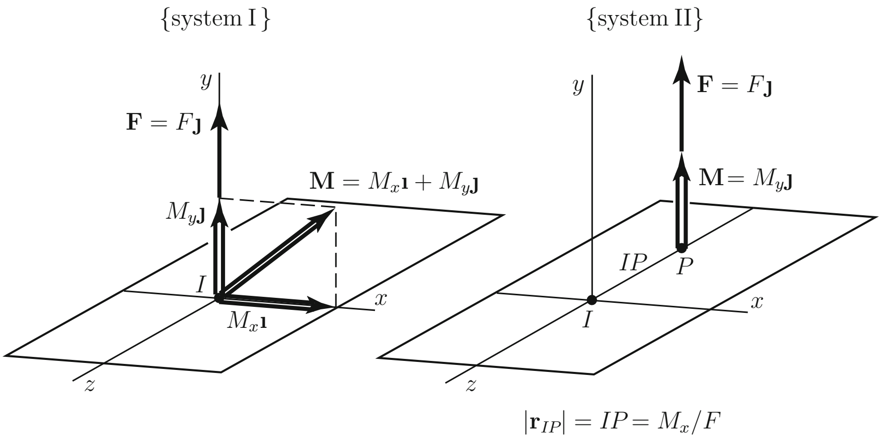


Fig. 2.12 System equipollent with a wrench

$$\mathbf{M} = M_x\mathbf{\imath} + M_y\mathbf{\jmath}.$$

The equivalent system, {system II}, consists of the force **F** acting at a point P on the z axis

$$\mathbf{F} = F\mathbf{\jmath},$$

and the component of **M** parallel to **F**

$$\mathbf{M}_p = M_y\mathbf{\jmath}.$$

The distance IP is chosen so that $|\mathbf{r}_{IP}| = IP = M_x/F$. The {system I} is equipollent to {system II}. The sum of the forces in each system is the same **F**. The sum of the moments about I in {system I} is **M**, and the sum of the moments about I in {system II} is

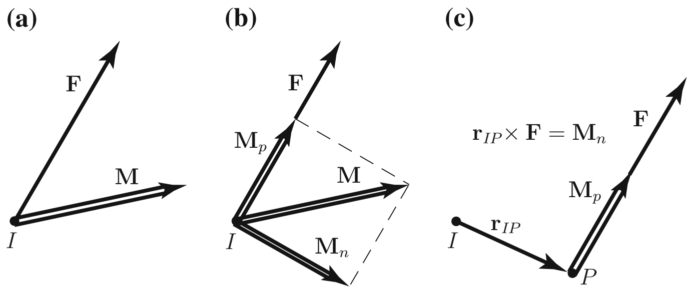


Fig. 2.13 Steps required for a system of forces and moments to be equipollent with wrench

$$\sum \mathbf{M}_I^{\{\text{system II}\}} = \mathbf{r}_{PI} \times \mathbf{F} + M_y\mathbf{J} = [-(IP)\,\mathbf{k}] \times (F\mathbf{J}) + M_y\mathbf{J} = M_x\mathbf{\imath} + M_y\mathbf{J} = \mathbf{M}.$$

The system of the force $\mathbf{F} = F\mathbf{J}$ and the couple $\mathbf{M}_p = M_y\mathbf{J}$ that is parallel to $\mathbf{F}$ is a wrench. A wrench is the simplest system equipollent to an arbitrary system of forces and moments.

A given system of forces and moments is made equipollent with wrench following the steps:

1. Choose a convenient point I the application point of force $\mathbf{F}$ and the moment $\mathbf{M}$, see Fig. 2.13a.
2. Determine the components of $\mathbf{M}$ parallel and normal to $\mathbf{F}$, see Fig. 2.13b:

$$\mathbf{M} = \mathbf{M}_p + \mathbf{M}_n, \quad \text{where} \ \ \mathbf{M}_p || \mathbf{F}.$$

3. The wrench consists of the force $\mathbf{F}$ acting at a point P and the parallel component $\mathbf{M}_p$, see Fig. 2.13c. For equipollence, the following condition must be satisfied:

$$\mathbf{r}_{IP} \times \mathbf{F} = \mathbf{M}_n,$$

where $\mathbf{M}_n$ is the normal component of $\mathbf{M}$.

In general, the {system I} cannot be represented by a force $\mathbf{F}$ alone.

2.5 Examples

Example 2.1 Calculate the moment about the base point O of the the force F, as shown in Fig. 2.14a. For the numerical application use: $F = 500$ N, $\theta = 45°$, $a = 1$ m, and $b = 5$ m.

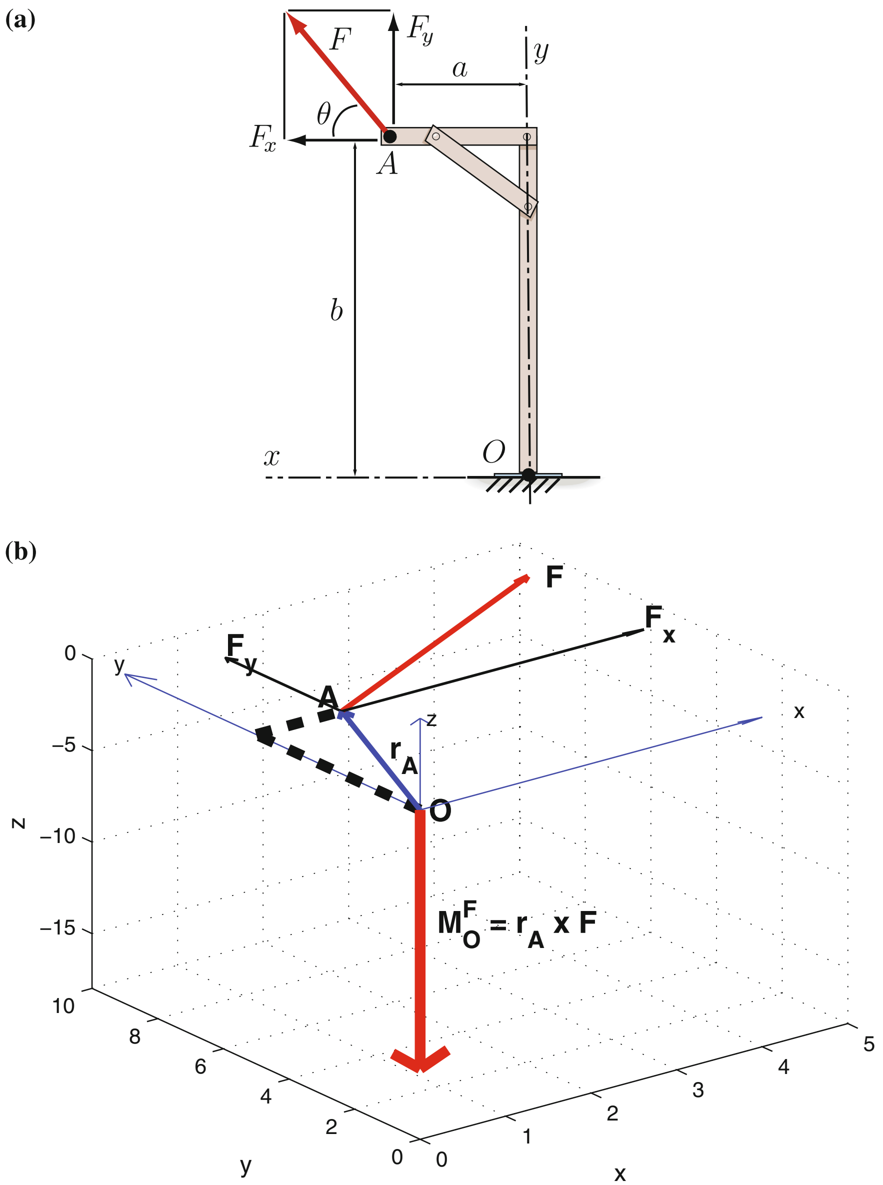


Fig. 2.14 **a** Example 2.1 and **b** MATLAB figure

Solution A cartesian reference frame with the origin at O, as shown in Fig. 2.14a, is selected. The moment of the force $\mathbf{F}$ with respect to the point O is

$$\mathbf{M}_O^{\mathbf{F}} = \mathbf{r}_{OA} \times \mathbf{F} = \begin{vmatrix} \imath & \jmath & \mathbf{k} \\ x_A - x_O & y_A - y_O & 0 \\ F_x & F_y & 0 \end{vmatrix} = \begin{vmatrix} \imath & \jmath & \mathbf{k} \\ a & b & 0 \\ F\cos\theta & F\sin\theta & 0 \end{vmatrix}$$
$$= (a F \cos\theta - b F \sin\theta)\,\mathbf{k} = [(1)\,5\cos 45^\circ - (5)\,500\sin 45^\circ]\,\mathbf{k}$$
$$= -1414.2\,\mathbf{k}\ \text{Nm}.$$

The minus sign indicates that the moment vector is in the negative z-direction. The MATLAB program for the the moment of the force **F** about the point O is:

```
syms F theta a b  real
rA_ = [a b 0];
FA_ = [F*cos(theta) F*sin(theta) 0];
MO_ = cross(rA_, FA_);
MOz= MO_(3);
sl = {F, theta, a, b};
nl = {5, pi/4, 1, 5};
fprintf('MOz = %s =',char(MOz))
fprintf('%6.3f (kN m)\n',subs(MOz,sl,nl))
```

and the output of the program is

```
MOz = a*F*sin(theta)-b*F*cos(theta) = -14.142 (kN m)
```

The MATLAB program for plotting the vectors and the figure are:

```
% numerical values for vectors
rAn_ = double(subs(rA_,sl,nl));
Fn_ = double(subs(FA_,sl,nl));
Mn_ = subs(MO_,sl,nl);
% figure plotting
line([0 0],[0 rAn_(2)],[0,0],'LineStyle','--',...
    'Color','k','LineWidth',4)
line...
([0 rAn_(1)],[rAn_(2) rAn_(2)],[0,0],...
'LineStyle','--','Color','k','LineWidth',4)
% vector plotting
% rAn_
quiver3(0,0,0,rAn_(1),rAn_(2),0,1,...
    'Color','b','LineWidth',2)
% rFn_
quiver3(rAn_(1),rAn_(2),0,Fn_(1),Fn_(2),0,1,...
    'Color','r','LineWidth',2)
% rFn_(1)
quiver3(rAn_(1),rAn_(2),0,Fn_(1),0,0,1,...
    'Color','k','LineWidth',1)
% rFn_(2)
```

```
quiver3(rAn_(1),rAn_(2),0,0,Fn_(2),0,1,...
    'Color','k','LineWidth',1)
% Mn_
quiver3(0,0,0,0,0,Mn_(3),1,...
    'Color','r','LineWidth',4)
```

The vector representation with MATLAB is shown in Fig. 2.14b.

Example 2.2 The beam in Fig. 2.15a is subjected to a T tension that is directed from A to B. Find the the moment created by the force about the support at O. For the numerical application use: $T = 10$ kN, $a = 12$ m, $b = 9$ m, and $c = 15$ m.

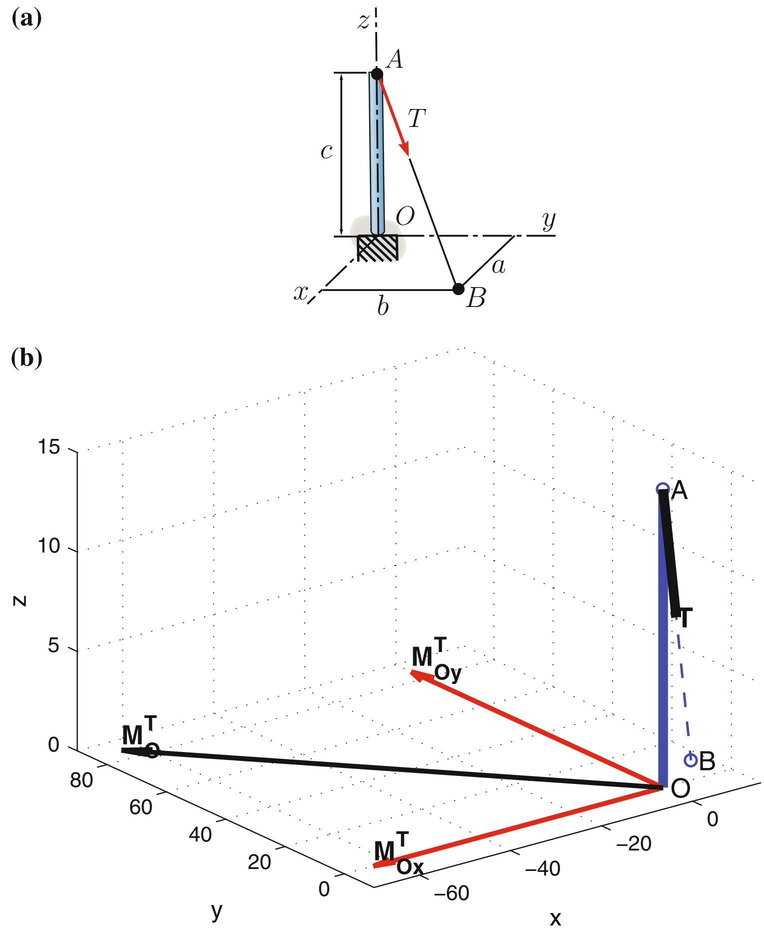


Fig. 2.15 **a** Example 2.2 and **b** MATLAB figure

Solution The vector expression for the tension **T** is

$$\mathbf{T} = T\,\mathbf{u}_{AB} = T\,\frac{\mathbf{r}_{AB}}{|\mathbf{r}_{AB}|} = T\,\frac{(x_B - x_A)\,\mathbf{\imath} + (y_B - y_A)\,\mathbf{\jmath} + (z_B - z_A)\,\mathbf{k}}{\sqrt{(x_B - x_A)^2 + (y_B - y_A)^2 + (z_B - z_A)^2}}$$

$$= T\,\frac{a\,\mathbf{\imath} + b\,\mathbf{\jmath} - c\,\mathbf{k}}{\sqrt{a^2 + b^2 + c^2}} = (10)\,\frac{12\,\mathbf{\imath} + 9\,\mathbf{\jmath} - 15\,\mathbf{k}}{\sqrt{12^2 + 9^2 + 15^2}} = 5.657\,\mathbf{\imath} + 4.243\,\mathbf{\jmath} - 7.071\,\mathbf{k}\ \text{kN},$$

where $r_B = x_B\,\mathbf{\imath} + y_B\,\mathbf{\jmath} + z_B\,\mathbf{k} = a\,\mathbf{\imath} + b\,\mathbf{\jmath}$ and $r_C = x_C\,\mathbf{\imath} + y_C\,\mathbf{\jmath} + z_C\,\mathbf{k} = c\,\mathbf{k}$. The moment of the tension **T** with respect to the point O is

$$\mathbf{M}_O^{\mathbf{T}} = \mathbf{r}_{OA} \times \mathbf{T} = \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ x_A & y_A & z_A \\ T_x & T_y & T_z \end{vmatrix} = \frac{T}{\sqrt{a^2 + b^2 + c^2}} \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ 0 & 0 & c \\ a & b & -c \end{vmatrix} = \frac{T\,(-b\,c\,\mathbf{\imath} + a\,c\,\mathbf{\jmath})}{\sqrt{a^2 + b^2 + c^2}}$$

$$= \frac{10\,[-9\,(15)\,\mathbf{\imath} + 12\,(9)\,\mathbf{\jmath})}{\sqrt{12^2 + 9^2 + 15^2}} = -63.640\,\mathbf{\imath} + 84.853\,\mathbf{\jmath}\ \text{kN m},$$

and $|\mathbf{M}_O^{\mathbf{T}}| = 106.066\,\text{kN m}$.
The MATLAB program for the calculation of **T** and $\mathbf{M}_O^{\mathbf{T}}$ and is given by:

```
syms T a b c real
rB_ = [a b 0];
rA_ = [0 0 c];
rAB_= rB_-rA_;
uAB_= rAB_/sqrt(dot(rAB_, rAB_));
TAB_= T*uAB_;
MO_ = cross(rA_, TAB_);
% numerical calculations
sl = {T, a, b, c};
nl = {10, 12, 9, 15};
Tn_ = subs(TAB_,sl,nl);
Mn_ = subs(MO_,sl,nl);
```

and the output is:

```
T = [ 5.657  4.243 -7.071] (kN)
MOx = -c*T*b/(a^2+b^2+c^2)^(1/2) = -63.640 (kN m)
MOy = c*T*a/(a^2+b^2+c^2)^(1/2) = 84.853 (kN m)
MOz = 0 =  0.000 (kN m)
|MO| = 106.066 (kN m)
```

The MATLAB program for plotting the vectors is:

```
rAn_ = double(subs(rA_,sl,nl));
Fn_ = double(subs(FA_,sl,nl));
Mn_ = subs(MO_,sl,nl);
```

```
axis([0 5  0 10  -18 0])
line([0 0],[0 rAn_(2)],[0,0],'LineStyle','--',...
    'Color','k','LineWidth',4)
line...
([0 rAn_(1)],[rAn_(2) rAn_(2)],[0,0],...
'LineStyle','--','Color','k','LineWidth',4)
quiver3(0,0,0,rAn_(1),rAn_(2),0,1,...
    'Color','b','LineWidth',2)
quiver3(rAn_(1),rAn_(2),0,Fn_(1),Fn_(2),0,1,...
    'Color','r','LineWidth',2)
quiver3(rAn_(1),rAn_(2),0,Fn_(1),0,0,1,...
    'Color','k','LineWidth',1)
quiver3(rAn_(1),rAn_(2),0,0,Fn_(2),0,1,...
    'Color','k','LineWidth',1)
quiver3(0,0,0,0,0,Mn_(3),1,...
    'Color','r','LineWidth',4)
```

The vector representation with MATLAB is shown in Fig. 2.15b.

Example 2.3 Determine the moment of the force F about A as shown in Fig. 2.16a. For the numerical application use: $F = 1$ kN, $a = 1$ m, $b = 3$ m, and $c = 2$ m.

Solution The moment of a force about a point is given by the cross product of a position vector with the force vector. The position vector must run from the point about which the moment is being calculated to a point on the line of action of the force. Figure 2.16a shows the location of the point A, the force F, and the line of action of the force. Point B is on the line of action of the force. Thus the position

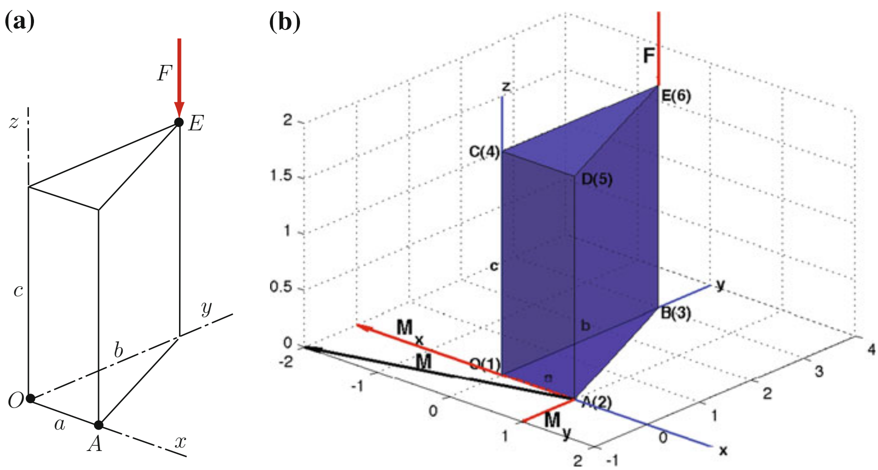


Fig. 2.16 **a** Example 2.3 and **b** MATLAB figure

vector of interest is the vector from point A to point B. From the figure this position vector can be seen to be a units in the -x followed by b units in the positive y.

$$\mathbf{r}_{AB} = -a\,\mathbf{\imath} + b\,\mathbf{\jmath}$$

The force vector is parallel to the z-axis with magnitude F. Thus it can be expressed in vector form as: $\mathbf{F} = -F\,\mathbf{k}$. The desired moment is the cross product of these two vectors

$$\mathbf{M}_A^{\mathbf{F}} = (-a\,\mathbf{\imath} + b\,\mathbf{\jmath}) \times (-F\mathbf{k})\,.$$

Recalling that $\mathbf{\imath} \times \mathbf{k}$ is $-\mathbf{\jmath}$ and $\mathbf{\jmath} \times \mathbf{k}$ is $\mathbf{\imath}$ yields

$$\mathbf{M}_A^{\mathbf{F}} = -b\,F\,\mathbf{\imath} - aF\,\mathbf{\jmath}.$$

The MATLAB program for the moment of the force $\mathbf{F}$ about point A is given by:

```
syms a b c F
rA_ = [a 0 0];
rB_ = [0 b 0];
rE_ = [0 b c];
rAE_ = rE_ - rA_;
rAB_ = rB_ - rA_;
f_ = [0 0 -F];

ME_ = cross(rAE_, f_); % M = rAE x F
MB_ = cross(rAB_, f_); % M = rAB x F
T = ME_ == MB_; % rAB x F = rAE x F
fprintf('ME_ == MB_ => [%d %d %d]\n',T)
fprintf('1=TRUE 0=FALSE\n')

% numerical calculation
sl = {a, b, c, F};
nl = {1, 3, 2, 1};
MEn_ = double(subs(ME_,sl,nl));
MBn_ = double(subs(MB_,sl,nl));
```

The output of the MATLAB program is:

```
ME_ == MB_ => [1 1 1]
1=TRUE 0=FALSE

M_ = rAB_ x F_ = rAE_ x F_
Mx = -F*b; My = -F*a; Mz = 0.
ME_ = [-3.000 -1.000 0] (kN m)
MB_ = [-3.000 -1.000 0] (kN m)
```

The MATLAB program for plotting the vectors and the triangular prism is:

```
F=1; % kN
a=1; b=3; c=2; % m

axis([-2 2 -1 4 0 2])
hold on, grid on

% Cartesian axes
line ...
([0 4],[0 0],[0,0],'Color','b','LineWidth',1.5)
text(3,0,0,'x','fontweight','b')

line ...
([0 0],[0 4],[0,0],'Color','b','LineWidth',1.5)
text(0,4.1,0,'y','fontweight','b')

line ...
([0 0],[0 0],[0,2.5],'Color','b','LineWidth',1.5)
text(0,0,2.6,'z','fontweight','b')

text(-.45,0,0,'O(1)','fontweight','b')
text(a+.1,0,0,'A(2)','fontweight','b')
text(.1,b-.1,0,'B(3)','fontweight','b')
text(-.45,0,c-.1,'C(4)','fontweight','b')
text(a+.1,0,c,'D(5)','fontweight','b')
text(0,b+.05,c-.1,'E(6)','fontweight','b')

text((a+.1)/3,.3,0,'a','fontweight','b')
text(.05,(b-.1)/2,.17,'b','fontweight','b')
text(-.16,0,(c-.1)/2,'c','fontweight','b')

view(42,34);
% view(AZ,EL) set the angle of the view from
% which an observer sees the current 3-D plot
% AZ is the azimuth or horizontal rotation
% EL is the vertical elevation
% (both in degrees)

% Generate data
vert=...
[0 0 0; a 0 0; 0 b 0; 0 0 c; a 0 c; 0 b c];
% define the matrix of the vertices
```

```
% O: 0,0,0 defined as vertex 1
% A: a,0,0 defined as vertex 2
% B: 0,b,0 defined as vertex 3
% C: 0,0,c defined as vertex 4
% D: a,0,c defined as vertex 5
% E: 0,b,c defined as vertex 6

face_up=[1 2 3; 4 5 6];
% define the lower and upper face of
% the triangular prism
% lower face is defined by vertices
% 1, 2, 3 (O, A, B)
% upper face is defined by vertices
% 4, 5, 6 (C, D, E)

face_l=[1 2 5 4; 2 3 6 5; 1 3 6 4];
% generate the lateral faces
% lateral face 1 is defined by 1, 2, 5, 4
% lateral face 2 is defined by 2, 3, 6, 5
% lateral face 3 is defined by 1, 3, 6, 4
% when defined a face the order of the vertices
% has to be given clockwise or counterclockwise

% draw the lower and upper triangular patches
patch...
('Vertices',vert,'Faces',face_up,'facecolor','b')
% patch(x,y,C) adds the "patch" or
% filled 2-D polygon defined by
% vectors x and y to the current axes.
% C specifies the color of the face(s)
% X represents the matrix vert
% Y represents the matrix face_up

% draw the lateral rectangular patches
patch...
('Vertices',vert,'Faces',face_l,'facecolor','b')

quiver3 ...
(0,b,F+c,0,0,-F,1,'Color','r','LineWidth',1.75)
text ...
(-.3,b,c+.2,' F','fontsize',14,'fontweight','b')

quiver3(a,0,0,MBn_(1),MBn_(2),MBn_(3),1,...
    'Color','k','LineWidth',2)
text((a+MBn_(1))/2,MBn_(2)/2,MBn_(3)/2,...
```

```
    '  M','fontsize',14,'fontweight','b')

quiver3 ...
(a,0,0,MBn_(1),0,0,1,'Color','r','LineWidth',2)
text((a+MBn_(1))/1.3,0,0,...
    ' M_x','fontsize',14,'fontweight','b')

quiver3 ...
(a,0,0,0,MBn_(2),0,1,'Color','r','LineWidth',2)
text(a+.3,MBn_(2),0,...
    'M_y','fontsize',14,'fontweight','b')

light('Position',[1 2 3]);
% light('PropertyName',propertyvalue,...)
% light creates a light object in current axes
% Lights affect only patch and surface objects

% light the peaks surface plot with a light source
% located at infinity and oriented along the
% direction defined by the vector [1 2 3]

material shiny

% material shiny makes the objects shiny

alpha('color');
% alpha get or set alpha properties for
% objects in the current axis
% alpha('color') set the alphadata to be
% the same as the color data
```

The vector representation with MATLAB is shown in Fig. 2.16b.

Example 2.4 A force F acts on a link at the point A as shown in Fig. 2.17a. Find an equivalent system consisting of a force at O and a couple. Numerical application: $F = 100$ lb, $OA = l = 1$ ft, $\theta = 45°$, and $\alpha = 100°$.

Solution The original F force is equivalent to the force at O as shown in Fig. 2.17b

$$\begin{aligned}
&\mathbf{R} = \mathbf{F} = -F\cos(\alpha - \theta)\,\imath + F\sin(\alpha - \theta)\,\jmath \\
&= -100\cos(100° - 45°)\,\imath + 100\sin(100° - 45°)\,\jmath = -57.358\,\imath + 81.915\,\jmath \text{ lb.}
\end{aligned}$$

The moment of the force **F** with respect to the point O, as shown in Fig. 2.17b, is

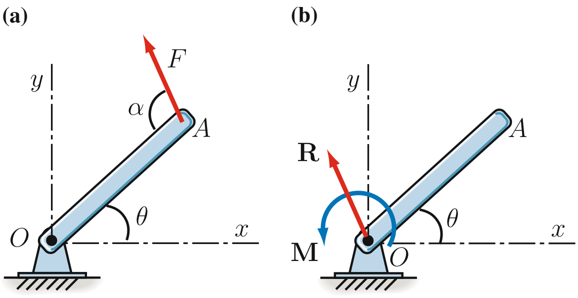


Fig. 2.17 Example 2.4

$$\mathbf{M} = \mathbf{M}_O^{\mathbf{F}} = \mathbf{r}_{OA} \times \mathbf{F} = \begin{vmatrix} \mathbf{ı} & \mathbf{j} & \mathbf{k} \\ x_A & y_A & 0 \\ F_x & F_y & 0 \end{vmatrix} = \begin{vmatrix} \mathbf{ı} & \mathbf{j} & \mathbf{k} \\ l\cos\theta & l\sin\theta & 0 \\ -F\cos(\alpha-\theta) & F\sin(\alpha-\theta) & 0 \end{vmatrix}$$
$$= [l\,F\,(\cos\theta)\sin(\alpha-\theta) + l\,F\,(\sin\theta)\cos(\alpha-\theta)]\,\mathbf{k}$$
$$= [1(100)\,(\cos 45^\circ)\sin(100^\circ - 45^\circ) + 1(100)\,(\sin 45^\circ)\cos(100^\circ - 45^\circ)]\,\mathbf{k}$$
$$= 98.481\,\mathbf{k}\ \text{ lb ft.}$$

The MATLAB program is:

```
syms F l theta alfa real
sl = {F, l, theta, alfa};
nl = {100, 1, pi/4, pi/1.8};
FA_ = [-F*cos(alfa-theta), F*sin(alfa-theta), 0];
rA_ = [l*cos(theta), l*sin(theta), 0];
FAn_= subs(FA_, sl, nl);
MO_ = cross(rA_, FA_);
MOz= simplify(MO_(3));
MOzn= subs(MOz, sl, nl);
```

and the results are:

```
R_ = [-57.358 81.915 0](lb)
MOz = F*l*sin(alfa) = 98.481 (lb.ft)
```

Example 2.5 Three forces $\mathbf{F}_A$, $\mathbf{F}_B$, and $\mathbf{F}_C$, as shown in Fig. 2.18, are acting on a rectangular planar plate ($\mathbf{F}_A || Oz$, $\mathbf{F}_B || Oy$, $\mathbf{F}_C || Ox$). The three forces acting on the plate are replaced by a wrench. Find: (a) the resultant force for the wrench; (b) the magnitude of couple moment, M, for the wrench and the point $T(x, z)$ where its line of action intersects the plate. For the numerical application use: $F_A = 900$ lb, $F_B = 500$ lb, $F_C = 300$ lb, $a = BC = 4$ ft, and $b = OC = 6$ ft.

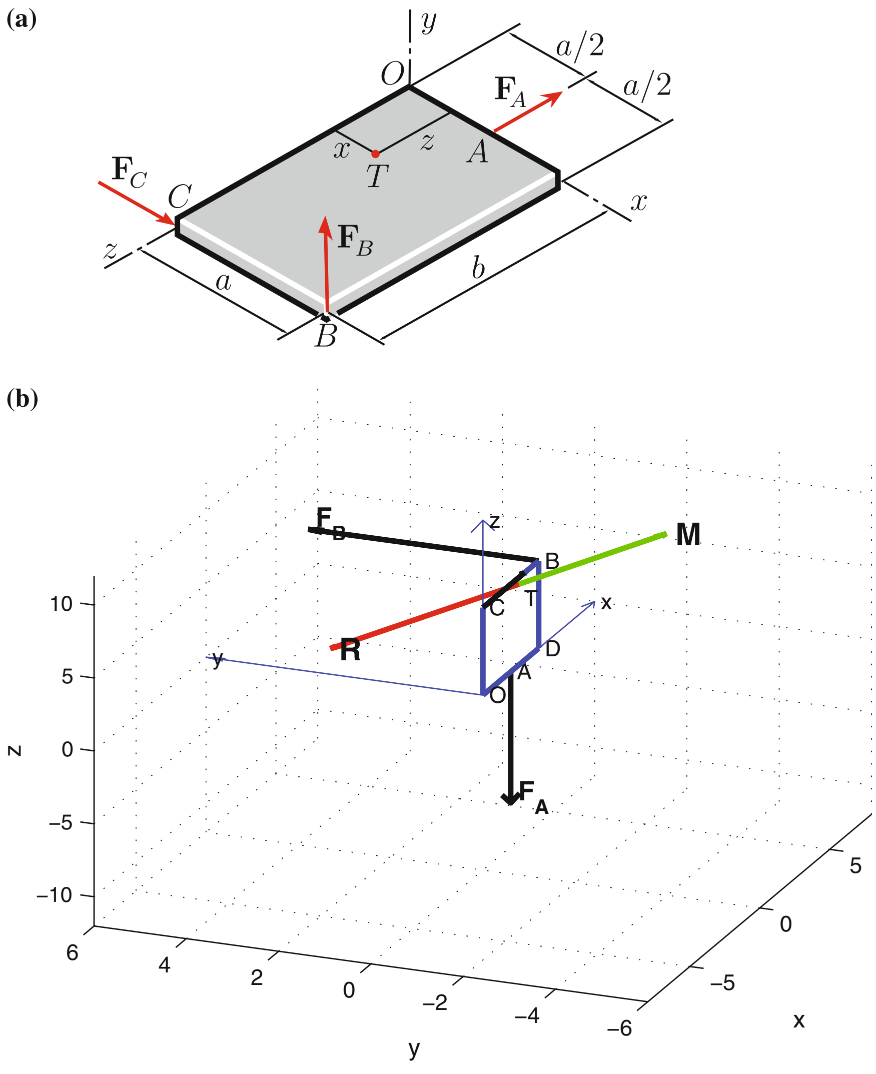


Fig. 2.18 **a** Example 2.5 and **b** MATLAB figure

Solution (a) The direction cosines of the resultant force **R**, are the same as those of the moment **M** of the couple of the wrench, assuming that the wrench is positive. The resultant force is

$$\mathbf{R} = \mathbf{F}_A + \mathbf{F}_B + \mathbf{F}_C = F_C\,\mathbf{\imath} + F_B\,\mathbf{\jmath} - F_A\,\mathbf{k} = 300\,\mathbf{\imath} + 500\,\mathbf{\jmath} - 900\,\mathbf{k} \ \text{ lb}$$

$$R = |\mathbf{R}| = \sqrt{F_A^2 + F_B^2 + F_C^2} = \sqrt{300^2 + 500^2 + 900^2} = 1072.381 \ \text{ lb} = 1.072 \ \text{ kip}.$$

The direction cosines of the resultant force are

$$\cos\theta_x = \frac{F_C}{R} = 0.280, \quad \cos\theta_y = \frac{F_B}{R} = 0.466, \quad \cos\theta_z = \frac{-F_A}{R} = -0.839.$$

The MATLAB program for calculating the direction cosines or the components of the unit vector of the resultant force are:

```
syms a b FA FB FC x z M
sl = {a, b, FA, FB, FC};
nl = {4, 6, 0.9, 0.5, 0.3};
FA_ = [0 0 -FA]; rA_ = [a/2 0 0];
FB_ = [0 FB  0]; rB_ = [a 0 b];
FC_ = [FC 0  0]; rC_ = [0 0 b];
R_ = FA_+FB_+FC_;
Rn_ = subs(R_, sl, nl);
uR_ = R_/magn(R_);
uRn_ = subs(uR_, sl, nl);
```

The function `magn` is:

```
function val = magn(v)
% The symbolic magnitude function of a vector
% v = [v(1) v(2) v(3)]
% The function accepts sym as the input argument
val=sqrt(v(1)*v(1)+v(2)*v(2)+v(3)*v(3));
```

(b) The moment of the wrench couple must equal the sum of the moments of the given forces about point T through which the resultant passes. The moments about $T(x,\ 0,\ z)$ of the three forces are

$$\mathbf{M}_T = \mathbf{M}_T^{\mathbf{F}_A} + \mathbf{M}_T^{\mathbf{F}_B} + \mathbf{M}_T^{\mathbf{F}_C},$$

where

$$\mathbf{M}_T^{\mathbf{F}_A} = \mathbf{r}_{TA} \times \mathbf{F}_A = \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ x_A - x & y_A & z_A - z \\ 0 & 0 & -F_A \end{vmatrix} = \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ a - x & 0 & -z \\ 0 & 0 & -F_A \end{vmatrix} = (a-x)\,F_A\,\mathbf{\jmath}.$$

$$\mathbf{M}_T^{\mathbf{F}_B} = \mathbf{r}_{TB} \times \mathbf{F}_B = \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ x_B - x & y_B & z_B - z \\ 0 & F_B & 0 \end{vmatrix} = \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ a - x & 0 & b - z \\ 0 & F_B & 0 \end{vmatrix} = (z-b)\,F_B\,\mathbf{\imath}$$
$$+ (a-x)\,F_B\,\mathbf{k}.$$

$$\mathbf{M}_T^{\mathbf{F}_C} = \mathbf{r}_{TC} \times \mathbf{F}_C = \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ x_C - x & y_C & z_C - z \\ F_C & 0 & 0 \end{vmatrix} = \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ -x & 0 & b - z \\ F_C & 0 & 0 \end{vmatrix} = (b-z)\,F_C\,\mathbf{\jmath}.$$

The total moment about the point T of the forces is

$$\mathbf{M} = (z - b)\,F_B\,\mathbf{\imath} + [(a - x)\,F_A + (b - z)\,F_C]\,\mathbf{\jmath} + (a - x)\,F_B\,\mathbf{k}.$$

The direction cosines of the moment $\mathbf{M}$, of magnitude M, are the same as the direction cosines of the resultant $\mathbf{R}$ and three scalar equations can be written

$$\cos\theta_x = \frac{M_x}{M},\ \cos\theta_y = \frac{M_y}{M},\ \cos\theta_z = \frac{M_z}{M}, \quad \text{or}$$
$$\frac{F_C}{R} = \frac{(z-b)\,F_B}{M},\ \frac{F_B}{R} = \frac{(a-x)\,F_A + (b-z)\,F_C}{M},\ \frac{-F_A}{R} = \frac{(a-x)\,F_B}{M} \quad \text{or}$$

$$-3000 + 500\,z = 0.280\,M,$$
$$3600 - 900\,x - 300\,z = 0.465\,M,$$
$$2000 - 500\,x = -0.839\,M.$$

There are three scalar equations with three unknowns M, x, and z. The solution of the equations is obtained using the MATLAB function `solve`:

```
rT_ = [x 0 z];
MTA_ = cross(rA_-rT_, FA_);
MTB_ = cross(rB_-rT_, FB_);
MTC_ = cross(rC_-rT_, FC_);
MT_ = MTA_ + MTB_ + MTC_;
eq1 = MT_(1)/M - uR_(1);
eq2 = MT_(2)/M - uR_(2);
eq3 = MT_(3)/M - uR_(3);
eq1n = subs(eq1, sl, nl);
eq2n = subs(eq2, sl, nl);
eq3n = subs(eq3, sl, nl);
digits(3)
fprintf('first equation:\n')
pretty(eq1)
fprintf('%s = 0 \n\n',char(vpa(eq1n)))
fprintf('second equation:\n')
pretty(eq2)
fprintf('%s = 0 \n\n',char(vpa(eq2n)))
fprintf('third equation:\n')
pretty(eq3)
fprintf('%s = 0 \n\n',char(vpa(eq3n)))
sol = solve(eq1, eq2, eq3,'x, z, M');
Ms = sol.M; Mn = subs(Ms, sl, nl);
xs = sol.x; xn = subs(xs, sl, nl);
zs = sol.z; zn = subs(zs, sl, nl);
```

```
fprintf('M = ')
pretty(Ms)
fprintf('M = %6.3f (kip ft)\n', Mn)
fprintf('x = ')
pretty(xs)
fprintf('x = %6.3f (ft)\n', xn)
fprintf('z = ')
pretty(zs)
fprintf('z = %6.3f (ft)\n', zn)
```

The function `pretty(x)` prints the symbolic expression `x` in a format that looks like type-set mathematics. The results obtained with MATLAB are:

```
first equation:
             FC                FB (b - z)
  - -------------------- - ----------
        2      2      2 1/2         M
     (FA   + FB   + FC )
(0.5*(z - 6.0))/M - 0.28 = 0

second equation:
                   / a      \
  FC (b - z) + FA | - - x |
                   \ 2      /            FB
  ------------------------- - --------------------
              M                     2      2      2 1/2
                                 (FA   + FB   + FC )
- (1.0*(0.9*x + 0.3*z - 3.6))/M - 0.466 = 0

third equation:
           FA                FB (a - x)
  -------------------- + ----------
      2      2      2 1/2         M
  (FA   + FB   + FC )
0.839 - (0.5*(x - 4.0))/M = 0

M =
                 2      2      2 1/2
    FA FB a (FA   + FB   + FC )
  - ----------------------------
             2        2        2
        2 FA   + 2 FB   + 2 FC
M = -0.839 (kip ft)
x =
       2         2         2
```

```
  a FA  + 2 a FB  + 2 a FC
  -------------------------
        2       2       2
    2 FA  + 2 FB  + 2 FC
x =  2.591 (ft)
z =
        2                   2         2
  2 b FA  - a FA FC + 2 b FB  + 2 b FC
  -------------------------------------
              2       2       2
          2 FA  + 2 FB  + 2 FC
z =  5.530 (ft)
```

The moment $M = -839.254$ lb ft = -0.839 kip ft is negative, and that is why the couple vector is pointing in the direction opposite to **R**, which makes the wrench negative. The MATLAB program for plotting the vectors and the figure are:

```
a=4; b=6;
axis([-2*a 2*a  -b b -2*b 2*b])

xA=a/2; yA=0; zA=0;
xB=a; yB=0; zB=b;
xC=0; yC=0; zC=b;
xD=a; yD=0; zD=0;
xT=xn; yT=0; zT=zn;

line([0 xC],[0 yC],[0,zC],...
    'Color','b','LineWidth',2)
line([0 xD],[0 yD],[0,zD],...
    'Color','b','LineWidth',2)
line([xD xB],[yD yB],[zD,zB],...
    'Color','b','LineWidth',2)
line([xC xB],[yC yB],[zC,zB],...
    'Color','b','LineWidth',2)

fs=10; % force scale
FAn_ = fs*subs(FA_, sl, nl);
FBn_ = fs*subs(FB_, sl, nl);
FCn_ = fs*subs(FC_, sl, nl);
Rtn_ = fs*Rn_;
Mtn_ = fs*Mn*uRn_;

quiver3...
(xA,yA,zA,FAn_(1),FAn_(2),FAn_(3),1,...
'Color','k','LineWidth',2)
```

```
quiver3...
(xB,yB,zB,FBn_(1),FBn_(2),FBn_(3),1,...
'Color','k','LineWidth',2)
quiver3...
(xC,yC,zC,FCn_(1),FCn_(2),FCn_(3),1,...
'Color','k','LineWidth',2)
quiver3...
(xT,yT,zT,Rtn_(1),Rtn_(2),Rtn_(3),1,...
'Color','r','LineWidth',2)
quiver3...
(xT,yT,zT,Mtn_(1),Mtn_(2),Mtn_(3),1,...
'Color','G','LineWidth',2)
```

The vector representation with MATLAB is shown in Fig. 2.18b.

2.6 Problems

2.1 (a) Determine the resultant of the forces $\mathbf{F}_1 = F_{1x}\,\mathbf{ı} + F_{1y}\,\mathbf{ȷ} + F_{1z}\,\mathbf{k}$, $\mathbf{F}_2 = F_{2x}\,\mathbf{ı} + F_{2y}\,\mathbf{ȷ} + F_{2z}\,\mathbf{k}$, and $\mathbf{F}_3 = F_{3x}\,\mathbf{ı} + F_{3y}\,\mathbf{ȷ} + F_{3z}\,\mathbf{k}$, which are concurrent at the point $P(x_P,\ y_P,\ z_P)$, where $F_{1x} = 2$, $F_{1y} = 3.5$, $F_{1z} = -3$, $F_{2x} = -1.5$, $F_{2y} = 4.5$, $F_{2z} = -3$, $F_{3x} = 7$, $F_{3y} = -6$, $F_{3z} = 5$, $x_P = 1$, $y_P = 2$, and $z_P = 3$. (b) Find the total moment of the given forces about the origin $O(0,\ 0,\ 0)$. The units for the forces are in Newtons and for the coordinates are given in meters.

2.2 (a) Determine the resultant of the three forces shown in Fig. 2.19. The force $\mathbf{F}_1$ acts along the x-axis, the force $\mathbf{F}_2$ acts along the z-axis, and the direction of the force $\mathbf{F}_3$ is given by the line O_3P_3, where $O_3 = O(x_{O_3},\ y_{O_3},\ z_{O_3})$ and $P_3 = P(x_{P_3},\ y_{P_3},\ z_{P_3})$. The application point of the forces $\mathbf{F}_1$ and $\mathbf{F}_2$ is the origin $O(0,\ 0,\ 0)$ of the reference frame as shown in Fig. 2.19. (b) Find the total moment of the given forces about the point P_3. Numerical application: $|\mathbf{F}_1| = F_1 = 250$ N, $|\mathbf{F}_2| = F_2 = 300$ N, $|\mathbf{F}_3| = F_3 = 300$ N, $O_3 = O_3(1,\ 2,\ 3)$ and $P_3 = P_3(5,\ 7,\ 9)$. The coordinates are given in meters.

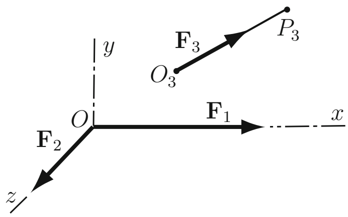


Fig. 2.19 Problem 2.2

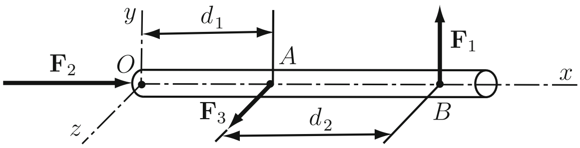


Fig. 2.20 Problem 2.3

2.3 Replace the three forces $\mathbf{F}_1$, $\mathbf{F}_2$, and $\mathbf{F}_3$, shown in Fig. 2.20, by a resultant force, $\mathbf{R}$, through O and a couple. The force $\mathbf{F}_2$ acts along the x-axis, the force $\mathbf{F}_1$ is parallel to the y-axis, and the force $\mathbf{F}_3$ is parallel to the z-axis. The application point of the forces $\mathbf{F}_2$ is O, the application point of the forces $\mathbf{F}_1$ is B, and the application points of the force $\mathbf{F}_3$ is A. The distance between O and A is d_1 and the distance between A and B is d_2 as shown in Fig. 2.20. Numerical application: $|\mathbf{F}_1| = F_1 = 250$ N, $|\mathbf{F}_2| = F_2 = 300$ N, $|\mathbf{F}_3| = F_3 = 400$ N, $d_1 = 1.5$ m and $d_2 = 2$ m.

2.4 Two forces $\mathbf{F}_1$ and $\mathbf{F}_2$ and a couple of moment M in the xy plane are given. The force $\mathbf{F}_1 = F_{1x}\,\mathbf{\imath} + F_{1y}\,\mathbf{\jmath} + F_{1z}\,\mathbf{k}$ acts at the point $P_1 = P_1(x_1,\ y_1,\ z_1)$ and the force $\mathbf{F}_2 = F_{2x}\,\mathbf{\imath} + F_{2y}\,\mathbf{\jmath} + F_{2z}\,\mathbf{k}$ acts at the point $P_2 = P_2(x_2,\ y_2,\ z_2)$. Find the resultant force-couple system at the origin $O(0,\ 0,\ 0)$. Numerical application: $F_{1x} = 10$, $F_{1y} = 5$, $F_{1z} = 40$, $F_{2x} = 30$, $F_{2y} = 10$, $F_{2z} = -30$, $F_{3x} = 7$, $F_{3y} = -6$, $F_{3z} = 5$, $P_1 = P_1(0,\ 1,\ -1)$, $P_2 = P_2(1,\ 1,\ 1)$ and $M = -30$ N·m. The units for the forces are in Newtons and for the coordinates are given in meters.

2.5 Replace the three forces $\mathbf{F}_1$, $\mathbf{F}_2$, and $\mathbf{F}_3$, shown in Fig. 2.21, by a resultant force at the origin O of the reference frame and a couple. The force $\mathbf{F}_1$ acts along the x-axis, the force $\mathbf{F}_2$ is parallel with the z-axis, and the force $\mathbf{F}_3$ is parallel with the y-axis. The application point of the force $\mathbf{F}_1$ is at O, the application point of the forces $\mathbf{F}_2$ is at A, and the application points of the force $\mathbf{F}_3$ is at B. The distance between the origin O and the point A is d_1 and the distance between the point A and the point B is d_2. The line AB is parallel with the z-axis. Numerical application: $|\mathbf{F}_1| = F_1 = 50$ N, $|\mathbf{F}_2| = F_2 = 30$ N, $|\mathbf{F}_3| = F_3 = 60$ N, $d_1 = 1$ m, and $d_2 = 0.7$ m

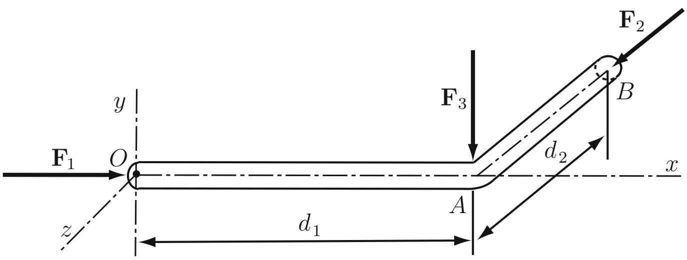


Fig. 2.21 Problem 2.5

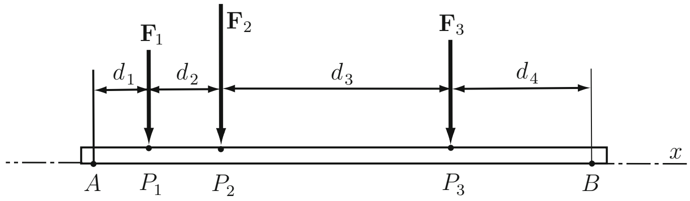


Fig. 2.22 Problem 2.6

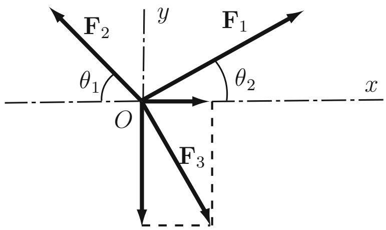
2.6 Three forces $\mathbf{F}_1$, $\mathbf{F}_2$ and $\mathbf{F}_3$ act on a beam as shown in Fig. 2.22. The directions of the forces are parallel with y-axis. The application points of the forces are P_1, P_2, and P_3, and the distances $AP_1 = d_1$, $P_1P_2 = d_2$, $P_2P_3 = d_3$ and $P_3B = d_4$ are given. (a) Find the resultant of the system. (b) Resolve this resultant into two components at the points A and B. Numerical application: $|\mathbf{F}_1| = F_1 = 30$ N, $|\mathbf{F}_2| = F_2 = 60$ N, $|\mathbf{F}_3| = F_3 = 50$ N, $d_1 = 0.1$ m, $d_2 = 0.3$ m, $d_3 = 0.4$ m and $d_4 = 0.4$ m.

2.7 A force $\mathbf{F}$ acts vertically downward, parallel to the y-axis, and intersects the xz plane at the point $P_1(x_1, y_1, z_1)$. Resolve this force into three components acting through the points $P_2 = P_2(x_2, y_2, z_2)$, $P_3 = P_3(x_3, y_3, z_3)$ and $P_4 = P_4(x_4, y_4, z_4)$. Numerical application: $|\mathbf{F}| = F = 50$ N, $P_1 = P_1(2, 0, 4)$, $P_2 = P_2(1, 1, 1)$, $P_3 = P_3(6, 0, 0)$, and $P_4 = P_4(0, 0, 3)$. The coordinates are given in meters.

2.8 Determine the resultant of the given system of forces $\mathbf{F}_1$, $\mathbf{F}_2$, and $\mathbf{F}_3$, shown in the Fig. 2.23. The angle between the direction of the force $\mathbf{F}_1$ and the Ox axis is θ_1 and the angle between the direction of the force $\mathbf{F}_2$ with the x-axis is θ_2. The x and y components of the force $\mathbf{F}_3 = |\mathbf{F}_{3x}|\,\imath + |\mathbf{F}_{3y}|\,\jmath = F_{3x}\,\imath + F_{3y}\,\jmath$ are given. Numerical application: $|\mathbf{F}_1| = F_1 = 250$ N, $|\mathbf{F}_2| = F_2 = 220$ N, $|\mathbf{F}_{3x}| = F_{3x} = 50$ N, $|\mathbf{F}_{3y}| = F_{3y} = 120$ N, $\theta_1 = 30°$, and $\theta_2 = 45°$.

2.9 The rectangular plate in Fig. 2.24 is subjected to four parallel forces. Determine the magnitude and direction of a resultant force equivalent to the given force system and locate its point of application on the plate. Numerical application: $F_O = 700$ lb, $F_A = 600$ lb, $F_B = 500$ lb, $F_C = 100$ lb, $a = 8$ ft, and $b = 10$ ft. Hint: the moments about the x-axis and y-axis of the resultant force, are equal

Fig. 2.23 Problem 2.8



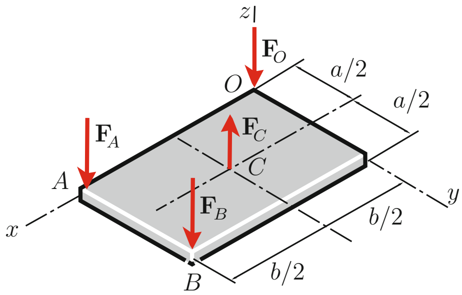


Fig. 2.24 Problem 2.9

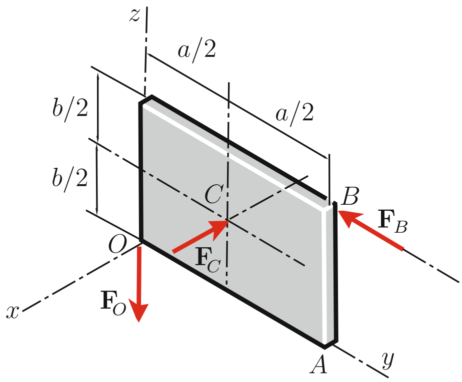


Fig. 2.25 Problem 2.10

to the sum of the moments about the x-axis and y-axis of all the forces in the system.

2.10 Three forces $\mathbf{F}_O$, $\mathbf{F}_B$, and $\mathbf{F}_C$, as shown in Fig. 2.25 , are acting on a rectangular planar plate ($\mathbf{F}_O||Oz$, $\mathbf{F}_B||Oy$, $\mathbf{F}_C||Ox$). The three forces acting on the plate are replaced by a wrench. Find: (a) the resultant force for the wrench; (b) the magnitude of couple moment, M, for the wrench and the point $Q(y, z)$ where its line of action intersects the plate. Numerical application: $F_O = 800$ lb, $F_B = F_C = 500$ lb, $a = OA = 6$ ft, and $b = AB = 5$ ft.

2.7 Programs

2.7.1 Program 2.1

```
% example 2.1
clear all; clc; close all
syms F theta a b real
rA_ = [a b 0];
```

```
FA_ = [F*cos(theta) F*sin(theta) 0];
MO_ = cross(rA_, FA_);
MOz= MO_(3);
sl = {F, theta, a, b};
nl = {5, pi/4, 1, 5};
fprintf('MOz = %s =',char(MOz))
fprintf('%6.3f (kN m)\n',subs(MOz,sl,nl))

% numerical values
rAn_ = double(subs(rA_,sl,nl));
Fn_ = double(subs(FA_,sl,nl));
Mn_ = subs(MO_,sl,nl);

% vector plotting
axis([0 5  0 10  -18 0])
xlabel('x'), ylabel('y'), zlabel('z')
hold on, grid on

% Cartesian axes
text(0,0,0,' O','fontsize',14,'fontweight','b')
quiver3(0,0,0,4,0,0,1,'Color','b')
text(4.1,0,0,'x')
quiver3(0,0,0,0,9,0,1,'Color','b')
text(0,9.4,0,'y')
quiver3(0,0,0,0,0,5,1,'Color','b')
text(0,0,5.5,' z')

line([0 0],[0 rAn_(2)],[0,0],'LineStyle','--',...
    'Color','k','LineWidth',4)
line...
([0 rAn_(1)],[rAn_(2) rAn_(2)],[0,0],...
'LineStyle','--','Color','k','LineWidth',4)
text(rAn_(1),rAn_(2),0,'    A',...
     'fontsize',14,'fontweight','b')

quiver3(0,0,0,rAn_(1),rAn_(2),0,1,...
    'Color','b','LineWidth',2)
text(rAn_(1)/2,rAn_(2)/2,0,...
    ' r_A','fontsize',14,'fontweight','b')

quiver3(rAn_(1),rAn_(2),0,Fn_(1),Fn_(2),0,1,...
    'Color','r','LineWidth',2)
quiver3(rAn_(1),rAn_(2),0,Fn_(1),0,0,1,...
    'Color','k','LineWidth',1)
quiver3(rAn_(1),rAn_(2),0,0,Fn_(2),0,1,...
```

```
    'Color','k','LineWidth',1)
text(rAn_(1)+Fn_(1),rAn_(2),0,...
    'F_x','fontsize',14,'fontweight','b')
text(rAn_(1),rAn_(2)+Fn_(2),0,...
    'F_y','fontsize',14,'fontweight','b')
text(rAn_(1)+Fn_(1),rAn_(2)+Fn_(2),0,...
'  F','fontsize',14,'fontweight','b')

quiver3(0,0,0,0,0,Mn_(3),1,...
    'Color','r','LineWidth',4)
text(Mn_(1)/2,Mn_(2)/2,Mn_(3)/2,...
    '  M_O^F = r_A x F',...
    'fontsize',14,'fontweight','b')

% end of program
```

2.7.2 Program 2.2

```
% example 2.1
clear all; clc; close all
syms F theta a b real
rA_ = [a b 0];
FA_ = [F*cos(theta) F*sin(theta) 0];
MO_ = cross(rA_, FA_);
MOz= MO_(3);
sl = {F, theta, a, b};
nl = {5, pi/4, 1, 5};
fprintf('MOz = %s =',char(MOz))
fprintf('%6.3f (kN m)\n',subs(MOz,sl,nl))

% numerical values
rAn_ = double(subs(rA_,sl,nl));
Fn_ = double(subs(FA_,sl,nl));
Mn_ = subs(MO_,sl,nl);

% vector plotting
axis([0 5  0 10  -18 0])
xlabel('x'), ylabel('y'), zlabel('z')
hold on, grid on

% Cartesian axes
text(0,0,0,' O','fontsize',14,'fontweight','b')
quiver3(0,0,0,4,0,0,1,'Color','b')
```

```
text(4.1,0,0,'x')
quiver3(0,0,0,0,9,0,1,'Color','b')
text(0,9.4,0,'y')
quiver3(0,0,0,0,0,5,1,'Color','b')
text(0,0,5.5,' z')

line([0 0],[0 rAn_(2)],[0,0],'LineStyle','--',...
    'Color','k','LineWidth',4)
line...
([0 rAn_(1)],[rAn_(2) rAn_(2)],[0,0],...
'LineStyle','--','Color','k','LineWidth',4)
text(rAn_(1),rAn_(2),0,'   A',...
     'fontsize',14,'fontweight','b')

quiver3(0,0,0,rAn_(1),rAn_(2),0,1,...
    'Color','b','LineWidth',2)
text(rAn_(1)/2,rAn_(2)/2,0,...
    ' r_A','fontsize',14,'fontweight','b')

quiver3(rAn_(1),rAn_(2),0,Fn_(1),Fn_(2),0,1,...
    'Color','r','LineWidth',2)
quiver3(rAn_(1),rAn_(2),0,Fn_(1),0,0,1,...
    'Color','k','LineWidth',1)
quiver3(rAn_(1),rAn_(2),0,0,Fn_(2),0,1,...
    'Color','k','LineWidth',1)
text(rAn_(1)+Fn_(1),rAn_(2),0,...
    'F_x','fontsize',14,'fontweight','b')
text(rAn_(1),rAn_(2)+Fn_(2),0,...
    'F_y','fontsize',14,'fontweight','b')
text(rAn_(1)+Fn_(1),rAn_(2)+Fn_(2),0,...
'  F','fontsize',14,'fontweight','b')

quiver3(0,0,0,0,0,Mn_(3),1,...
    'Color','r','LineWidth',4)
text(Mn_(1)/2,Mn_(2)/2,Mn_(3)/2,...
    '  M_O^F = r_A x F',...
    'fontsize',14,'fontweight','b')

% end of program
```

2.7.3 Program 2.3

```
% example 2.3
clear all; clc; close all
syms a b c F
rA_ = [a 0 0];
rB_ = [0 b 0];
rE_ = [0 b c];
rAE_ = rE_ - rA_;
rAB_ = rB_ - rA_;
f_ = [0 0 -F];

ME_ = cross(rAE_, f_); % M = rAE x F
MB_ = cross(rAB_, f_); % M = rAB x F
T = ME_ == MB_; % rAB x F = rAE x F
fprintf('ME_ == MB_ => [%d %d %d]\n',T)
fprintf('1=TRUE 0=FALSE\n')
fprintf('\n')
fprintf('M_ = rAB_ x F_ = rAE_ x F_ \n')
fprintf('Mx = %s; ',char(ME_(1)))
fprintf('My = %s; ',char(ME_(2)))
fprintf('Mz = %s.\n',char(ME_(3)))

% numerical calculation
sl = {a, b, c, F};
nl = {1, 3, 2, 1};
MEn_ = double(subs(ME_,sl,nl));
MBn_ = double(subs(MB_,sl,nl));

fprintf('ME_ = [%6.3f %6.3f %d] (kN m)\n',MEn_)
fprintf('MB_ = [%6.3f %6.3f %d] (kN m)\n',MBn_)

% graphical representation
F=1; % kN
a=1; b=3; c=2; % m

axis([-2 2 -1 4 0 2])
hold on, grid on

% Cartesian axes
line ...
([0 4],[0 0],[0,0],'Color','b','LineWidth',1.5)
text(3,0,0,'x','fontweight','b')

line ...
```

```
([0 0],[0 4],[0,0],'Color','b','LineWidth',1.5)
text(0,4.1,0,'y','fontweight','b')

line ...
([0 0],[0 0],[0,2.5],'Color','b','LineWidth',1.5)
text(0,0,2.6,'z','fontweight','b')

text(-.45,0,0,'O(1)','fontweight','b')
text(a+.1,0,0,'A(2)','fontweight','b')
text(.1,b-.1,0,'B(3)','fontweight','b')
text(-.45,0,c-.1,'C(4)','fontweight','b')
text(a+.1,0,c,'D(5)','fontweight','b')
text(0,b+.05,c-.1,'E(6)','fontweight','b')

text((a+.1)/3,.3,0,'a','fontweight','b')
text(.05,(b-.1)/2,.17,'b','fontweight','b')
text(-.16,0,(c-.1)/2,'c','fontweight','b')

view(42,34);
% view(AZ,EL) set the angle of the view from
% which an observer sees the current 3-D plot
% AZ is the azimuth or horizontal rotation
% EL is the vertical elevation
% (both in degrees)

% Generate data
vert=...
[0 0 0; a 0 0; 0 b 0; 0 0 c; a 0 c; 0 b c];
% define the matrix of the vertices
% O: 0,0,0 defined as vertex 1
% A: a,0,0 defined as vertex 2
% B: 0,b,0 defined as vertex 3
% C: 0,0,c defined as vertex 4
% D: a,0,c defined as vertex 5
% E: 0,b,c defined as vertex 6

face_up=[1 2 3; 4 5 6];
% define the lower and upper face of
% the triangular prism
% lower face is defined by vertices
% 1, 2, 3 (O, A, B)
% upper face is defined by vertices
% 4, 5, 6 (C, D, E)

face_l=[1 2 5 4; 2 3 6 5; 1 3 6 4];
```

```
% generate the lateral faces
% lateral face 1 is defined by 1, 2, 5, 4
% lateral face 2 is defined by 2, 3, 6, 5
% lateral face 3 is defined by 1, 3, 6, 4
% when defined a face the order of the vertices
% has to be given clockwise or counterclockwise

% draw the lower and upper triangular patches
patch...
('Vertices',vert,'Faces',face_up,'facecolor','b')
% patch(x,y,C) adds the "patch" or
% filled 2-D polygon defined by
% vectors x and y to the current axes.
% C specifies the color of the face(s)
% X represents the matrix vert
% Y represents the matrix face_up

% draw the lateral rectangular patches
patch...
('Vertices',vert,'Faces',face_l,'facecolor','b')

quiver3 ...
(0,b,F+c,0,0,-F,1,'Color','r','LineWidth',1.75)
text ...
(-.3,b,c+.2,' F','fontsize',14,'fontweight','b')

quiver3(a,0,0,MBn_(1),MBn_(2),MBn_(3),1,...
    'Color','k','LineWidth',2)
text((a+MBn_(1))/2,MBn_(2)/2,MBn_(3)/2,...
    '  M','fontsize',14,'fontweight','b')

quiver3 ...
(a,0,0,MBn_(1),0,0,1,'Color','r','LineWidth',2)
text((a+MBn_(1))/1.3,0,0,...
    ' M_x','fontsize',14,'fontweight','b')

quiver3 ...
(a,0,0,0,MBn_(2),0,1,'Color','r','LineWidth',2)
text(a+.3,MBn_(2),0,...
    'M_y','fontsize',14,'fontweight','b')

light('Position',[1 2 3]);
% light('PropertyName',propertyvalue,...)
% light creates a light object in current axes
% Lights affect only patch and surface objects
```

```
% light the peaks surface plot with a light source
% located at infinity and oriented along the
% direction defined by the vector [1 2 3]


material shiny

% material shiny makes the objects shiny

alpha('color');
% alpha get or set alpha properties for
% objects in the current axis
% alpha('color') set the alphadata to be
% the same as the color data.

% end of program
```

2.7.4 Program 2.4

```
% example 2.4
clear all; clc; close all
syms F l theta alfa real
sl = {F, l, theta, alfa};
nl = {100, 1, pi/4, pi/1.8};
FA_ = [-F*cos(alfa-theta), F*sin(alfa-theta), 0];
rA_ = [l*cos(theta), l*sin(theta), 0];
FAn_= subs(FA_, sl, nl);
fprintf('R_ = [%6.3f %6.3f %g](lb)\n', FAn_)
MO_ = cross(rA_, FA_);
MOz= simplify(MO_(3));
MOzn= subs(MOz, sl, nl);
fprintf('MOz = %s ',char(MOz))
fprintf('= %6.3f (lb ft)\n',MOzn)

% end of program
```

2.7.5 Program 2.5

```
% example 2.5
clear all; clc; close all
```

```
syms a b FA FB FC x z M
% a)
sl = {a, b, FA, FB, FC};
nl = {4, 6, 0.9, 0.5, 0.3};
FA_ = [0 0 -FA]; rA_ = [a/2 0 0];
FB_ = [0 FB  0]; rB_ = [a 0 b];
FC_ = [FC 0  0]; rC_ = [0 0 b];
R_ = FA_+FB_+FC_;
Rn_ = subs(R_, sl, nl);
uR_ = R_/magn(R_);
uRn_ = subs(uR_, sl, nl);
fprintf('R_ = [%6.3f %6.3f %6.3f] (kip)\n', Rn_)
fprintf('|R_| = %6.3f (kip)\n', magn(Rn_))
fprintf('uR_ = [%6.3f %6.3f %6.3f]\n\n', uRn_)

% b)
rT_ = [x 0 z];
MTA_ = cross(rA_-rT_, FA_);
MTB_ = cross(rB_-rT_, FB_);
MTC_ = cross(rC_-rT_, FC_);
MT_ = MTA_ + MTB_ + MTC_;
eq1 = MT_(1)/M - uR_(1);
eq2 = MT_(2)/M - uR_(2);
eq3 = MT_(3)/M - uR_(3);
eq1n = subs(eq1, sl, nl);
eq2n = subs(eq2, sl, nl);
eq3n = subs(eq3, sl, nl);
digits(3)
fprintf('first equation:\n')
pretty(eq1)
fprintf('%s = 0 \n\n',char(vpa(eq1n)))
fprintf('second equation:\n')
pretty(eq2)
fprintf('%s = 0 \n\n',char(vpa(eq2n)))
fprintf('third equation:\n')
pretty(eq3)
fprintf('%s = 0 \n\n',char(vpa(eq3n)))
sol = solve(eq1, eq2, eq3,'x, z, M');
Ms = sol.M; Mn = subs(Ms, sl, nl);
xs = sol.x; xn = subs(xs, sl, nl);
zs = sol.z; zn = subs(zs, sl, nl);
fprintf('M = ')
pretty(Ms)
fprintf('M = %6.3f (kip ft)\n', Mn)
fprintf('x = ')
```

```
pretty(xs)
fprintf('x = %6.3f (ft)\n', xn)
fprintf('z = ')
pretty(zs)
fprintf('z = %6.3f (ft)\n', zn)

a=4; b=6;

axis([-2*a 2*a  -b b -2*b 2*b])
xlabel('x'), ylabel('y'), zlabel('z')
hold on, grid on

% Cartesian axes
quiver3(0,0,0,2*a,0,0,1,'Color','b')
text(2*a,0,0,' x')
quiver3(0,0,0,0,b,0,1,'Color','b')
text(0,b,0,' y')
quiver3(0,0,0,0,0,2*b,1,'Color','b')
text(0,0,2*b,' z')

xA=a/2; yA=0; zA=0;
xB=a; yB=0; zB=b;
xC=0; yC=0; zC=b;
xD=a; yD=0; zD=0;
xT=xn; yT=0; zT=zn;

line([0 xC],[0 yC],[0,zC],...
    'Color','b','LineWidth',2)
line([0 xD],[0 yD],[0,zD],...
    'Color','b','LineWidth',2)
line([xD xB],[yD yB],[zD,zB],...
    'Color','b','LineWidth',2)
line([xC xB],[yC yB],[zC,zB],...
    'Color','b','LineWidth',2)

text(0,0,0,' O')
text(xA,yA,zA,' A')
text(xB,yB,zB,' B')
text(xC,yC,zC,' C')
text(xD,yD,zD,' D')
text(xT,yT,zT-1,' T')

fs=10; % force scale
FAn_ = fs*subs(FA_, sl, nl);
FBn_ = fs*subs(FB_, sl, nl);
```

```
FCn_ = fs*subs(FC_, sl, nl);
Rtn_ = fs*Rn_;
Mtn_ = fs*Mn*uRn_;

quiver3...
(xA,yA,zA,FAn_(1),FAn_(2),FAn_(3),1,...
'Color','k','LineWidth',2)
quiver3...
(xB,yB,zB,FBn_(1),FBn_(2),FBn_(3),1,...
'Color','k','LineWidth',2)
quiver3...
(xC,yC,zC,FCn_(1),FCn_(2),FCn_(3),1,...
'Color','k','LineWidth',2)
quiver3...
(xT,yT,zT,Rtn_(1),Rtn_(2),Rtn_(3),1,...
'Color','r','LineWidth',2)
quiver3...
(xT,yT,zT,Mtn_(1),Mtn_(2),Mtn_(3),1,...
'Color','G','LineWidth',2)

text(xA+FAn_(1),yA+FAn_(2),zA+FAn_(3),...
   ' F_A','fontsize',12,'fontweight','b')
text(xB+FBn_(1),yB+FBn_(2),zB+FBn_(3),...
   ' F_B','fontsize',12,'fontweight','b')
text(xT+Rtn_(1),yT+Rtn_(2),zT+Rtn_(3),...
   ' R','fontsize',14,'fontweight','b')
text(xT+Mtn_(1),yT+Mtn_(2),zT+Mtn_(3),...
   ' M','fontsize',14,'fontweight','b')

view(-68,30);

% end of program
```

References

1. P. Appell, *Traité de mécanique rationnelle* (Gauthier-Villars, Paris, 1955)
2. M. Atanasiu, *Mechanics* (EDP, Bucharest, 1973)
3. H. Baruh, *Analytical Dynamics* (WCB/McGraw-Hill, Boston, 1999)
4. F.P. Beer, E.R. Johnston, Vector Mechanics for Engineers: Statics and Dynamics 5/e (McGraw-Hill Publishing Company, New York, 1988)
5. F.P. Beer, E.R. Johnston, D.F. Mazurek, Vector Mechanics for Engineers: Statics, 10/e (McGraw-Hill Publishing Company, New York, 2012)
6. A.M. Bedford, W. Fowler, K.M. Liechti, *Statics and Mechanics of Materials* (Prentice Hall, Inc., Upper Saddle River, 2002)

7. A.M. Bedford, W. Fowler, *Engineering Mechanics: Statics, 5/e* (Prentice Hall, Inc., Upper Saddle River, 2007)
8. A.P. Boresi, R.J. Schmidt, Engineering Mechanics: Statics (PWS Publishing Company, Boston, 2000)
9. M.I. Buculei, *Mechanics* (University of Craiova Press, Craiova, 1974)
10. M.I. Buculei, D. Bagnaru, G. Nanu, D.B. Marghitu, *Analysis of Mechanisms with Bars* (Scrisul romanesc, Craiova, 1986)
11. I. Stroe et al., Analytical Mechanics Problems (University Politehnica of Bucharest, Bucharest, 1997)
12. V. Ceausu, N. Enescu, F. Ceausu, Mechanics Problems, (Printech, Bucharest, 1999)
13. S.J. Chapman, MATLAB Programming for Engineers (Thomson Learning, Pacific Grove, CA, 2002)
14. S. Attaway, MATLAB: A Practical Introduction to Programming and Problem Solving, Butterworth-Heinemann (Elsevier, Amsterdam, 2012)
15. D.M. Etter, D.C. Kuncicky, *Introduction to MATLAB for Engineers and Scientists* (Prentice Hall, Inc., Upper Saddle River, 1996)
16. C. Iacob, *Theoretical Mechanics* (EDP, Bucharest, 1980)
17. J.H. Ginsberg, *Advanced Engineering Dynamics* (Cambridge University Press, Cambridge, 1995)
18. D.T. Greenwood, *Principles of Dynamics* (Prentice-Hall, Inc., Englewood Cliffs, 1998)
19. L.E. Goodman, W.H. Warner, *Statics* (Dover Publications, Inc, 2001)
20. R.C. Hibbeler, *Engineering Mechanics: Statics and Dynamics 13/e* (Prentice-Hall, Inc., Upper Saddle River, 2013)
21. T.R. Kane, *Analytical Elements of Mechanics*, vol. 1 (Academic Press, New York, 1959)
22. T.R. Kane, *Analytical Elements of Mechanics*, vol. 2 (Academic Press, New York, 1961)
23. T.R. Kane, D.A. Levinson, *Dynamics* (McGraw-Hill, New York, 1985)
24. R. Maeder, *Programming in Mathematica* (Addison-Wesley Publishing Company, Redwood City, 1990)
25. N.H. Madsen, Statics and Dynamics, class notes, available at http://www.eng.auburn.edu/users/nmadsen/
26. D.B. Marghitu, *Mechanical Engineer's Handbook* (Academic Press, San Diego, 2001)
27. D.B. Marghitu, M.J. Crocker, *Analytical Elements of Mechanisms* (Cambridge University Press, Cambridge, 2001)
28. D.B. Marghitu, *Kinematic Chains and Machine Component Design* (Elsevier, Amsterdam, 2005)
29. D.B. Marghitu, Mechanisms and Robots Analysis with MATLAB (Springer, London, 2009)
30. D.B. Marghitu, M. Dupac, Advanced Dynamics: Analytical and Numerical Calculations with MATLAB (Springer, New York, 2012)
31. D.B. Marghitu, Statics and Dynamics, class notes, available at http://www.eng.auburn.edu/users/marghitu/
32. D.J. McGill, W.W. King, *Engineering Mechanics: Statics and an Introduction to Dynamics* (PWS Publishing Company, Boston, 1995)
33. J.L. Meriam, L.G. Kraige, *Engineering Mechanics: Statics, 7/e* (Wiley, New York, 2011)
34. R.L. Mott, *Machine Elements in Mechanical Design* (Prentice Hall, Upper Saddle River, 1999)
35. R.L. Norton, *Machine Design* (Prentice-Hall, Upper Saddle River, 1996)
36. M. Plesha, G. Gray, F. Costanzo, Engineering Mechanics: Statics, 2/e, McGraw-Hill Publishing Company (2012)
37. M. Radoi, E. Deciu, *Mechanics* (EDP, Bucharest, 1981)
38. W.F. Riley, L.D. Sturges, Engineering Mechanics: Statics, 2/e (Wiley, New York, 1995)
39. A. Ruina, R. Pratap, Introduction to Statics and Dynamics (Oxford University Press, Oxford, 2002)
40. A. Ripianu, P. Popescu, B. Balan, *Technical Mechanics* (EDP, Bucharest, 1979)
41. I.H. Shames, Engineering Mechanics Statics, 4/e, Prentice Hall, Inc. (1996)

42. S.D. Sheppard, B.H. Tongue, *Statics: Analysis and Design of Systems in Equilibrium* (Wiley, New York, 2005)
43. D. Smith, *Engineering Computation with MATLAB* (Pearson Education, Upper Saddle River, 2008)
44. R.W. Soutas-Little, D.J. Inman, *Engineering Mechanics: Statics and Dynamics* (Prentice-Hall, Upper Saddle River, 1999)
45. R.W. Soutas-Little, D.J. Inman, D. Balint, Engineering Mechanics (Statics, Cengage, Learning, 2007)
46. S. Staicu, *Theoretical Mechanics* (EDP, Bucharest, 1998)
47. A. Stan, M. Grumazescu, *Mechanics Problems* (EDP, Bucharest, 1973)
48. J. Sticklen, M.T. Eskil, *An Introduction to Technical Problem Solving with MATLAB* (Great Lakes Press, Wildwood, 2006)
49. A. Stoenescu, G. Silas, *Theoretical Mechanics* (ET, Bucharest, 1957)
50. J.H. Jackson, H.G. Wirtz, Schaum's Outline of Theory and Problems of Statics and Strength of Materials (McGraw-Hill Publishing Company, New York, 1983)
51. The MathWorks: https://www.mathworks.com/
52. Statics eBook : https://ecourses.ou.edu/
53. R. Voinea, D. Voiculescu, V. Ceausu, *Mechanics* (EDP, Bucharest, 1983)
54. V. Valcovici, S. Balan, R. Voinea, *Theoretical Mechanics* (ET, Bucharest, 1959)
55. K.J. Waldron, G.L. Kinzel, *Kinematics, Dynamics, and Design of Machinery* (Wiley, New York, 1999)
56. H.B. Wilson, L.H. Turcotte, *D* Halpern, Advanced Mathematics and Mechanics Applications Using MATLAB (Chapman & Hall/CRC, Boca Raton, FL, 2003)
57. J.H. Williams, *Fundamentals of Applied Dynamics* (Wiley, New York, 1996)
58. S. Wolfram, *Mathematica* (Wolfram Media/Cambridge University Press, Cambridge, 1999)

Chapter 3
Centers of Mass

3.1 First Moment

Figure 3.1 shows a set of n points P_i, $\{S\} = \{P_1,\ P_2, \ldots, P_n\} = \{P_i\}_{i=1,2,\ldots,n}$. The position vector of a point P_i relative to an arbitrarily selected reference point O is $\mathbf{r}_{P_i}$, where $\mathbf{r}_{P_i} = \mathbf{r}_i$. A scalar s_i can be associated with P_i as for example the mass m_i of a particle situated at P_i. The *first moment* of a point P_i with respect to a point O is the vector $\mathbf{M}_i = s_i\,\mathbf{r}_{P_i}$. The scalar s_i is called the *strength* of P_i. The strengths of the points P_i are $s_i,\ i = 1, 2, \ldots, n$ and are n scalars, having the same dimension, and associated with one of the points of $\{S\}$.

The *centroid* of the set $\{S\}$ is the point C with respect to which the sum of the first moments of the points of $\{S\}$ is equal to zero. The position vector of C relative to a point O is $\mathbf{r}_C$. The position vector of P_i relative to C is $\mathbf{r}_i - \mathbf{r}_C$. The sum of the first moments of the points P_i with respect to C is $\sum\limits_{i=1}^{n} s_i(\mathbf{r}_i - \mathbf{r}_C)$. If C is the centroid of $\{S\}$ then

$$\sum_{i=1}^{n} s_i(\mathbf{r}_i - \mathbf{r}_C) = \sum_{i=1}^{n} s_i\mathbf{r}_i - \mathbf{r}_C\sum_{i=1}^{n} s_i = 0.$$

The position vector $\mathbf{r}_C$ of the centroid C is given by

$$\mathbf{r}_C = \frac{\sum\limits_{i=1}^{n} s_i\,\mathbf{r}_i}{\sum\limits_{i=1}^{n} s_i}.$$

If $\sum_{i=1}^{n} s_i = 0$ the centroid is not defined. The centroid C of a set of points of given strength does not depend on the choice of the reference point O.

D. B. Marghitu et al., *Statics with MATLAB®*,
DOI: 10.1007/978-1-4471-5110-4_3, © Springer-Verlag London 2013

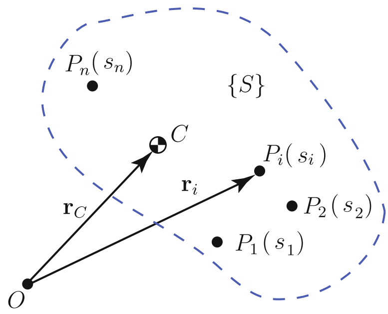


Fig. 3.1 Set of points and centroid of a set of points

3.2 Center of Mass of a Set of Particles

The *center of mass* of a set of particles $\{S\} = \{P_1, P_2, \ldots, P_n\} = \{P_i\}_{i=1,2,\ldots,n}$ is the centroid of the set of points with $s_i = m_i,\ i = 1, 2, \ldots, n$, where m_i is the mass of the particle P_i. The position vector of the center of mass, C, of the system with n particles is

$$\mathbf{r}_C = \frac{\sum_{i=1}^{n} m_i \, \mathbf{r}_i}{\sum_{i=1}^{n} m_i} = \frac{\sum_{i=1}^{n} m_i \, \mathbf{r}_i}{M}, \tag{3.1}$$

where M is the total mass of the system.

3.3 Center of Mass of a Body

The position vector of the center of mass C, Fig. 3.2, of a body of mass m and volume V relative to a point O is

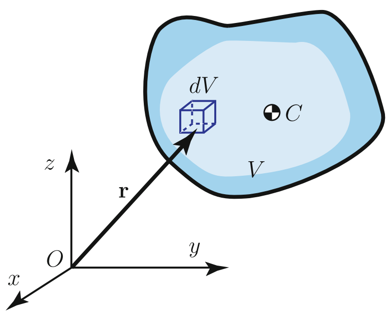


Fig. 3.2 Center of mass of a volume V

$$\mathbf{r}_C = \frac{\iiint\limits_V \mathbf{r}\,\mathrm{d}m}{\iiint\limits_V \mathrm{d}m}. \tag{3.2}$$

The mass of a differential element of volume $\mathrm{d}V$ is $\mathrm{d}m = \rho\,\mathrm{d}V$ where ρ is the density of a body (mass per unit volume). The orthogonal cartesian coordinates of C are

$$x_C = \frac{\iiint\limits_V x\,\rho\mathrm{d}V}{\iiint\limits_V \rho\mathrm{d}V},\ y_C = \frac{\iiint\limits_V y\,\rho\mathrm{d}V}{\iiint\limits_V \rho\mathrm{d}V},\ z_C = \frac{\iiint\limits_V z\,\rho\mathrm{d}V}{\iiint\limits_V \rho\mathrm{d}V}. \tag{3.3}$$

The center of mass of a body is the point at which the total moment of the body's mass about that point is zero. If the mass density ρ of the body is the same at all points of the body, the body is uniform and the coordinates of the center of the mass C are

$$x_C = \frac{\iiint\limits_V x\,\mathrm{d}V}{\iiint\limits_V \mathrm{d}V},\ y_C = \frac{\iiint\limits_V y\,\mathrm{d}V}{\iiint\limits_V \mathrm{d}V},\ z_C = \frac{\iiint\limits_V z\,\mathrm{d}V}{\iiint\limits_V \mathrm{d}V}. \tag{3.4}$$

For a uniform curve $\rho = \rho_l = m/L$ is the mass per unit of length and

$$x_C = \frac{\int\limits_L x\,\mathrm{d}l}{\int\limits_L \mathrm{d}l},\ y_C = \frac{\int\limits_L y\,\mathrm{d}l}{\int\limits_L \mathrm{d}l},\ z_C = \frac{\int\limits_L z\,\mathrm{d}l}{\int\limits_L \mathrm{d}l}, \tag{3.5}$$

where L is the length of the curve. For a uniform surface $\rho = \rho_s = m/A$ is the mass per unit of area and

$$x_C = \frac{\iint\limits_A x\,\mathrm{d}A}{\iint\limits_A \mathrm{d}A},\ y_C = \frac{\iint\limits_A y\,\mathrm{d}A}{\iint\limits_A \mathrm{d}A},\ z_C = \frac{\iint\limits_A z\,\mathrm{d}A}{\iint\limits_A \mathrm{d}A}, \tag{3.6}$$

where A is the area of the surface.

The *method of decomposition* is used to locate the center of mass of a composite body:

1. divide the body into a number of simpler body shapes, which may be particles, curves, surfaces, or solids; Holes are considered as pieces with negative size, mass, or volume.
2. locate the coordinates $x_{C_i},\ y_{C_i},\ z_{C_i}$ of the center of mass of each part of the body;
3. determine the center of mass using the equations

$$x_C = \frac{\sum_{i=1}^{n} \int_\tau x\, d\tau}{\sum_{i=1}^{n} \int_\tau \mathrm{d}\tau},\ y_C = \frac{\sum_{i=1}^{n} \int_\tau y\, d\tau}{\sum_{i=1}^{n} \int_\tau \mathrm{d}\tau},\ z_C = \frac{\sum_{i=1}^{n} \int_\tau z\, d\tau}{\sum_{i=1}^{n} \int_\tau \mathrm{d}\tau}, \tag{3.7}$$

where τ is a curve, area, or volume. Equation (3.7) can be simplify as

$$x_C = \frac{\sum_{i=1}^{n} x_{C_i}\,\tau_i}{\sum_{i=1}^{n} \tau_i},\ y_C = \frac{\sum_{i=1}^{n} y_{C_i}\,\tau_i}{\sum_{i=1}^{n} \tau_i},\ z_C = \frac{\sum_{i=1}^{n} z_{C_i}\,\tau_i}{\sum_{i=1}^{n} \tau_i}, \tag{3.8}$$

where τ_i is the length, area, or volume of the i^{th} object.

3.4 First Moment of an Area

A planar surface of area A is shown in Fig. 3.3. The first moment of the area A about the x-axis is

$$M_x = \iint_A y\ \mathrm{d}A. \tag{3.9}$$

The first moment about the y-axis is

$$M_y = \iint_A x\ \mathrm{d}A. \tag{3.10}$$

The first moment of area gives information of the shape, size, and orientation of the area. The coordinates x_C and y_C of the center of mass of the area A are calculated with

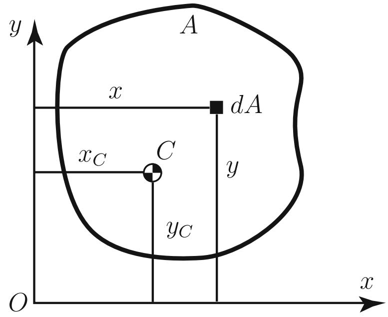


Fig. 3.3 Planar surface of area A

$$x_C = \frac{\displaystyle\iint_A x \, \mathrm{d}A}{A} = \frac{M_y}{A}, \tag{3.11}$$

$$y_C = \frac{\displaystyle\iint_A y \, \mathrm{d}A}{A} = \frac{M_x}{A}. \tag{3.12}$$

The location of the center of mass of an area is independent of the reference axes employed. If the axes xy have their origin at the centroid, $O \equiv C$, then these axes are called *centroidal axes*. The first moments about the centroidal axes are zero. The center of mass of an area with one axis of symmetry is located along the axis of symmetry. The axis of symmetry is a centroidal axis and the first moment of area must be zero about the axis of symmetry. If a body has two orthogonal axes of symmetry the centroid is at the intersection of these axes. For surfaces as circles, rectangles, triangles, the center of mass can be determined by inspection.

3.5 Center of Gravity

The *center of gravity* is a point which locates the resultant weight of a system of particles or body. The sum of moments due to individual particle weight about any point is the same as the moment due to the resultant weight located at the center of gravity. The sum of moments due to the individual particles weights about center of gravity is equal to zero. Similarly, the center of mass is a point which locates the resultant mass of a system of particles or body. The center of gravity of a body is the point at which the total moment of the force of gravity is zero. The coordinates of the center of gravity are

$$x_C = \frac{\iiint\limits_V x\rho\, g\, \mathrm{d}V}{\iiint\limits_V \rho\, g\, \mathrm{d}V},\ y_C = \frac{\iiint\limits_V y\rho\, g\, \mathrm{d}V}{\iiint\limits_V \rho\, g\, \mathrm{d}V},\ z_C = \frac{\iiint\limits_V z\rho\, g\, \mathrm{d}V}{\iiint\limits_V \rho\, g\, \mathrm{d}V}. \tag{3.13}$$

If the acceleration of gravity g is constant throughout the body, then the location of the center of gravity is the same as that of the center of mass. The acceleration of gravity is $g = 9.81$ m/s^2 or $g = 32.2$ ft/s^2.

3.6 Theorems of Guldinus-Pappus

Theorem 1 Consider a planar generating curve and an axis of revolution in the plane of this curve Fig. 3.4. The axis of revolution does not intersect the curve. It can only touch the generating curve. The surface of revolution A developed by rotating the generating curve about the axis of revolution equals the product of the length of the generating L curve times the circumference of the circle formed by the centroid of the generating curve y_C in the process of generating a surface of revolution

$$A = 2\,\pi y_C L. \tag{3.14}$$

Proof A length element dl of the generating curve is considered as shown in Fig. 3.4. For a revolution of the generating curve about axis of revolution, x-axis, the length element dl describes the area

$$\mathrm{d}A = 2\,\pi y \mathrm{d}l.$$

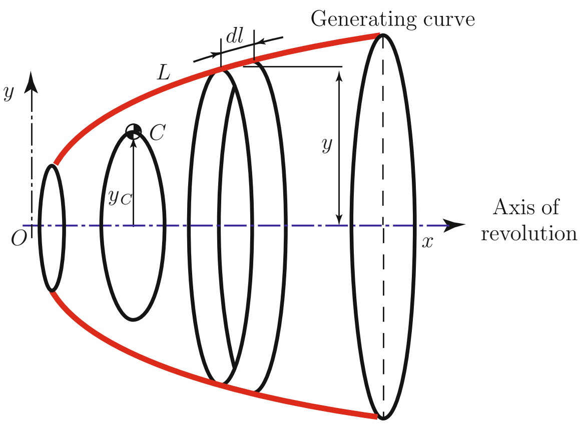


Fig. 3.4 Surface of revolution

For the total surface of revolution developed the area is

$$A = 2\pi \int y\,\mathrm{d}l = 2\pi y_C L,$$

where L is the length of the curve and y_C is the centroidal coordinate of the curve. The circumferential length of the circle formed by having the centroid of the curve rotate about the x-axis is $2\pi y_C$. The surface of revolution A is equal to 2π times the first moment of the generating curve about the axis of revolution. For a composite generating curve the following formula is used

$$A = 2\pi \left(\sum_i L_i y_{C_i} \right), \tag{3.15}$$

where y_{C_i} is the centroidal coordinate for the i^{th} line segment L_i. The generating curve is composed of simple curves, L_i and the axis of revolution is the x-axis.

Theorem 2 A generating planar surface A and an axis of revolution located in the same plane as the surface is considered in Fig. 3.5. The volume of revolution V developed by rotating the generating planar surface about the axis of revolution equals the product of the area of the surface times the circumference of the circle formed by the centroid of the surface y_C in the process of generating the body of revolution

$$V = 2\pi y_C A. \tag{3.16}$$

The axis of revolution does not intersect the generating surface. It can only touch the generating plane surface as a tangent at the boundary.

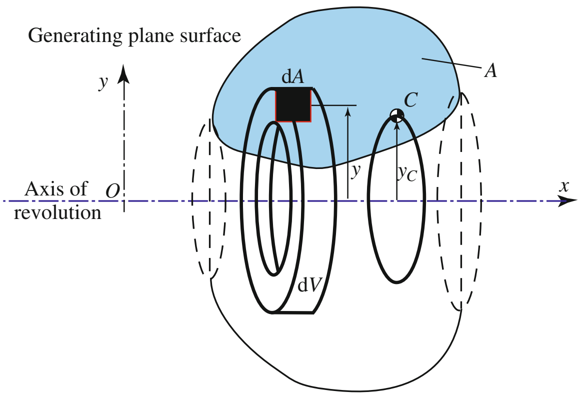


Fig. 3.5 Volume of revolution

Proof The volume generated by rotating an element dA of the plane surface, A is shown in Fig. 3.5, about the x-axis is

$$\mathrm{d}V = 2\,\pi y\,\mathrm{d}A.$$

The volume of the body of revolution formed from A is

$$V = 2\,\pi \int_A y\,\mathrm{d}A = 2\,\pi y_C\,A.$$

Thus, the volume V equals the area of the generating surface A times the circumferential length of the circle of radius y_C. The volume V equals $2\,\pi$ times the first moment of the generating area A about the axis of revolution.

The areas and center of mass for some practical configurations are shown in Fig. 3.6.

3.7 Examples

Example 3.1 Find the length and the position of the center of mass for the homogeneous curve given by the Cartesian equation $y = b\sqrt{x^a}$ m, where $a = 3$, $b = 2$ and $0 \le x \le 1$ m.

Solution The differential element of the curve, $\mathrm{d}l = \sqrt{1 + (\mathrm{d}y/\mathrm{d}x)^2}$, is given in MATLAB by:

```
syms x real
a = 3;
b = 2;
y  = b*sqrt(x^a);
dy = diff(y,x);
dl =sqrt(1+dy^2);
% dl = (1+ (dy/dx)^2)^0.5
%  0< x < 1
```

The MATLAB statement `int (f, x, a, b)` is the definite integral of `f` with respect to its symbolic variable `x` from `a` to `b`. The length of the homogeneous curve is:

```
L = eval(int(dl,0,1));
```

and the coordinates of the center of mass C are:

```
My=eval(int(x*dl,0,1));
xC=My/L;
```

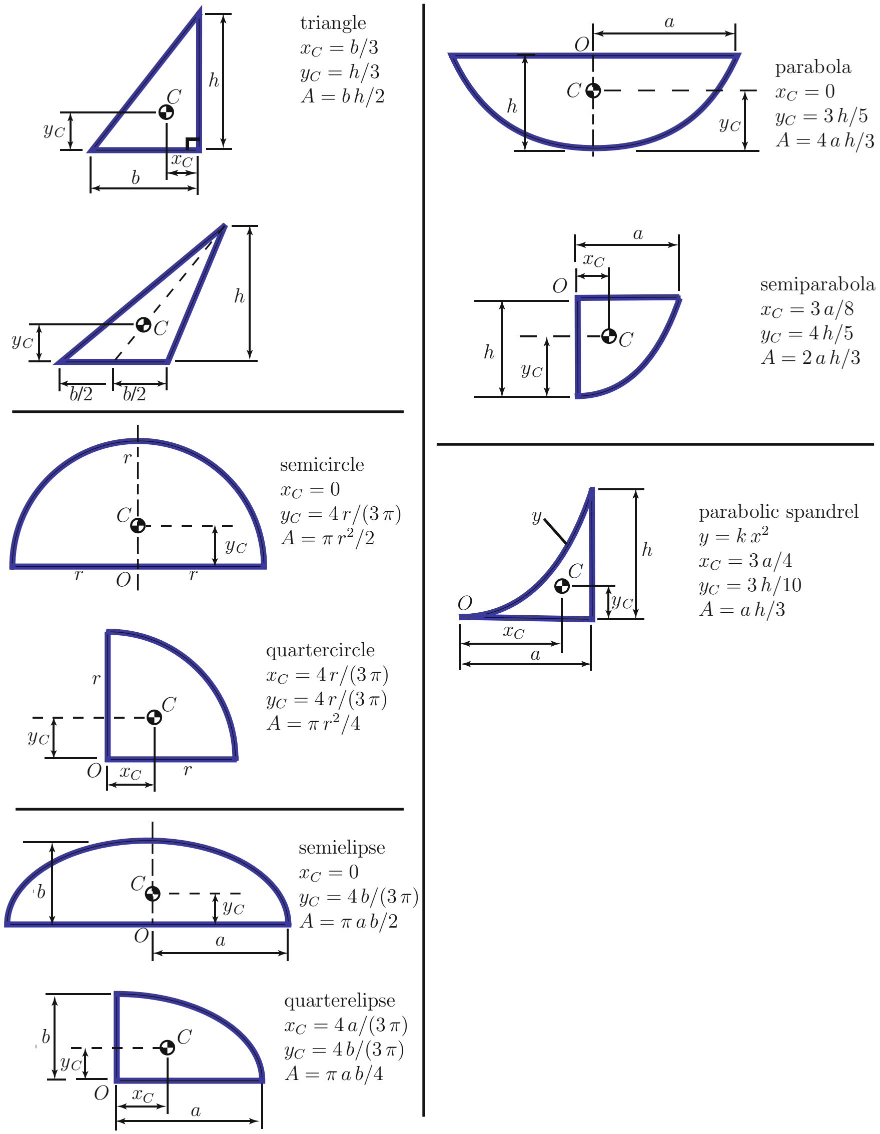


Fig. 3.6 Coordinates of center of mass, x_C and y_C, and area A

```
Mx=eval(int(y*dl,0,1));
yC=Mx/L;
```

The numerical values for the length and the centroid are:

```
L = 2.268 (m)
xC = 0.575 (m)
```

```
yC = 0.952 (m)
```

The MATLAB statements for the graphical representation are:

```
% plot the curve and CM
xf=1;
xn = 0:xf/100:xf;
yn = b*sqrt(xn.^a);
axis ([0 1 0 1])
plot(xn,yn,'-b','LineWidth',2)
hold on
plot(xC,yC,'o','MarkerSize',12,...
               'MarkerEdgeColor','k',...
               'MarkerFaceColor','r')
text(xC,yC,'      C','FontSize',18)
title('y=f(x)=2 x^{3/2}')
```

and the results are depicted in Fig. 3.7.

Example 3.2 A homogeneous circle is given by the Cartesian equation $x^2+y^2 = r^2$, where $r = 1$ m. (a) Find the length of the homogeneous circle. (b) Find the length and the position of the center of mass for the homogeneous semi-circle, $-1 \leq x \leq 1$ and $0 \leq y \leq 1$. (c) Find the length and the position of the center of mass for the homogeneous quarter-circle, $0 \leq x \leq 1$ and $0 \leq y \leq 1$.

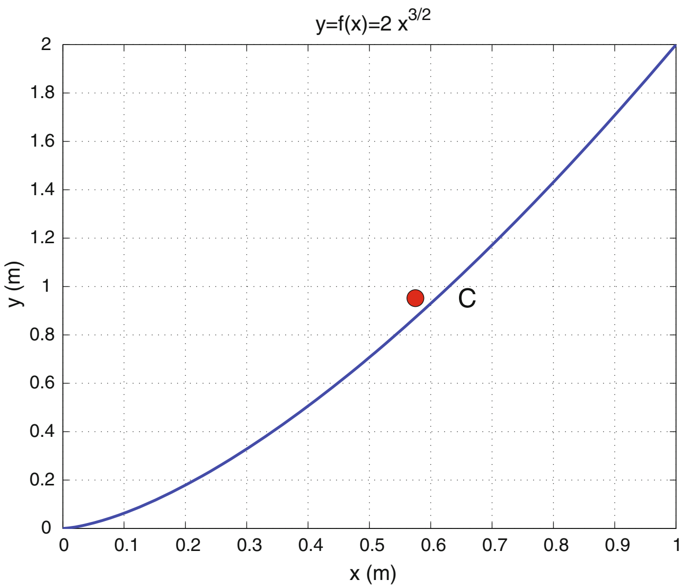


Fig. 3.7 Example 3.1

Solution (a) The parametric equations for the circle are:

```
syms r t real
x = r*cos(t);
y = r*sin(t);
% 0 < t < 2*pi
% r > 0
```

The differential arc length is calculated in MATLAB with:

```
dx = diff(x,t);
dy = diff(y,t);
% dl = ((dx/dt)^2+(dy/dt)^2)^0.5 dt
dl = (dx^2+dy^2)^0.5;
dl = simplify(dl);
```

and the result is:

```
dl = abs(r) dt
```

The length of the circle is given by:

```
L = int(dl,t,0,2*pi);
```

and the MATLAB result is:

```
L = 2*pi*abs(r)
```

(b) For the semi-circle the length and the center of mass are:

```
Ls  = int(dl,t,0,pi);
Mys = int(x*dl,t,0,pi);
xCs = simplify(Mys/Ls);
Mxs = int(y*dl,t,0,pi);
yCs = simplify(Mxs/Ls);
```

The results are:

```
Ls = pi*abs(r)
xCs = 0
yCs = (2*r)/pi
```

(c) For the quarter-circle the length and the center of mass are:

```
Lq  = int(dl,t,0,pi/2);
Myq = int(x*dl,t,0,pi/2);
xCq = simplify(Myq/Lq);
Mxq = int(y*dl,t,0,pi/2);
yCq = simplify(Mxq/Lq);
```

and the MATLAB results are:

```
Lq = (pi*abs(r))/2
xCq = (2*r)/pi
yCq = (2*r)/pi
```

The MATLAB statements for the semi-circle and the quarter-circle graphical representation are:

```
rn=1;
% plot the semi-circle and CM
figure(1)
xCsn = subs(xCs,r,1);
yCsn = subs(yCs,r,1);
tn = 0:pi/18:pi;
xn = rn*cos(tn);
yn = rn*sin(tn);
axis manual
axis equal
hold on
grid on
sa = 1;
axis ([-sa sa -sa sa])
plot(xn,yn,'-b','LineWidth',2)
text(0,0,' O','fontsize',14)
line([-sa,sa],[0,0],'Color','k')
line([0,0],[0,sa],'Color','k')
plot(xCsn,yCsn,'o','MarkerSize',12,...
               'MarkerEdgeColor','k',...
               'MarkerFaceColor','r')
% plot the quarter-circle and CM
figure(2)
xCqn = subs(xCq,r,1);
yCqn = subs(yCq,r,1);
tn = 0:pi/18:pi/2;
xn = rn*cos(tn);
yn = rn*sin(tn);
axis manual
axis equal
hold on
grid on
sa = 1;
axis ([-sa sa -sa sa])
plot(xn,yn,'-b','LineWidth',2)
text(0,0,' O','fontsize',14)
line([0,sa],[0,0],'Color','k')
line([0,0],[0,sa],'Color','k')
plot(xCqn,yCqn,'o','MarkerSize',12,...
```

```
                    'MarkerEdgeColor','k',...
                    'MarkerFaceColor','r')
text(xCqn,yCqn,'  C','FontSize',18)
```

The graphics are depicted in Fig. 3.8.

Example 3.3 A homogeneous quarter-astroid (one cusp) is given by the Cartesian equation $x^{2/3} + y^{2/3} = a^{2/3}$, where $a = 1$ m and $0 \le x \le 1$. Find the length and the position of the center of mass for the homogeneous given curve.

Solution The parametric equations for the astroid are:

```
syms t real
a = 1; % (m)
x = a*cos(t)^3;
y = a*sin(t)^3;
% 0 < t < pi/2 - quarter-astroid
```

The differential arc length, $\mathrm{d}l = \sqrt{(\mathrm{d}x/\mathrm{d}t)^2 + (\mathrm{d}y/\mathrm{d}t)^2)}\,\mathrm{d}t$, is calculated in MATLAB with:

```
dx = diff(x,t);
dy = diff(y,t);
% dl=((dx/dt)^2+(dy/dt)^2)^0.5 dt
dl = (dx^2+dy^2)^0.5;
dl = simplify(dl);
```

and the result is:

```
dl = (3*(sin(2*t)^2)^(1/2))/2
```

The length of the quarter-astroid is given by:

```
L = int(dl,t,0,pi/2);
L = double(L);
```

and the MATLAB result is:

```
L = 6*a/4
L = 1.500 (m)
```

For the quarter-astroid the length and the center of mass are:

```
My = int(x*dl,t,0,pi/2);
xC = My/L;
xC = double(xC);
Mx = int(y*dl,t,0,pi/2);
yC = Mx/L;
yC = double(yC);
```

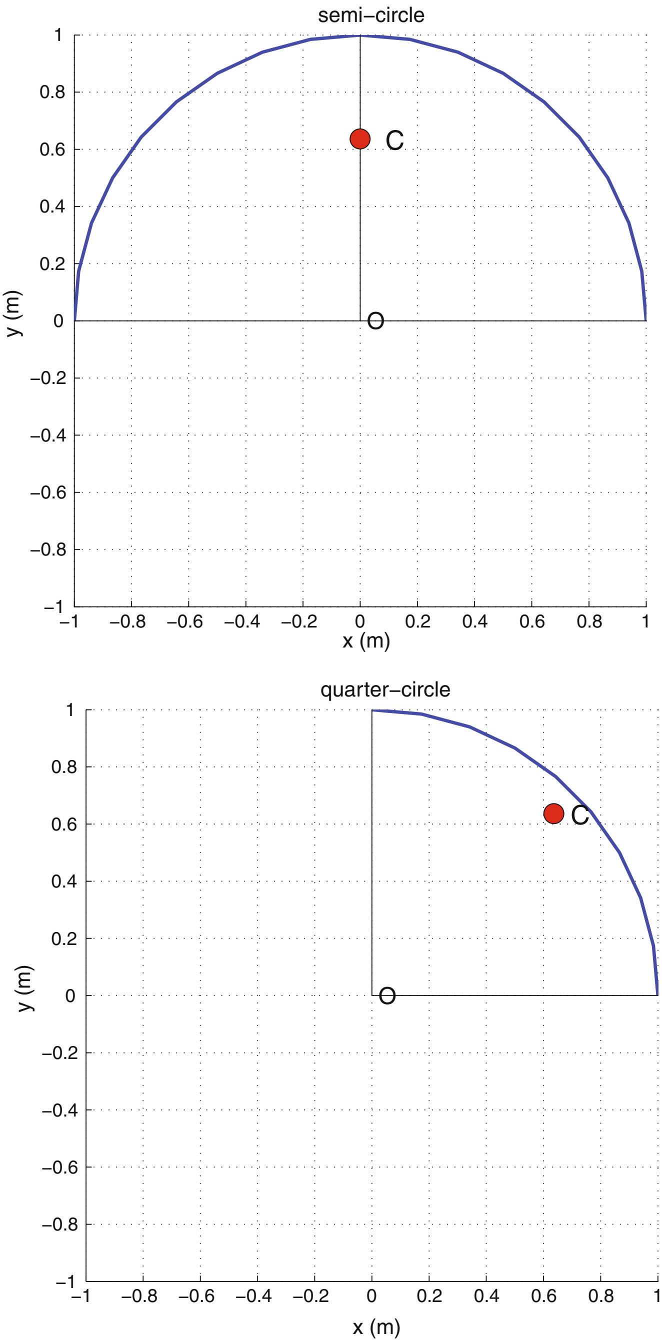


Fig. 3.8 Example 3.2

and the MATLAB results are:

```
xC = 0.400 (m)
yC = 0.400 (m)
```

The MATLAB statements for the semi-circle and the quarter-circle graphical representation are:

```
tn = 0:pi/18:pi/2;
xn = a*cos(tn).^3;
yn = a*sin(tn).^3;
sa = 1;
axis ([0 sa 0 sa])
plot(xn,yn,'-b','LineWidth',2)
text(0,0,' O','fontsize',14)
line([0,sa],[0,0],'Color','k')
line([0,0],[0,sa],'Color','k')
plot(xC,yC,'o','MarkerSize',12,...
               'MarkerEdgeColor','k',...
               'MarkerFaceColor','r')
```

The graphics are depicted in Fig. 3.9.

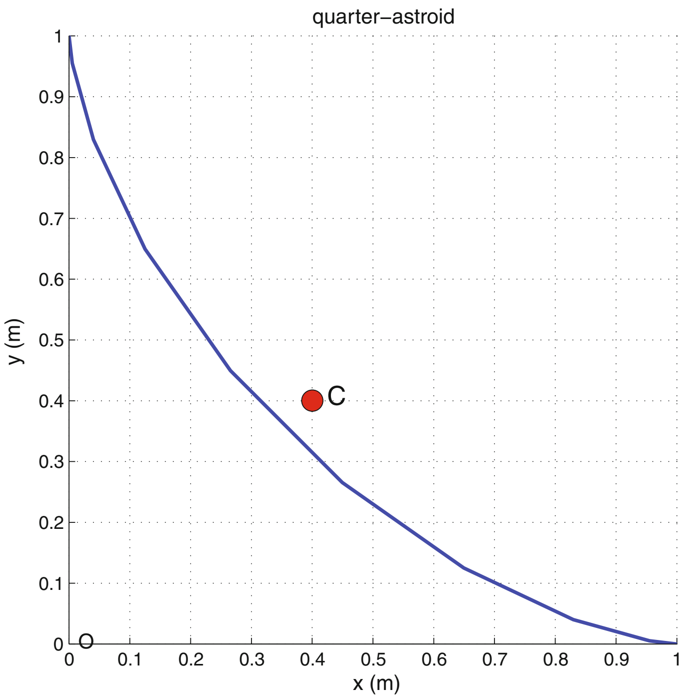


Fig. 3.9 Example 3.3

Example 3.4 A homogeneous circular helix is given by the Cartesian equation

$$x = a \cos t;\ \ y = a \sin t;\ \ \text{and } z = h\,t,$$

where $a = 1$ m is the radius of the helix and $2\,\pi\,h$ is the pitch of the helix, $h = 1$ m. Find the length and the position of the center of mass for the spatial homogeneous helix.

Solution The differential arc length for the spatial curve is

$$\mathrm{d}l = \sqrt{(\mathrm{d}x/\mathrm{d}t)^2 + (\mathrm{d}y/\mathrm{d}t)^2 + (\mathrm{d}z/\mathrm{d}t)^2}\,\mathrm{d}t$$

and is calculated in MATLAB with:

```
syms a h t real
x = a*cos(t);
y = a*sin(t);
z = h*t;
dx = diff(x,t);
dy = diff(y,t);
dz = diff(z,t);
%dl=((dx/dt)^2+(dy/dt)^2+(dz/dt)^2)^0.5 dt
dl = (dx^2+dy^2+dz^2)^0.5;
```

The MATLAB result for the differential arc length is:

```
dl = (a^2 + h^2)^(1/2) dt
```

The length of the helix is calculated with:

```
tf = 6*pi;
L = int(dl,t,0,tf);
```

and the coordinates of the center of mass are:

```
xC = int(x*dl,t,0,tf)/L;
yC = int(y*dl,t,0,tf)/L;
zC = int(z*dl,t,0,tf)/L;
```

The numerical results are:

```
L = 26.657 (m)
xC = 0.000 (m)
yC = 0.000 (m)
zC = 9.425 (m)
```

The MATLAB statements for the helix graphical representation are:

```
tn = 0:pi/50:tf;
plot3(sin(tn),cos(tn),tn)
```

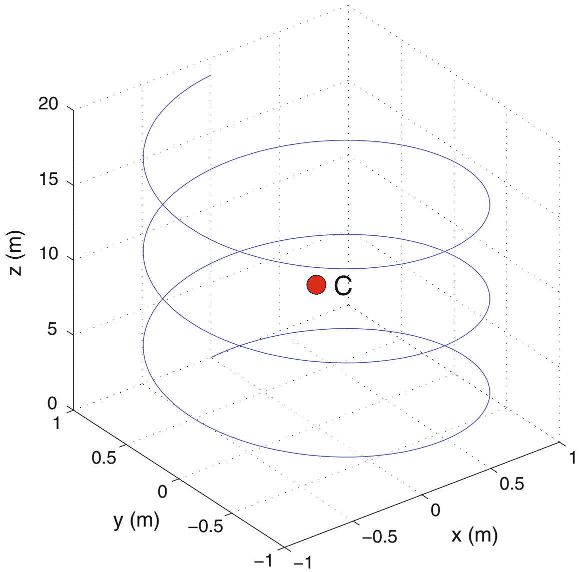


Fig. 3.10 Example 3.4

```
plot3(xC,yC,zC,...
'o','MarkerSize',12,...
'MarkerEdgeColor','k',...
'MarkerFaceColor','r')
```

and the graphics are depicted in Fig. 3.10.

Example 3.5 A homogeneous spatial curve is given by the Cartesian equation

$$x = ae^{kt}\ \cos\ t\ \text{(m)};\ \ y = ae^{kt}\ \sin\ t\ \ \text{(m)};\ \ \text{and}\ z = ae^{kt}\ \ \text{(m)},$$

where $a = 2$ and $k = 1$. Find the length and the position of the center of mass for the spatial homogeneous curve for $t \in [0,\ 3]$.

Solution The differential arc length for the spatial curve is

$$\mathrm{d}l = \sqrt{(\mathrm{d}x/\mathrm{d}t)^2 + (\mathrm{d}y/\mathrm{d}t)^2 + (\mathrm{d}z/\mathrm{d}t)^2}\,\mathrm{d}t$$

and is calculated in MATLAB with:

```
syms t real
a=2; k=1;
x = a*exp(k*t)*cos(t);
y = a*exp(k*t)*sin(t);
```

```
z = a*exp(k*t);
dx = diff(x,t);
dy = diff(y,t);
dz = diff(z,t);
dl = (dx^2+dy^2+dz^2)^0.5;
```

The MATLAB result for the differential arc length is:

```
dl = 2*3^(1/2)*exp(t) dt
```

The length of the helix is calculated with:

```
tf = 3;
L = int(dl,0,tf);
```

and the numerical value is `L = 66.114 (m)`. The coordinates of the center of mass are:

```
xC = int(x*dl,t,0,tf)/L;
yC = int(y*dl,t,0,tf)/L;
zC = int(z*dl,t,0,tf)/L;
```

The numerical values for C are:

```
xC = -15.590 (m)
yC = 10.778 (m)
zC = 21.086 (m)
```

The MATLAB statements for the curve graphical representation are:

```
tn = 0:pi/100:tf;
xn = a*exp(k*tn).*cos(tn);
yn = a*exp(k*tn).*sin(tn);
zn = a*exp(k*tn);
ht = plot3(xn,yn,zn);
hold on
plot3(xC,yC,zC,...
'o','MarkerSize',12,...
'MarkerEdgeColor','k',...
'MarkerFaceColor','r')
text(xC,yC,zC,'   C','FontSize',18)
grid on
axis square
```

and the graphics are depicted in Fig. 3.11.

Example 3.6 Find the coordinates of the mass center for a homogeneous planar plate located under the line of equation $y = bx/a$ from $x = 0$ to $x = a$. For the numerical application select $a = b = 1$ m.

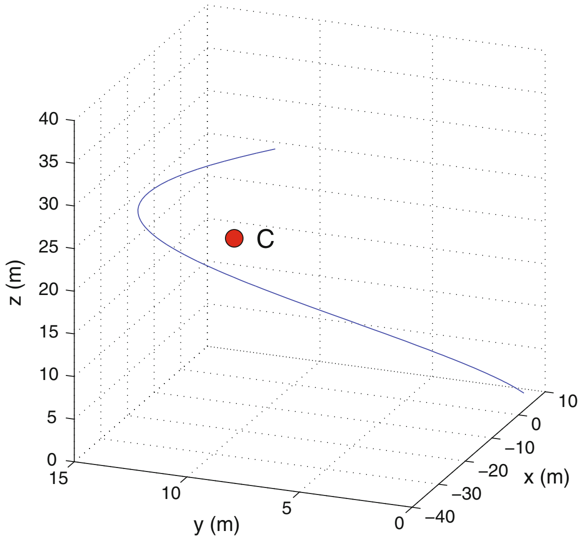


Fig. 3.11 Example 3.5

Solution The differential element of area is $\mathrm{d}A = \mathrm{d}x\,\mathrm{d}y$ and the area of the figure is

$$A = \int_A \mathrm{d}x\,\mathrm{d}y = \int_0^a \int_0^{bx/a} \mathrm{d}x\,\mathrm{d}y = \int_0^a \mathrm{d}x \int_0^{bx/a} \mathrm{d}y$$
$$= \int_0^a \mathrm{d}x\ \{y\}_0^{bx/a} = \int_0^a (bx/a)\mathrm{d}x = \left\{b\,x^2/(2\,a)\right\}_0^a = b\,a/2.$$

The MATLAB program for the area is given by:

```
syms x y a b real
f = b*x/a;
xf = a;
Ay = int(1,y,0,f);
Area = int(Ay,x,0,xf);
```

The first moment of the area A about the y axis is

$$M_y = \int_A x\,\mathrm{d}A = \int_0^a \int_0^{bx/a} x\,\mathrm{d}x\,\mathrm{d}y = \int_0^a x\,\mathrm{d}x \int_0^{bx/a} \mathrm{d}y$$
$$= \int_0^a x\,\mathrm{d}x\ \{y\}_0^{bx/a} = \int_0^a x\,(bx/a)\mathrm{d}x = \int_0^a (bx^2/a)\mathrm{d}x = b\,a^2/3.$$

The x coordinate of the mass center is $x_C = M_y/A = 2\,a/3 = 0.667$ m. The MATLAB program for x_C is :

```
% first moment of area about y-axis
% My = int(x dx dy) where
% 0<x<xf and 0<y<f
% Qyy = int(dy) ; 0<y<f
Qyy = int(1,y,0,f);
% My = int(x Qyy dx) where 0<x<xf
My = int(x*Qyy,x,0,xf);
% centroid xC = My/Area
xC = My/Area;
```

The y coordinate of the mass center is $y_C = M_x/A$, where the first moment of the area A about the x axis is

$$M_x = \int_A y\,\mathrm{d}A = \int_0^a \int_0^{bx/a} y\,\mathrm{d}x\,\mathrm{d}y = \int_0^a \mathrm{d}x \int_0^{bx/a} y\,\mathrm{d}y$$
$$= \int_0^a \mathrm{d}x \left\{\frac{y^2}{2}\right\}_0^{bx/a} = \int_0^a \frac{b^2\,x^2}{2\,a^2}\mathrm{d}x = \frac{b^2}{2\,a^2}\int_0^a x^2\,\mathrm{d}x = \frac{b^2\,a}{6}.$$

The coordinate y_C is

$$y_C = \frac{M_x}{A} = \frac{b}{3} = 0.333 \text{ m}.$$

The MATLAB program for y_C is :

```
Qxy = int(y,y,0,f);
% Mx = int(x Qxy dx) where 0<x<xf
Mx = int(Qxy,x,0,xf);
yC = Mx/Area;
```

The MATLAB statements for the graphical representation are:

```
ls = {a,b};
ln = {1,1};
xfn = subs(xf,ls,ln);
xCn = subs(xC,ls,ln);
yCn = subs(yC,ls,ln);
sa = 1.5;
axis([0 sa 0 sa])
hold on, grid on
xx = 0:.1:xfn;
fx = subs(f,{a,b,x},{1,1,xx});
plot(xx,fx,'--','LineWidth',2)
```

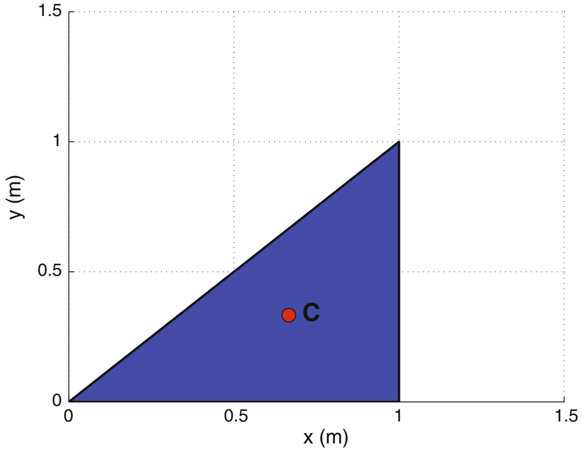


Fig. 3.12 Example 3.6

```
hold on
area(xx,fx,'FaceColor','b',...
           'EdgeColor','k',...
           'LineWidth',2)
hold on
plot(xCn,yCn,...
'o','MarkerSize',12,...
'MarkerEdgeColor','k',...
'MarkerFaceColor','r')
xlabel('x'), ylabel('y')
text(xCn,yCn,'C','fontsize',14,'fontweight','b')
```

and the graphic is shown in Fig. 3.12.

Example 3.7 Find the coordinates of the mass center for a homogeneous planar plate located under the curve of equation $y = A\ \sin(kx)$ from $x = 0$ to $x = 3\,\pi/(4\,k)$. For the numerical application use $A = 1.5$ m and $k = 0.75$ m^{-1}.

Solution The differential element of area is $\mathrm{d}A = \mathrm{d}x\,\mathrm{d}y$ and the area of the figure is

$$Area = \int_A \mathrm{d}x\,\mathrm{d}y = \int_0^{3\,\pi/(4\,k)} \int_0^{A\sin(k\,x)} \mathrm{d}x\,\mathrm{d}y = \int_0^{3\,\pi/(4\,k)} \mathrm{d}x \int_0^{A\ \sin(k\,x)} \mathrm{d}y$$
$$= \int_0^{3\,\pi/(4\,k)} \mathrm{d}x\,\{y\}_0^{\sin\,x} = \int_0^{3\,\pi/(4\,k)} A\sin(k\,x)\mathrm{d}x.$$

The MATLAB program for the area is given by:

```
syms x y A k real
% f(x) = y(x) = A*sin(k*x);
f = A*sin(k*x);
% 0 < x < xf
xf = (3*pi/4)/k;
% Area = int(dx dy) where
% 0<x<xf and 0<y<f
% Ay = int(dy) where 0<y<f
Ay = int(1,y,0,f);
% Ay = A*sin(k*x)
% Area = int(Ay dx) where 0<x<xf
Area = int(Ay,x,0,xf);
```

and the result is:

```
Area =
      1/2
  A (2    + 2)
  ------------
      2 k
```

The first moment of the area about the $y-$axis is

$$M_y = \int_A x\,\mathrm{d}A = \int_0^{3\pi/(4k)} \int_0^{A\,\sin(k\,x)} x\,\mathrm{d}x\,\mathrm{d}y = \int_0^{3\pi/(4k)} x\,\mathrm{d}x \int_0^{A\,\sin(k\,x)} \mathrm{d}y$$
$$= \int_0^{3\pi/(4k)} x\,\mathrm{d}x\,\{y\}_0^{A\,\sin(k\,x)} = \int_0^{3\pi/(4k)} A\,x\,\sin(k\,x)\mathrm{d}x,$$

With MATLAB the first moment of the area about the $y-$axis is:

```
Qyy = int(1,y,0,f);
My = int(x*Qyy,x,0,xf);
```

and the symbolic result is:

```
My =
   1/2
  2    A (3 pi + 4)
  -----------------
              2
           8 k
```

The x coordinate of the mass center is $x_C = M_y/Area$:

```
xC =
   1/2
  2    (3 pi + 4)
  ---------------
          1/2
  4 k (2    + 2)

xC = 1.854 (m)
```

The first moment of the area A about the $x-$axis is

$$M_x = \int_A y \, \mathrm{d}A = \int_0^{3\pi/(4k)} \int_0^{A \sin(kx)} y \, \mathrm{d}x \, \mathrm{d}y = \int_0^{3\pi/(4k)} \mathrm{d}x \int_0^{A \sin(kx)} y \, \mathrm{d}y$$

$$= \int_0^{3\pi/(4k)} \mathrm{d}x \left\{ \frac{y^2}{2} \right\}_0^{A \sin(kx)} = \int_0^{3\pi/(4k)} \frac{A^2 \sin^2(kx)}{2} \mathrm{d}x.$$

The first moment of the area about the $x-$axis in MATLAB is calculated with:

```
Qxy = int(y,y,0,f);
Mx = int(Qxy,x,0,xf);
```

and the symbolic result is:

```
Mx =
   2
  A  (3 pi + 2)
  -------------
      16 k
```

The y coordinate of the mass center is $y_C = M_x/Area$:

```
yC =
  A (3 pi + 2)
  ------------
      1/2
  8 (2    + 2)

yC = 0.627 (m)
```

The MATLAB statements for the graphical representation are:

```
A = 1.5;  % m
k = 0.75; % m^(-1)
sa = 4;
axis([0 sa  0 sa])
hold on, grid on
```

```
xx = 0:.1:xfn;
fx = A*sin(k*xx);
plot(xx,fx,'--','LineWidth',2)
hold on
area(xx,fx,'FaceColor','b',...
           'EdgeColor','k',...
           'LineWidth',2)
hold on
plot(xCn,yCn,...
'o','MarkerSize',12,...
'MarkerEdgeColor','k',...
'MarkerFaceColor','r')
xlabel('x (m)'), ylabel('y (m)')
text(xCn,yCn,'   C',...
'fontsize',14,'fontweight','b')
```

and the graphic is shown in Fig. 3.13.

Example 3.8 Find the coordinates of the centroid of the region bounded by the curves $y_1(x) = x/4$ and $y_2(x) = \sqrt{2\,(x-3)}$, $x_1 \leq x \leq x_2$, as shown in Fig. 3.14. All coordinates are in meters.

Solution The two curves will have two intersection points calculated in MATLAB with:

```
syms x y real
y1 = x/4;
y2 = sqrt(2*(x-3));
```

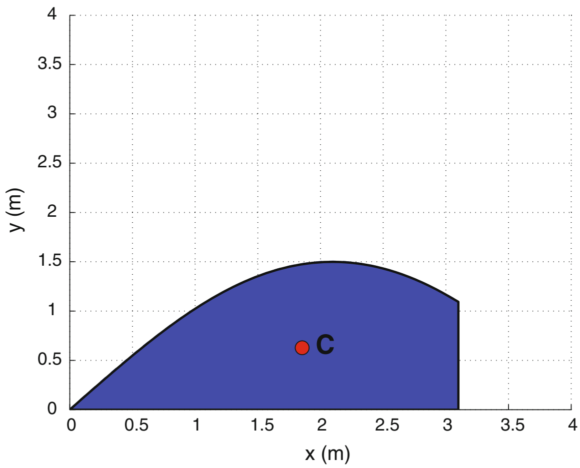


Fig. 3.13 Example 3.7

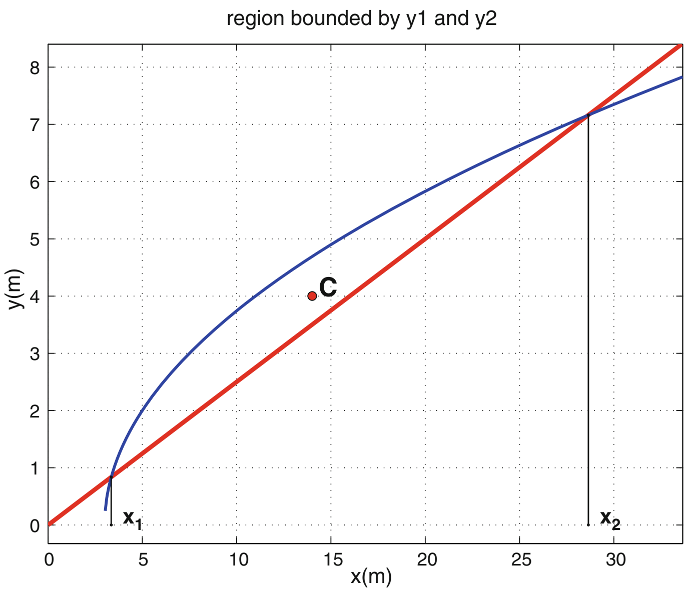


Fig. 3.14 Example 3.8

```
sol = eval(solve(y2-y1));
if sol(2) > sol(1)
x1 = sol(1); x2 = sol(2);
else
x1 = sol(2); x2 = sol(1);
end
y11 = subs(y1,x,x1);
y12 = subs(y1,x,x2);
```

The x values of the intersection points are:

```
x1 = 3.351 (m)
x2 = 28.649 (m)
```

The graphic shown in Fig. 3.14 is plotted with:

```
axis equal
g1=ezplot(y1,[0,x2+5])
set(g1, 'Color', 'r','LineWidth',3)
hold on
g2=ezplot(y2,[0,x2+5])
set(g2, 'Color', 'b','LineWidth',2)
hold on
line([x1 x1],[0 y11],...
    'Color','k','LineWidth',1,...
```

```
    'Marker','.','LineStyle','-')
hold on
line([x2 x2],[0 y12],...
    'Color','k','LineWidth',1,...
    'Marker','.','LineStyle','-')
hold on
grid on
title('region bounded by y1 and y2')
xlabel('x(m)'), ylabel('y(m)')
```

The area of the region is calculated with:

$$A = \int_A \mathrm{d}x\,\mathrm{d}y = \int_{x_1}^{x_2}\int_{y_1}^{y_2} \mathrm{d}x\,\mathrm{d}y = \int_{x_1}^{x_2} \mathrm{d}x \int_{x/4}^{\sqrt{2\,(x-3)}} \mathrm{d}y$$
$$= \int_{x_1}^{x_2} \left(\sqrt{2\,(x-3)} - x/4\right)\,\mathrm{d}x.$$

The command in MATLAB for calculating the area is:

```
A = double(int(int(1,y1,y2),x1,x2));
```

and the numerical result is:

```
A = 21.082 (m^2)
```

The first moment of the area about the $y-$axis is

$$M_y = \int_A x\,\mathrm{d}A = \int_{x_1}^{x_2}\int_{y_1}^{y_2} x\,\mathrm{d}x\,\mathrm{d}y = \int_{x_1}^{x_2} x\,\mathrm{d}x \int_{y_1}^{y_2} \mathrm{d}y$$
$$= \int_{x_1}^{x_2} x\,\mathrm{d}x\,\{y\}_{y_1}^{y_2} = \int_{x_1}^{x_2} x\,\left(\sqrt{2\,(x-3)} - x/4\right)\,\mathrm{d}x.$$

With MATLAB the first moment of the area about the $y-$axis and the x coordinate of the mass center $x_C = M_y/A$ are:

```
Qyy = int(1, y, y1, y2);
My = int(x*Qyy, x, x1, x2);
xC = eval(My/A);
```

and the result is:

```
xC = 14.000 (m)
```

The first moment of the area A about the $x-$axis is

$$M_x = \int_A y\,dA = \int_{x_1}^{x_2}\int_{y_1}^{y_2} y\,dx\,dy = \int_{x_1}^{x_2} dx \int_{y_1}^{y_2} y\,dy$$

$$= \int_{x_1}^{x_2} dx \left\{\frac{y^2}{2}\right\}_{y_1}^{y_2} = \int_{x_1}^{x_2} \frac{1}{2}\left[2\,(x-3) - \frac{x^2}{16}\right] dx.$$

The first moment M_x and y_C are calculated in MATLAB with:

```
Qxy = int(y, y, y1, y2);
Mx = int(Qxy, x, x1, x2);
yC = eval(Mx/A);
```

and the result is:

```
yC = 4.000 (m)
```

Example 3.9 Find the position of the center of mass the region defined by *OABDEF* as shown in Fig. 3.15, where $EF = DB = a = 4$ m and $AB = DE = b = 2$ m. The material is homogeneous.

Solution

The region is bounded by the lines of equations $y_1(x) = 2\,b$ for $0 \le x \le a$, $y_2(x) = b$ for $a < x \le 2\,a$ and the $x-$axis. The area of the region is given by

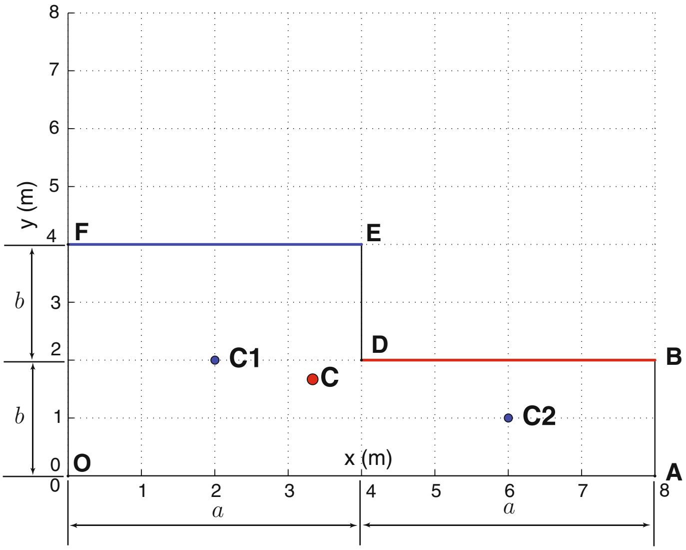


Fig. 3.15 Example 3.9

$$A = A_1 + A_2 = \int_0^a y_1 \, \mathrm{d}x + \int_a^{2a} y_2 \, \mathrm{d}x = \int_0^a (2\,b) \, \mathrm{d}x + \int_a^{2a} b \, \mathrm{d}x$$
$$= 2\,b\,a + a\,b = 3\,a\,b = 24 \text{ m}^2.$$

The first moment of the area about the $y-$axis for the composite region is

$$M_y = \int_A x \, \mathrm{d}A = \int_0^a x\,y_1 \, \mathrm{d}x + \int_a^{2\,a} x\,y_2 \, \mathrm{d}x = \int_0^a 2\,bx \, \mathrm{d}x + \int_a^{2\,a} bx \, \mathrm{d}x.$$

With MATLAB the first moment of the area about the $y-$axis and the x coordinate of the mass center $x_C = M_y/A$ are:

```
syms x a b  real
sl = {a,b};
sn = {4,2};
y1 = 2*b;
y2 = b;
A1 = int(y1,x,0,a);
A2 = int(y2,x,a,2*a);
A = A1+A2;
Mx1 = int(x*y1,x,0,a);
Mx2 = int(x*y2,x,a,2*a);
xC1 = Mx1/A1;
xC2 = Mx2/A2;
xC = (Mx1+Mx2)/A;
```

The results are:

```
My =
       2
  5 a b
  ------
    2

xC =
  5 a
  ---
   6

xC = 3.333 (m)
```

The first moment of the area about the $x-$axis is calculated with the general formula

$$M_x = 0.5 \int\limits_{x_1}^{x_2} y^2(x)\,\mathrm{d}x,$$

and for $A = A_1 + A_2$ it results

$$M_x = M_{x_1} + M_{x_2} = 0.5 \int\limits_0^a {y_1}^2\,\mathrm{d}x + 0.5 \int\limits_a^{2a} {y_2}^2\,\mathrm{d}x$$

$$= 0.5 \int\limits_0^a (2\,b)^2\,\mathrm{d}x + 0.5 \int\limits_a^{2a} b^2\,\mathrm{d}x.$$

The first moment M_x and y_C are calculated in MATLAB with:

```
My1 = 0.5*int(y1^2,x,0,a);
My2 = 0.5*int(y2^2,x,a,2*a);
My = My1 + My2;
yC1 = My1/A1;
yC2 = My2/A2;
yC = (My1+My2)/A;
```

and the result are:

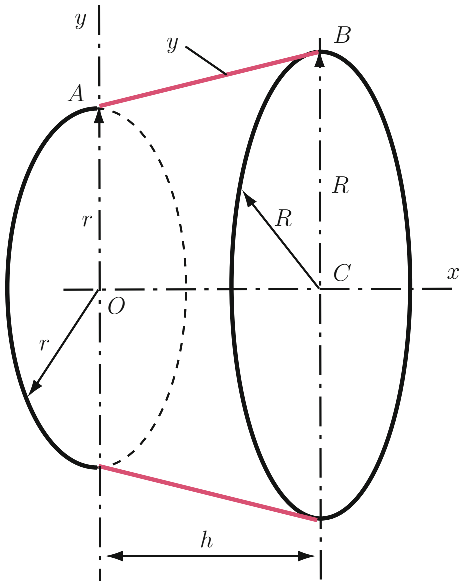
```
Mx =
       2
  5 a  b
  ------
    2

yC =
  5 b
  ---
   6

yC = 1.667 (m)
```

Example 3.10 Find the volume of the frustum of a cone, shown in Fig. 3.16, where $h = 2$ m is the height, $R = 2$ m is the radius of large base, and $r = 1$ m radius of small base. The material is homogeneous.

Solution The formula for calculating the volume is:

$$V = \pi \int\limits_a^b f^2(x)\mathrm{d}x$$

Fig. 3.16 Example 3.10



where $y = f(x)$ is the generating equation of the planar curve. For the frustum of a cone the generating equation is

$$y = f(x) = f = \frac{(R-r)\,x}{h} + r,$$

and the MATLAB program is;

```
 syms R r h x  real
f = (R-r)*x/h+r;
V = pi*int(f^2,x,0,h);
ls = {R,r,h};
ln = {2,1,2};
fn = subs(f,ls,ln);
Vn = subs(V,ls,ln);
```

The results are:

```
V =
          2           2
   pi h (R  + R r + r )
   --------------------
            3
 V = 14.661 (m^3)
```

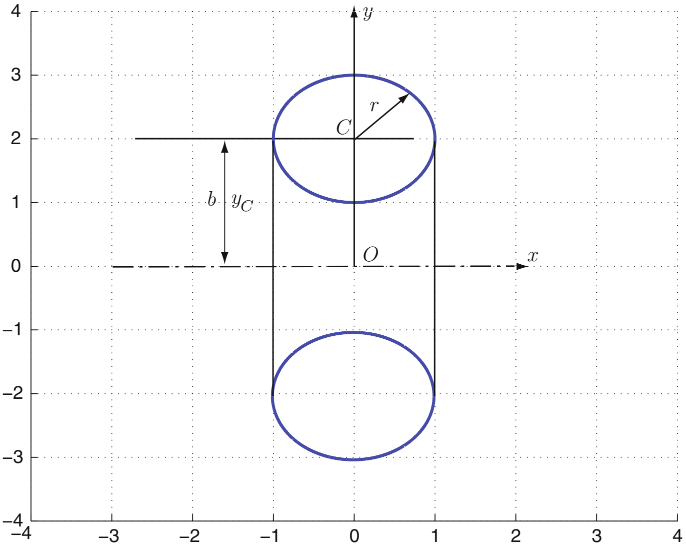


Fig. 3.17 Example 3.11

Example 3.11 Find the volume and surface area of the complete torus of circular cross section of radius $r = 1$ m as shown in Fig. 3.17 where $b = 2$ m.

Solution The torus is generated by revolving the circular area of radius r through 360° about the $x-$axis. With the first theorem of Guldinus-Pappus, the surface of revolution is $S = 2\pi y_C L$, where $L = 2\pi r$ is the length of generating circle and $y_C = b$ is the centroid of generating circle

$$S = 2\pi b\, 2\pi r = 4\pi^2 b\, r = 78.957 \text{ m}^2.$$

The second theorem of Guldinus-Pappus gives the volume of revolution $V = 2\pi y_C A$, where $A = \pi r^2$ is the area of generating circular surface and $y_C = b$ is the centroid of generating circular surface

$$V = 2\pi b\, \pi r^2 = 2\pi^2 b\, r^2 = 39.478 \text{ m}^3.$$

The equation of the generating circle is

$$x^2 + (y - b)^2 - r^2 = 0.$$

The volume of the torus can also be calculated with the formula

$$V = \pi \int_a^b f^2(x)\mathrm{d}x = \pi \int_{-r}^{r} (f_2^2 - f_1^2)\mathrm{d}x,$$

where

$$f_1 = b - \sqrt{r^2 - x^2} \text{ and } f_2 = b + \sqrt{r^2 - x^2},$$

and the MATLAB program is:

```
syms b r x  real
f1 = b-sqrt(r^2-x^2);
f2 = b+sqrt(r^2-x^2);
V = pi*int(f2^2-f1^2,x,-r,r);
```

3.8 Problems

3.1 Locate the centroid of the uniform wire bent in the shape shown in Fig. 3.18. For the numerical application use $r = 1$ m, $a = 2$ m, and $b = 1.75$ m.

3.2 Find the location of the centroid C of the uniform area shown in Fig. 3.19 where $a = 0.4$ m, $b = 0.8$ m, and $c = 0.6$ m.

3.3 Find the location of the centroid of the area shown in Fig. 3.20. For the numerical application use $r = 0.1$ m and $h = 0.2$ m.

3.4 Determine the location of the centroid of the uniform area shown in Fig. 3.21. For the numerical application use $a = 0.2$ m, $b = 0.25$ m, and $c = 0.27$ m.

3.5 Find the location of the centroid of the uniform area shown in Fig. 3.22. For the numerical application use $a = 0.6$ m, $b = 0.4$ m, and $c = 0.3$ m.

3.6 Locate the centroid of the volume shown in Fig. 3.23, where $r = 0.5$ m, and $h = 1.2$ m. The material is homogeneous.

3.7 Locate the centroid of the volume shown in Fig. 3.24, where $r = 0.3$ m, and $h = 0.9$ m. The material is homogeneous.

3.8 Locate the centroid of the homogeneous volume shown in Fig. 3.25, where $R = 0.6$ m, $r = 0.4$ m, $a = 0.5$ m, and $b = 0.6$ m. The material is homogeneous.

3.9 Locate the centroid of the volume shown in Fig. 3.26, where $R = 0.7$ m, $r = 0.4$ m, $p = 0.5$ m, $a = 0.4$ m, and $b = 0.5$ m. The material is homogeneous.

3.10 Find the centroid of the volume depicted in Fig. 3.27, where $r = 25$ mm, $a = 200$ m, $b = 100$ mm, and $t = 15$ mm. The material is homogeneous.

3.11 Find the centroid of the volume shown in Fig. 3.28, where $a = 200$ mm, $b = 150$ mm, and $t = 30$ mm. The material is homogeneous.

3.12 Find the centroid of the volume shown in Fig. 3.29, where $a = 400$ mm, $b = 200$ mm, and $c = 100$ mm. The material is homogeneous.

3.13 Find the centroid of the volume shown in Fig. 3.30, where $a = 100$ mm, $b = 125$ mm, $c = 150$ mm, and $t = 25$ mm. The material is homogeneous.

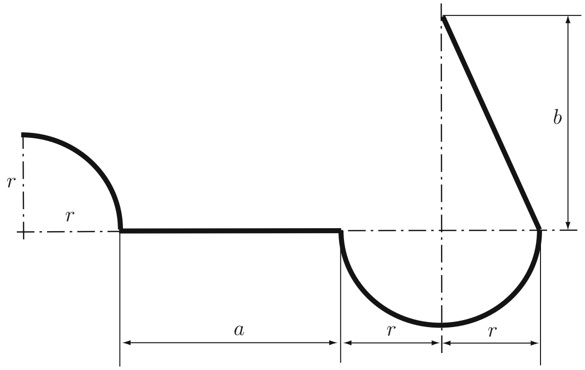


Fig. 3.18 Problem 3.1

Fig. 3.19 Problem 3.2

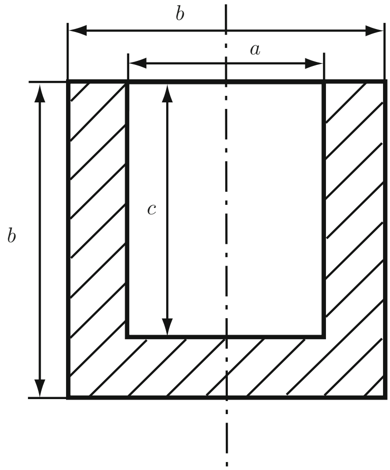
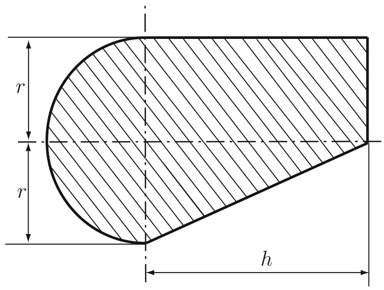


Fig. 3.20 Problem 3.3



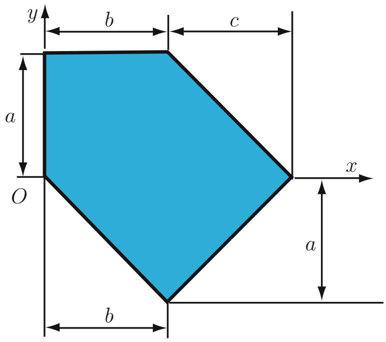


Fig. 3.21 Problem 3.4

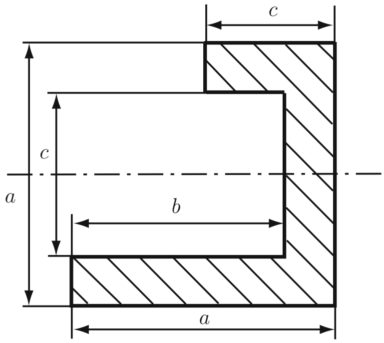


Fig. 3.22 Problem 3.5

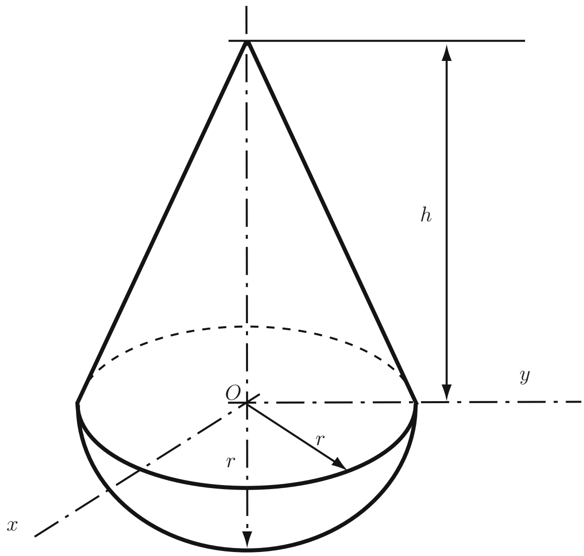


Fig. 3.23 Problem 3.6

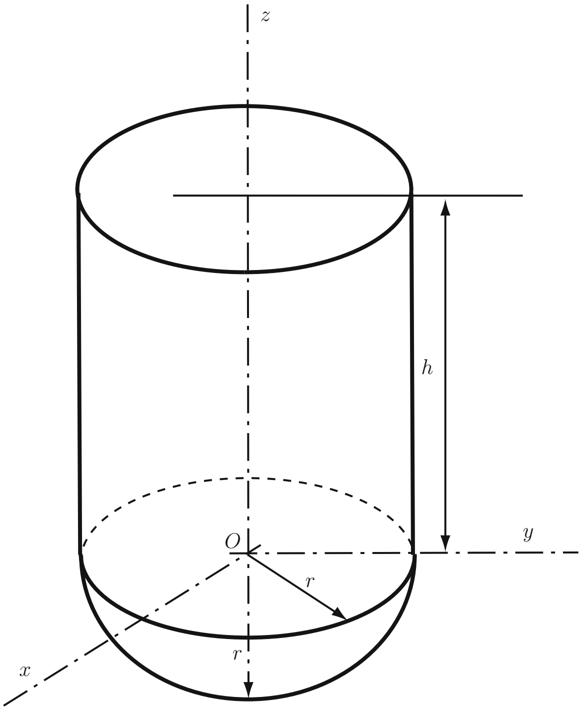


Fig. 3.24 Problem 3.7

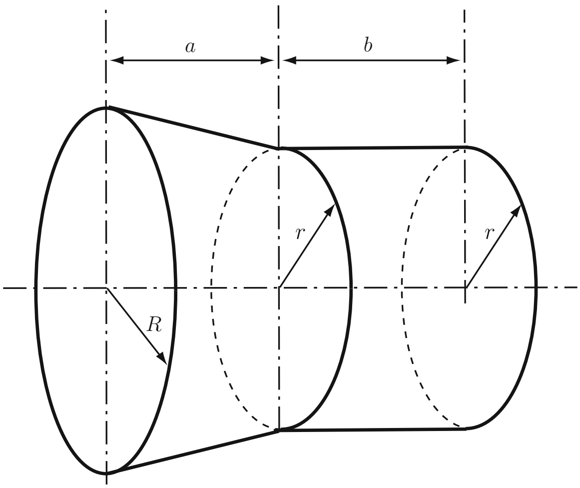


Fig. 3.25 Problem 3.8

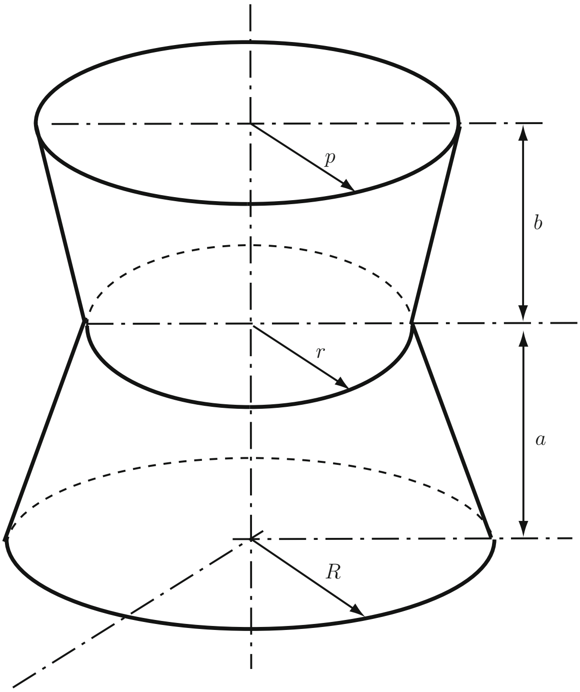


Fig. 3.26 Problem 3.9

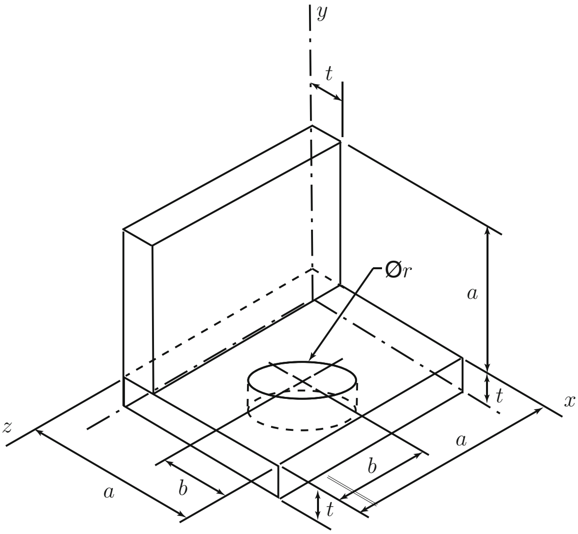


Fig. 3.27 Problem 3.10

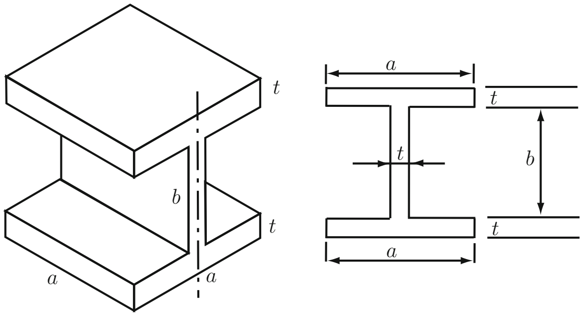


Fig. 3.28 Problem 3.11

Fig. 3.29 Problem 3.12

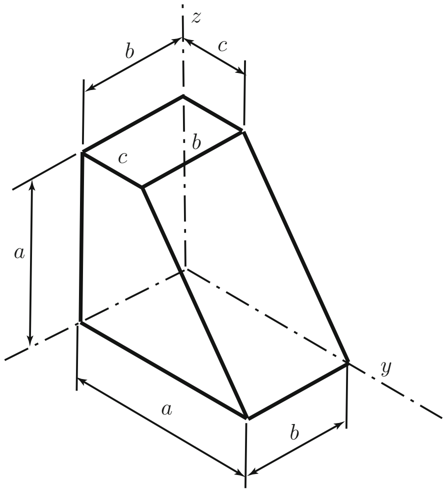
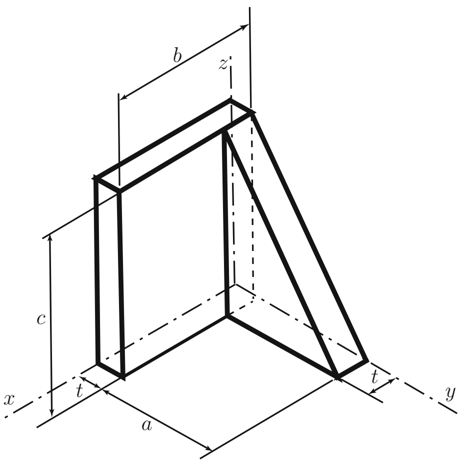


Fig. 3.30 Problem 3.13



3.14 Find the coordinates of the centroid of the region is bounded by the curves $y = x$ and $y = \sqrt{x}$ where $0 \le x \le 1$. All coordinates may be treated as dimensionless.

3.15 Determine the coordinates of the centroid of the region is bounded by the curves $y = x^2$ and $y = \sqrt{x}$ where $0 \le x \le 1$. All coordinates may be treated as dimensionless.

3.9 Programs

3.9.1 Program 3.1

```
% example 3.1
% center of mass of a curve
clear all; clc; close all;
syms x real

a = 3;
b = 2;
y  = b*sqrt(x^a);
dy = diff(y,x);
dl =sqrt(1+dy^2);
% ds = (1+ (dy/dx)^2)^0.5
%  0< x < 1
L = eval(int(dl,0,1));
My=eval(int(x*dl,0,1));
xC=My/L;
Mx=eval(int(y*dl,0,1));
yC=Mx/L;
fprintf('L = %4.3f (m) \n', L)
fprintf('xC = %4.3f (m) \n', xC)
fprintf('yC = %4.3f (m) \n', yC)

% plot the curve and CM
xf=1;
xn = 0:xf/100:xf;
yn = b*sqrt(xn.^a);
axis ([0 1 0 1])
plot(xn,yn,'-b','LineWidth',2)
hold on
plot(xC,yC,'o','MarkerSize',12,...
               'MarkerEdgeColor','k',...
               'MarkerFaceColor','r')
```

```
text(xC,yC,'       C','FontSize',18)
title('y=f(x)=2 x^{3/2}')
grid on
xlabel('x(m)')
ylabel('y(m)')

% end of program
```

3.9.2 Program 3.2

```
% example 3.2
% center of mass
% circle (Cartesian equation)
% x^2+y^2=1
% semi-circle
% quarter-circle
clear all; clc; close all;
syms r t real

% parametric equation
x = r*cos(t);
y = r*sin(t);
% 0 < t < 2*pi
% r > 0
dx = diff(x,t);
dy = diff(y,t);
% arc length
% dl = ((dx/dt)^2+(dy/dt)^2)^0.5 dt

dl = (dx^2+dy^2)^0.5;
dl = simplify(dl);
fprintf('dl = %s dt \n', char(dl));
L = int(dl,t,0,2*pi);
fprintf('L = %s \n', char(L))
fprintf('\n');

% semi-circle
Ls  = int(dl,t,0,pi);
Mys = int(x*dl,t,0,pi);
xCs = simplify(Mys/Ls);
Mxs = int(y*dl,t,0,pi);
yCs = simplify(Mxs/Ls);
fprintf('Ls = %s \n', char(Ls))
```

```
fprintf('xCs = %s \n', char(xCs))
fprintf('yCs = %s \n', char(yCs))
fprintf('\n');

% quarter-circle
Lq  = int(dl,t,0,pi/2);
Myq = int(x*dl,t,0,pi/2);
xCq = simplify(Myq/Lq);
Mxq = int(y*dl,t,0,pi/2);
yCq = simplify(Mxq/Lq);
fprintf('Lq = %s \n', char(Lq))
fprintf('xCq = %s \n', char(xCq))
fprintf('yCq = %s \n', char(yCq))

rn=1;
% plot the semi-circle and CM
figure(1)
xCsn = subs(xCs,r,1);
yCsn = subs(yCs,r,1);

tn = 0:pi/18:pi;
xn = rn*cos(tn);
yn = rn*sin(tn);
axis manual
axis equal
hold on
grid on
sa = 1;
axis ([-sa sa -sa sa])
plot(xn,yn,'-b','LineWidth',2)
text(0,0,' O','fontsize',14)
line([-sa,sa],[0,0],'Color','k')
line([0,0],[0,sa],'Color','k')
plot(xCsn,yCsn,'o','MarkerSize',12,...
               'MarkerEdgeColor','k',...
               'MarkerFaceColor','r')
text(xCsn,yCsn,'   C','FontSize',18)
title('semi-circle')
xlabel('x(m)')
ylabel('y(m)')

% plot the quarter-circle and CM

figure(2)
xCqn = subs(xCq,r,1);
```

```
yCqn = subs(yCq,r,1);

tn = 0:pi/18:pi/2;
xn = rn*cos(tn);
yn = rn*sin(tn);
axis manual
axis equal
hold on
grid on
sa = 1;
axis ([-sa sa -sa sa])
plot(xn,yn,'-b','LineWidth',2)
text(0,0,' O','fontsize',14)
line([0,sa],[0,0],'Color','k')
line([0,0],[0,sa],'Color','k')
plot(xCqn,yCqn,'o','MarkerSize',12,...
               'MarkerEdgeColor','k',...
               'MarkerFaceColor','r')
text(xCqn,yCqn,'  C','FontSize',18)
title('quarter-circle')
xlabel('x(m)')
ylabel('y(m)')

% end of program
```

3.9.3 Program 3.3

```
% example 3.3
% quarter-astroid (one cusp)
% astroid=hypocycloid with 4 cusps
% x^(2/3)+y^(2/3)=a^(2/3)
clear all; clc; close all;
syms t real
a = 1; % (m)
x = a*cos(t)^3;
y = a*sin(t)^3;
% a > 0
% 0 < t < pi/2 - quarter-astroid
dx = diff(x,t);
dy = diff(y,t);
% dl=((dx/dt)^2+(dy/dt)^2)^0.5 dt
dl = (dx^2+dy^2)^0.5;
dl = simplify(dl);
```

```
L = int(dl,t,0,pi/2);
L = double(L);
% L = int(dl,0,pi/2);
My = int(x*dl,t,0,pi/2);
xC = My/L;
xC = double(xC);
Mx = int(y*dl,t,0,pi/2);
yC = Mx/L;
yC = double(yC);
fprintf('L = %4.3f (m)\n', L)
fprintf('xC = %4.3f (m)\n', xC)
fprintf('yC = %4.3f (m)\n', yC)
fprintf('\n');

% plot and CM
tn = 0:pi/18:pi/2;
xn = a*cos(tn).^3;
yn = a*sin(tn).^3;
axis manual
axis equal
hold on
grid on
sa = 1;
axis ([0 sa 0 sa])
plot(xn,yn,'-b','LineWidth',2)
text(0,0,' O','fontsize',14)
line([0,sa],[0,0],'Color','k')
line([0,0],[0,sa],'Color','k')
plot(xC,yC,'o','MarkerSize',12,...
               'MarkerEdgeColor','k',...
               'MarkerFaceColor','r')
text(xC,yC,'   C','FontSize',18)
title('quarter-astroid')
xlabel('x(m)')
ylabel('y(m)')

% end of program
```

3.9.4 Program 3.4

```
% example 3.4
% three-dimensional helix
clear all; clc; close all
```

```
syms a h t real
% circular helix
% radius a and pitch 2*pi*h
% cartesian coordinates
x = a*cos(t);
y = a*sin(t);
z = h*t;

dx = diff(x,t);
dy = diff(y,t);
dz = diff(z,t);
%dl=((dx/dt)^2+(dy/dt)^2+(dz/dt)^2)^0.5 dt
dl = (dx^2+dy^2+dz^2)^0.5;
dl = simplify(dl);
fprintf('dl = %s dt \n', char(dl));
fprintf('\n');
% dl =(a^2 + h^2)^(1/2)

% numerical data a=h=1 (m)
x = cos(t);
y = sin(t);
z = t;
dl=(diff(x)^2+diff(y)^2+diff(z)^2)^0.5;

tf = 6*pi;
L = int(dl,t,0,tf);
L = double(L);
xC = int(x*dl,t,0,tf)/L;
xC = double(xC);
yC = int(y*dl,t,0,tf)/L;
yC = double(yC);
zC = int(z*dl,t,0,tf)/L;
zC = double(zC);
fprintf('L = %4.3f (m)\n', L)
fprintf('xC = %4.3f (m)\n', xC)
fprintf('yC = %4.3f (m)\n', yC)
fprintf('zC = %4.3f (m)\n', zC)
fprintf('\n');

% plot helix
tn = 0:pi/50:tf;
plot3(sin(tn),cos(tn),tn)
hold on
plot3(xC,yC,zC,...
'o','MarkerSize',12,...
```

```
'MarkerEdgeColor','k',...
'MarkerFaceColor','r')
text(xC,yC,zC,'   C','FontSize',18)
xlabel('x(m)')
ylabel('y(m)')
zlabel('z(m)')
grid on
axis square

% end of program
```

3.9.5 Program 3.5

```
% example 3.5
clear all; clc; close all;
syms t real
a=2; k=1;
x = a*exp(k*t)*cos(t);
y = a*exp(k*t)*sin(t);
z = a*exp(k*t);

dx = diff(x,t);
dy = diff(y,t);
dz = diff(z,t);
%dl=((dx/dt)^2+(dy/dt)^2+(dz/dt)^2)^0.5 dt
dl = (dx^2+dy^2+dz^2)^0.5;
dl = simplify(dl);
fprintf('dl = %s dt\n',char(dl))

tf = 3;
L = int(dl,0,tf);
L = double(L);

xC = int(x*dl,t,0,tf)/L;
xC = double(xC);
yC = int(y*dl,t,0,tf)/L;
yC = double(yC);
zC = int(z*dl,t,0,tf)/L;
zC = double(zC);
fprintf('L = %4.3f (m)\n', L)
fprintf('xC = %4.3f (m)\n', xC)
fprintf('yC = %4.3f (m)\n', yC)
fprintf('zC = %4.3f (m)\n', zC)
```

```
fprintf('\n');

% plot the curve
tn = 0:pi/100:tf;
xn = a*exp(k*tn).*cos(tn);
yn = a*exp(k*tn).*sin(tn);
zn = a*exp(k*tn);
ht = plot3(xn,yn,zn);
hold on
plot3(xC,yC,zC,...
'o','MarkerSize',12,...
'MarkerEdgeColor','k',...
'MarkerFaceColor','r')
text(xC,yC,zC,'   C','FontSize',18)
xlabel('x(m)')
ylabel('y(m)')
zlabel('z(m)')
grid on
axis square

% end of program
```

3.9.6 Program 3.6

```
% example 3.6
clear all; clc; close all
syms x y a b real

% f(x) = y(x) = b*x/a;
f = b*x/a;
% 0 < x < xf
xf = a;

% Area = int(dx dy) where
% 0<x<xf and 0<y<f
% Ay = int(dy) where 0<y<f
Ay = int(1,y,0,f);
% Area = int(Ay dx) where 0<x<xf
Area = int(Ay,x,0,xf);

% first moment of area about y-axis
% My = int(x dx dy) where
% 0<x<xf and 0<y<f
```

```
% Qyy = int(dy) ; 0<y<f
Qyy = int(1,y,0,f);
% My = int(x Qyy dx) where 0<x<xf
My = int(x*Qyy,x,0,xf);
% centroid xC = My/Area
xC = My/Area;

% first moment of area about x-axis
% Mx = int(y dx dy) where
% 0<x<xf and 0<y<f
% Qxy = int(y dy) ; 0<y<f
Qxy = int(y,y,0,f);
% Mx = int(x Qxy dx) where 0<x<xf
Mx = int(Qxy,x,0,xf);
yC = Mx/Area;

ls = {a,b};
ln = {1,1}; % (m)

xfn = subs(xf,ls,ln);
xCn = subs(xC,ls,ln);
yCn = subs(yC,ls,ln);

fprintf('xf = %s = %4.3f (m)\n',char(xf),double(xfn))
fprintf('Area = %s (m^2)\n',char(Area))
fprintf('My = %s (m^3)\n',char(My))
fprintf('xC = %s = %4.3f (m)\n',char(xC),double(xCn))
fprintf('Mx = %s (m^3)\n',char(Mx))
fprintf('yC = %s = %4.3f (m)\n',char(yC),double(yCn))

sa = 1.5;
axis([0 sa 0 sa])
hold on, grid on
xx = 0:.1:xfn;
fx = subs(f,{a,b,x},{1,1,xx});
plot(xx,fx,'--','LineWidth',2)
hold on
area(xx,fx,'FaceColor','b',...
           'EdgeColor','k',...
           'LineWidth',2)
hold on
plot(xCn,yCn,...
'o','MarkerSize',12,...
'MarkerEdgeColor','k',...
'MarkerFaceColor','r')
```

```
xlabel('x (m)'), ylabel('y (m)')
text(xCn,yCn,'   C','fontsize',14,'fontweight','b')

% end of program
```

3.9.7 Program 3.7

```
% example 3.7
clear all; clc; close all
syms x y A k real

% f(x) = y(x) = A*sin(k*x);
f = A*sin(k*x);
% 0 < x < xf
xf = (3*pi/4)/k;

% Area = int(dx dy) where
% 0<x<xf and 0<y<f
% Ay = int(dy) where 0<y<f
Ay = int(1,y,0,f);
% Ay = A*sin(k*x)
% Area = int(Ay dx) where 0<x<xf
Area = int(Ay,x,0,xf);

% first moment of area about y-axis
% My = int(x dx dy) where
% 0<x<xf and 0<y<f
% Qyy = int(dy) ; 0<y<f
Qyy = int(1,y,0,f);
% My = int(x Qyy dx) where 0<x<xf
My = int(x*Qyy,x,0,xf);
xC = My/Area;

% first moment of area about x-axis
% Mx = int(y dx dy) where
% 0<x<xf and 0<y<f
% Qxy = int(y dy) ; 0<y<f
Qxy = int(y,y,0,f);
% Mx = int(x Qxy dx) where 0<x<xf
Mx = int(Qxy,x,0,xf);
yC = Mx/Area;
```

```
% A = 1.5;  % m
% k = 0.75; % m^(-1)
xfn = subs(xf,{A,k},{1.5,0.75});
xCn = subs(xC,{A,k},{1.5,0.75});
yCn = subs(yC,{A,k},{1.5,0.75});
Arean = subs(Area,{A,k},{1.5,0.75});

fprintf('xf = %s = %4.3f (m)\n',char(xf),double(xfn))
fprintf('\n')
fprintf('Area = ')
pretty(Area)
fprintf('\n')
fprintf('My = ')
pretty(My)
fprintf('\n')
fprintf('xC = ')
pretty(xC)
fprintf('\n')
fprintf('xC = %4.3f (m)\n',double(xCn))
fprintf('\n')
fprintf('Mx = ')
pretty(Mx)
fprintf('\n')
fprintf('yC = ')
pretty(yC)
fprintf('\n')
fprintf('yC = %4.3f (m)\n',double(yCn))

A = 1.5;  % m
k = 0.75; % m^(-1)
sa = 4;
axis([0 sa  0 sa])
hold on, grid on
xx = 0:.1:xfn;
fx = A*sin(k*xx);
plot(xx,fx,'--','LineWidth',2)
hold on
area(xx,fx,'FaceColor','b',...
           'EdgeColor','k',...
           'LineWidth',2)
hold on
plot(xCn,yCn,...
'o','MarkerSize',12,...
'MarkerEdgeColor','k',...
'MarkerFaceColor','r')
```

```
xlabel('x (m)'), ylabel('y (m)')
text(xCn,yCn,'   C',...
'fontsize',14,'fontweight','b')

% end of program
```

3.9.8 Program 3.8

```
% example 3.8
clear all; clc; close all
syms x y real
% y1 = x/4
% y2 = sqrt(2*(x-3))

y1 = x/4;
y2 = sqrt(2*(x-3));

sol = eval(solve(y2-y1));
if sol(2) > sol(1)
x1 = sol(1); x2 = sol(2);
else
x1 = sol(2); x2 = sol(1);
end

y11 = subs(y1,x,x1);
y12 = subs(y1,x,x2);

axis equal
g1=ezplot(y1,[0,x2+5])
set(g1, 'Color', 'r','LineWidth',3)
hold on
g2=ezplot(y2,[0,x2+5])
set(g2, 'Color', 'b','LineWidth',2)
hold on
line([x1 x1],[0 y11],...
    'Color','k','LineWidth',1,...
    'Marker','.','LineStyle','-')
hold on
line([x2 x2],[0 y12],...
    'Color','k','LineWidth',1,...
    'Marker','.','LineStyle','-')
hold on
grid on
```

```
title('region bounded by y1 and y2')
xlabel('x(m)'), ylabel('y(m)')

% Area = eval(int(abs(y2-y1),x,x1,x2))
A = double(int(int(1,y1,y2),x1,x2));

Qyy = int(1, y, y1, y2);
My = int(x*Qyy, x, x1, x2);
xC = eval(My/A);

Qxy = int(y, y, y1, y2);
Mx = int(Qxy, x, x1, x2);
yC = eval(Mx/A);

plot(xC,yC,...
'o','MarkerSize',6,...
'MarkerEdgeColor','k',...
'MarkerFaceColor','r')
text(xC,yC,'   C',...
'fontsize',14,'fontweight','b')

text(x1,0,'  x_1',...
'fontsize',14,'fontweight','b')
text(x2,0,'  x_2',...
'fontsize',14,'fontweight','b')

fprintf('x1 = %4.3f (m)\n',double(x1))
fprintf('x2 = %4.3f (m)\n',double(x2))
fprintf('A = %4.3f (m^2)\n',double(A))
fprintf('xC = %4.3f (m)\n',double(xC))
fprintf('yC = %4.3f (m)\n',double(yC))

% end of program
```

3.9.9 Program 3.9

```
% example 3.9
clear all; clc; close all
% f = f(x)
% A = int(f,x,a,b)
% xC = int(x*f,x,a,b)/A
% yC = 0.5*int(f^2,x,a,b)/A

syms x a b  real
```

```
sl = {a,b};
sn = {4,2};

y1 = 2*b;
y2 = b;

A1 = int(y1,x,0,a);
A2 = int(y2,x,a,2*a);
A = A1+A2;

Mx1 = int(x*y1,x,0,a);
Mx2 = int(x*y2,x,a,2*a);
Mx = Mx1 + Mx2;

xC1 = Mx1/A1;
xC2 = Mx2/A2;

xC = (Mx1+Mx2)/A;

My1 = 0.5*int(y1^2,x,0,a);
My2 = 0.5*int(y2^2,x,a,2*a);
My = My1 + My2;

yC1 = My1/A1;
yC2 = My2/A2;

yC = (My1+My2)/A;

An = subs(A,sl,sn);
xCn = subs(xC,sl,sn);
yCn = subs(yC,sl,sn);

xC1n = subs(xC1,sl,sn);
yC1n = subs(yC1,sl,sn);

xC2n = subs(xC2,sl,sn);
yC2n = subs(yC2,sl,sn);

fprintf('A = ')
pretty(A)
fprintf('\n')
fprintf('A = %4.3f (m^2)\n',double(An))
fprintf('\n')
fprintf('My = ')
pretty(My)
```

```
fprintf('\n')
fprintf('xC = ')
pretty(xC)
fprintf('\n')
fprintf('xC = %4.3f (m)\n',double(xCn))
fprintf('\n')
fprintf('Mx = ')
pretty(Mx)
fprintf('\n')
fprintf('yC = ')
pretty(yC)
fprintf('\n')
fprintf('yC = %4.3f (m)\n',double(yCn))

an = subs(a,sl,sn);
bn = subs(b,sl,sn);
xA = 2*an; yA = 0;
xB = xA;   yB = bn;
xD = an;   yD = yB;
xE = xD;   yE = 2*bn;
xF = 0;    yF = yE;

% axis square

sa = 8;
axis([0 sa  0 sa])
hold on, grid on

line([0 xA],[0 yA],...
    'Color','k','LineWidth',1,...
    'Marker','.','LineStyle','-')
hold on
line([xA xB],[yA yB],...
    'Color','k','LineWidth',1,...
    'Marker','.','LineStyle','-')
hold on
line([xB xD],[yB yD],...
    'Color','r','LineWidth',2,...
    'Marker','.','LineStyle','-')
hold on
line([xD xE],[yD yE],...
    'Color','k','LineWidth',1,...
    'Marker','.','LineStyle','-')
hold on
line([xE xF],[yE yF],...
```

```
    'Color','b','LineWidth',2,...
    'Marker','.','LineStyle','-')
hold on
line([0 xF],[0 yF],...
    'Color','k','LineWidth',1,...
    'Marker','.','LineStyle','-')

plot(xCn,yCn,...
'o','MarkerSize',8,...
'MarkerEdgeColor','k',...
'MarkerFaceColor','r')
text(xCn,yCn,'   C',...
'fontsize',14,'fontweight','b')

plot(xC1n,yC1n,...
'o','MarkerSize',6,...
'MarkerEdgeColor','k',...
'MarkerFaceColor','b')
text(xC1n,yC1n,'   C1',...
'fontsize',14,'fontweight','b')

plot(xC2n,yC2n,...
'o','MarkerSize',6,...
'MarkerEdgeColor','k',...
'MarkerFaceColor','b')
text(xC2n,yC2n,'   C2',...
'fontsize',14,'fontweight','b')

xlabel('x(m)'), ylabel('y(m)')
text(xA,yA,'   A',...
'fontsize',14,'fontweight','b')
text(xB,yB,'   B',...
'fontsize',14,'fontweight','b')
text(xD,yD,'   D',...
'fontsize',14,'fontweight','b')
text(xE,yE,'   E',...
'fontsize',14,'fontweight','b')
text(xF,yF,'   F',...
'fontsize',14,'fontweight','b')
text(0,0,'   O',...
'fontsize',14,'fontweight','b')

% end of program
```

3.9.10 Program 3.10

```
% example 3.10

% frustum of a right-circular cone
% volume of a frustum of a cone
% h height
% R radius of large base
% r radius of small base

clear all; clc; close all
% f = y(x)
% V = pi*int(f^2,x,a,b)

syms R r h x  real

f = (R-r)*x/h+r;

% volume of thin disk differential element
% dV = pi f^2 dx

V = pi*int(f^2,x,0,h);

% centroid
% xC = int(x*pi*f^2,x,0,h)/V;

xC = int(x*pi*f^2,x,0,h)/V;
xC = simplify(xC);

fprintf('V = ')
fprintf('\n')
pretty(V)
fprintf('\n')
fprintf('xC = ')
fprintf('\n')
pretty(xC)
fprintf('\n')

ls = {R,r,h};
ln = {2,1,2};
fn = subs(f,ls,ln);
Vn = subs(V,ls,ln);
xCn= subs(xC,ls,ln);
```

```
fprintf('V = %4.3f (m^3)\n',double(Vn))
fprintf('xC = %4.3f (m^3)\n',double(xCn))

g1 = ezplot(fn,[0,2]);
set(g1,'Color','b','LineWidth',2)
grid on

% end of program
```

3.9.11 Program 3.11

```
% example 3.11
% torus volume
% x^2+(y-b)^2-r^2=, b>0
% torus can be generated by revolving
% the circular area of radius r
% through 360 deg. about the x-axis
clear all; clc; close all
% f = f(x)
% V = pi*int(f^2,x,x1,x2)

syms b r x  real

f1 = b-sqrt(r^2-x^2);
f2 = b+sqrt(r^2-x^2);
V = pi*int(f2^2-f1^2,x,-r,r);
fprintf('V = %s \n',char(V))

% theorems of Guldinus-Pappus
%
% S = 2 pi yC L  surface of revolution
% L length of generating curve
% yC centroid of generating curve
%
% V = 2 pi yC A  volume of revolution
% A area of generating plane surface
% yC centroid of generating plane surface

yC = b;
A = pi*r^2;
Vg = 2*pi*yC*A;
L = 2*pi*r;
S = 2*pi*yC*L;
```

```
fprintf('\n')
fprintf('Vg = %s \n',char(Vg))
fprintf('\n')
fprintf('S = %s \n',char(S))
fprintf('\n')

% end of program
```

References

1. P. Appell, *Traité de mécanique rationnelle* (Gauthier-Villars, Paris, 1955)
2. M. Atanasiu, *Mechanics* (EDP, Bucharest, 1973)
3. H. Baruh, *Analytical Dynamics* (WCB/McGraw-Hill, Boston, 1999)
4. G. Baumann, *Mathematica for Theoretical Physics: Classical Mechanics and Nonlinear Dynamics* (Springer-Verlag, New York, 2005a)
5. G. Baumann, *Mathematica for Theoretical Physics: Electrodynamics, Quantum Mechanics, General Relativity and Fractals* (Springer-Verlag, NewYork, 2005b)
6. F.P. Beer, E.R. Johnston, *Vector Mechanics for Engineers: Statics and Dynamics 5/e* (McGraw-Hill Publishing Company, New York, 1988)
7. F.P. Beer, E.R. Johnston, D.F. Mazurek, *Vector Mechanics for Engineers: Statics, 10/e* (McGraw-Hill Publishing Company, New York, 2012)
8. A.M. Bedford, W. Fowler, K.M. Liechti, *Statics and Mechanics of Materials* (Prentice Hall, Inc., Upper Saddle River, 2002)
9. A.M. Bedford, W. Fowler, *Engineering Mechanics: Statics, 5/e* (Prentice Hall, Inc., Upper Saddle River, 2007)
10. A.P. Boresi and R.J. Schmidt, *Engineering Mechanics: Statics* (PWS Publishing Company, Boston, 2000)
11. M.I. Buculei, *Mechanics* (University of Craiova Press, Craiova, 1974)
12. M.I. Buculei, D. Bagnaru, G. Nanu, D.B. Marghitu, *Analysis of Mechanisms with Bars* (Scrisul romanesc, Craiova, 1986)
13. I. Stroe et al., Analytical Mechanics Problems, University Politehnica of Bucharest, 1997
14. V. Ceausu, N. Enescu, F. Ceausu, *Mechanics Problems* (Printech, Bucuresti, 1999)
15. S.J. Chapman, *MATLAB Programming for Engineers* (Thomson Learning, Pacific Grove, CA, 2002)
16. S. Attaway, *MATLAB: A Practical Introduction to Programming and Problem Solving* (Butterworth-Heinemann, Elsevier, Amsterdam, 2012)
17. D.M. Etter, D.C. Kuncicky, *Introduction to MATLAB for Engineers and Scientists* (Prentice Hall, Inc., Upper Saddle River, 1996)
18. C. Iacob, *Theoretical Mechanics* (EDP, Bucharest, 1980)
19. J.H. Ginsberg, *Advanced Engineering Dynamics* (Cambridge University Press, Cambridge, 1995)
20. D.T. Greenwood, *Principles of Dynamics* (Prentice-Hall, Inc., Englewood Cliffs, 1998)
21. L.E. Goodman, W.H. Warner, *Statics* (Dover Publications Inc., Mineola, 2001)
22. R.C. Hibbeler, *Engineering Mechanics: Statics and Dynamics 13/e* (Prentice-Hall Inc., Upper Saddle River, 2013)
23. T.R. Kane, *Analytical Elements of Mechanics*, vol. 1 (Academic Press, New York, 1959)
24. T.R. Kane, *Analytical Elements of Mechanics*, vol. 2 (Academic Press, New York, 1961)
25. T.R. Kane, P.W. Likins, D.A. Levinson, *Spacecraft Dynamics* (McGraw-Hill, New York, 1983)
26. T.R. Kane, D.A. Levinson, *Dynamics* (McGraw-Hill, New York, 1985)

27. R. Maeder, *Programming in Mathematica* (Addison-Wesley Publishing Company, Redwood City, 1990)
28. N.H. Madsen, Statics and Dynamics, class notes, http://www.eng.auburn.edu/users/nmadsen/
29. D.B. Marghitu, *Mechanical Engineer's Handbook* (Academic Press, San Diego, 2001)
30. D.B. Marghitu, M.J. Crocker, *Analytical Elements of Mechanisms* (Cambridge University Press, Cambridge, 2001)
31. D.B. Marghitu, *Kinematic Chains and Machine Component Design* (Elsevier, Amsterdam, 2005)
32. D.B. Marghitu, *Mechanisms and Robots Analysis with MATLAB* (Springer, New York, 2009)
33. D.B. Marghitu, M. Dupac, *Advanced Dynamics: Analytical and Numerical Calculations with MATLAB* (Springer, New York, 2012)
34. D.B. Marghitu, Statics and Dynamics, class notes, http://www.eng.auburn.edu/users/marghitu/
35. D.J. McGill, W.W. King, *Engineering Mechanics: Statics and An Introduction to Dynamics* (PWS Publishing Company, Boston, 1995)
36. J.L. Meriam, L.G. Kraige, *Engineering Mechanics: Statics, 7/e* (Wiley, New York, 2011)
37. R.L. Mott, *Machine Elements in Mechanical Design* (Prentice Hall, Upper Saddle River, 1999)
38. R.L. Norton, *Machine Design* (Prentice-Hall, Upper Saddle River, 1996)
39. L.A. Pars, *A Treatise on Analytical Dynamics* (Wiley, New York, 1965)
40. M. Plesha, G. Gray, F. Costanzo, *Engineering Mechanics: Statics, 2/e* (McGraw-Hill Publishing Company, New York, 2012)
41. M. Radoi, E. Deciu, *Mechanics* (EDP, Bucharest, 1981)
42. W.F. Riley, L.D. Sturges, *Engineering Mechanics: Statics, 2/e* (Wiley, New York, 1995)
43. A. Ruina, R. Pratap, *Introduction to Statics and Dynamics* (Oxford University Press, Oxford, 2002)
44. A. Ripianu, P. Popescu, B. Balan, *Technical Mechanics* (EDP, Bucharest, 1979)
45. I.H. Shames, *Engineering Mechanics Statics, 4/e* (Prentice Hall Inc., Upper Saddle River, 1996)
46. S.D. Sheppard, B.H. Tongue, *Statics: Analysis and Design of Systems in Equilibrium* (Wiley, New York, 2005)
47. D. Smith, *Engineering Computation with MATLAB* (Pearson Education, Upper Saddle River, 2008)
48. R.W. Soutas-Little, D.J. Inman, *Engineering Mechanics: Statics and Dynamics* (Prentice-Hall, Upper Saddle River, 1999)
49. R.W. Soutas-Little, D.J. Inman, D. Balint, *Engineering Mechanics: Statics* (Cengage Learning, Independence, KY, 2007)
50. S. Staicu, *Theoretical Mechanics* (EDP, Bucharest, 1998)
51. A. Stan, M. Grumazescu, *Mechanics Problems* (EDP, Bucharest, 1973)
52. J. Sticklen, M.T. Eskil, *An Introduction to Technical Problem Solving with MATLAB* (Great Lakes Press, Wildwood, 2006)
53. A. Stoenescu, G. Silas, *Theoretical Mechanics* (ET, Bucharest, 1957)
54. J.H. Jackson, H.G. Wirtz, *Schaum's Outline of Theory and Problems of Statics and Strength of Materials* (McGraw-Hill Publishing Company, New York, 1983)
55. The MathWorks, https://www.mathworks.com/
56. Statics eBook, https://ecourses.ou.edu/
57. R. Voinea, D. Voiculescu, V. Ceausu, *Mechanics* (EDP, Bucharest, 1983)
58. V. Valcovici, S. Balan, R. Voinea, *Theoretical Mechanics* (ET, Bucharest, 1959)
59. K.J. Waldron, G.L. Kinzel, *Kinematics, Dynamics, and Design of Machinery* (Wiley, New York, 1999)
60. H.B. Wilson, L.H. Turcotte, D. Halpern, *Advanced Mathematics and Mechanics Applications Using MATLAB* (Chapman & Hall/CRC, Boca Raton, 2003)
61. J.H. Williams Jr, *Fundamentals of Applied Dynamics* (Wiley, New York, 1996)
62. S. Wolfram, *Mathematica, Wolfram Media* (Cambridge University Press, Cambridge, 1999)

Chapter 4
Equilibrium

4.1 Equilibrium Equations

A a body is in equilibrium when it is stationary or in steady translation relative to an inertial reference frame. The following conditions are satisfied when a body, acted upon by a system of forces and moments, is in equilibrium

1. the sum of the forces is zero
$$\sum \mathbf{F} = 0. \tag{4.1}$$

2. the sum of the moments about any point is zero
$$\sum \mathbf{M}_P = 0, \quad \forall P. \tag{4.2}$$

If the sum of the forces acting on a body is zero and the sum of the moments about one point is zero, then the sum of the moments about every point is zero.

Proof The body shown in Fig. 4.1, is subjected to forces $\mathbf{F}_{Ai}$, $i = 1, ..., n$, and couples $\mathbf{M}_j$, $j = 1, ..., m$. The sum of the forces is zero

$$\sum \mathbf{F} = \sum_{i=1}^{n} \mathbf{F}_{Ai} = \mathbf{0},$$

and the sum of the moments about a point P is zero

$$\sum \mathbf{M}_P = \sum_{i=1}^{n} \mathbf{r}_{PAi} \times \mathbf{F}_{Ai} + \sum_{j=1}^{m} \mathbf{M}_j = \mathbf{0},$$

where $\mathbf{r}_{PAi} = \overrightarrow{PA}_i$, $i = 1, ..., n$. The sum of the moments about any other point Q is

D. B. Marghitu et al., *Statics with MATLAB®*,
DOI: 10.1007/978-1-4471-5110-4_4, © Springer-Verlag London 2013

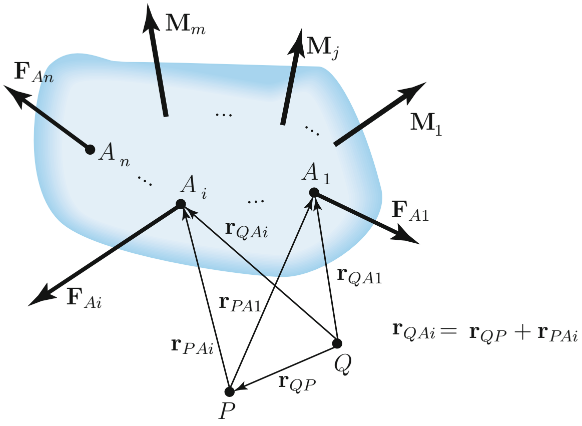


Fig. 4.1 Body subjected to forces $\mathbf{F}_{Ai}$ and couples $\mathbf{M}_j$

$$\begin{aligned}\sum \mathbf{M}_Q &= \sum_{i=1}^{n} \mathbf{r}_{QAi} \times \mathbf{F}_{Ai} + \sum_{j=1}^{m} \mathbf{M}_j \\ &= \sum_{i=1}^{n} \left(\mathbf{r}_{QP} + \mathbf{r}_{PAi}\right) \times \mathbf{F}_{Ai} + \sum_{j=1}^{m} \mathbf{M}_j \\ &= \mathbf{r}_{QP} \times \sum_{i=1}^{n} \mathbf{F}_{Ai} + \sum_{i=1}^{n} \mathbf{r}_{PAi} \times \mathbf{F}_{Ai} + \sum_{j=1}^{m} \mathbf{M}_j \\ &= \mathbf{r}_{QP} \times \mathbf{0} + \sum_{i=1}^{n} \mathbf{r}_{PAi} \times \mathbf{F}_{Ai} + \sum_{j=1}^{m} \mathbf{M}_j \\ &= \sum_{i=1}^{n} \mathbf{r}_{PAi} \times \mathbf{F}_{Ai} + \sum_{j=1}^{m} \mathbf{M}_j = \sum \mathbf{M}_P = \mathbf{0}.\end{aligned}$$

A body is subjected to concurrent forces $\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_n$ and no couples. If the sum of the concurrent forces is zero,

$$\mathbf{F}_1 + \mathbf{F}_2 + \cdots + \mathbf{F}_n = \mathbf{0},$$

the sum of the moments of the forces about the concurrent point is zero, so the sum of the moments about every point is zero. The only condition imposed by equilibrium on a set of concurrent forces is that their sum is zero.

4.2 Supports

4.2.1 Planar Supports

The *reactions* are forces and couples exerted on a body by its supports. The following force convention is defined: $\mathbf{F}_{ij}$ represents the force exerted by link i on link j.

Pin Support
Figure 4.2 shows a pin support. A beam 1 is attached by a smooth pin to a ground bracket 0. The pin passes through the bracket and the beam. The beam can rotate about the axis of the pin. The beam cannot translate relative to the bracket because the support exerts a reactive force that prevents this movement. The pin support is not capable of exerting a couple. Thus a pin support can exert a force on a body in any direction. The force of the pin support 0 on the beam 1 at point A, Fig. 4.3, is expressed in terms of its components in plane

$$\mathbf{F}_{01} = F_{01x}\,\mathbf{\imath} + F_{01y}\,\mathbf{\jmath}.$$

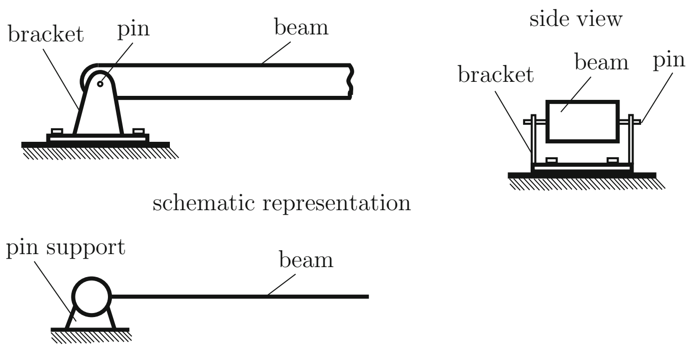


Fig. 4.2 Pin joint

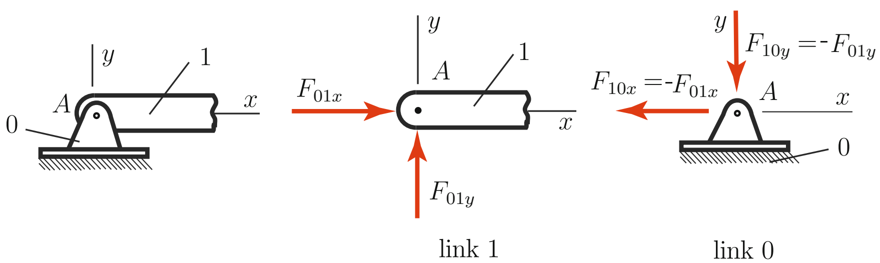


Fig. 4.3 Pin joint forces

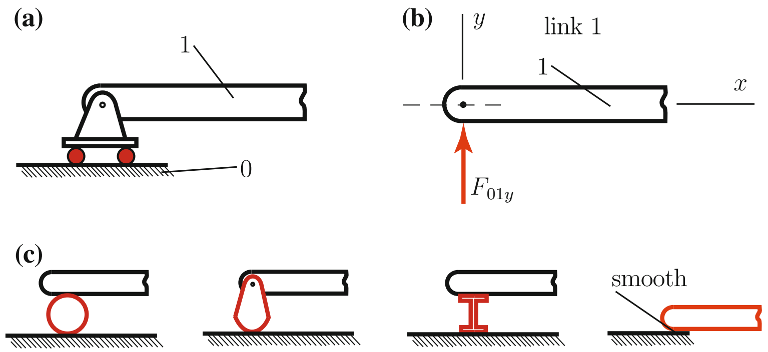


Fig. 4.4 Roller support

The directions of the reactions F_{01x} and F_{01y} are positive. If one determine F_{01x} or F_{01y} to be negative, the reaction is in the direction opposite to that of the arrow. The force of the beam 1 on the pin support 0 at point A, Fig. 4.3, is expressed

$$\mathbf{F}_{10} = F_{10x}\,\imath + F_{10y}\,\jmath = -F_{01x}\,\imath - F_{01y}\,\jmath,$$

where $F_{10x} = -F_{01x}$ and $F_{10y} = -F_{01y}$ The pin supports are used in mechanical devices that allow connected links to rotate relative to each other.

Roller Support

Figure 4.4a represents a roller support which is a pin support mounted on rollers.

The roller support 0 can only exert a force normal (perpendicular) to the surface 1 on which the roller support moves freely, Fig. 4.4b

$$\mathbf{F}_{01} = F_{01y}\,\jmath.$$

The roller support cannot exert a couple about the axis of the pin and it cannot exert a force parallel to the surface on which it translates. Figure 4.4c shows other schematic representations used for the roller support. A plane link on a smooth surface can also modeled by a roller support. Bridges and beams can be supported in this way and they will be capable of expansion and contraction.

Fixed Support

Figure 4.5 shows a fixed support or built-in support. The body is literally built into a wall. A fixed support 0 can exert two components of force and a couple on the link 1

$$\mathbf{F}_{01} = F_{01x}\imath + F_{01y}\jmath, \quad \text{and} \quad \mathbf{M}_{01} = M_{01z}\,\mathbf{k}.$$

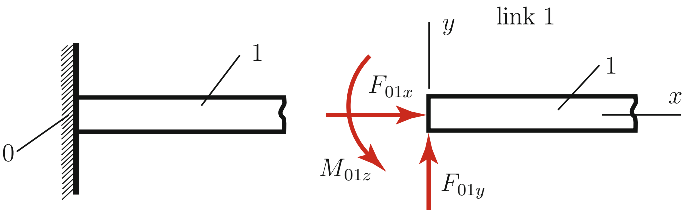


Fig. 4.5 Fixed support

4.2.2 Three-Dimensional Supports

Ball and Socket Support
Figure 4.6 shows a ball and socket support, where the supported body is attached to a ball enclosed within a spherical socket. The socket permits the body only to rotate freely. The ball and socket support cannot exert a couple to prevent rotation. The ball and socket support can exert three components of force

$$\mathbf{F}_{21} = F_{21x}\mathbf{\imath} + F_{21y}\mathbf{\jmath} + F_{21y}\mathbf{k}.$$

Bearing Support
The type of bearing shown in Fig. 4.7a supports a circular shaft while permitting it to rotate about its axis, z-axis. In the most general case, as shown in Fig. 4.7b, the bearing can exert a force on the supported shaft in each coordinate direction, $F_{21x},\ F_{21y},\ F_{21z}$, and can exert couples about axes perpendicular to the shaft, $M_{21x},\ M_{21y}$, but cannot exert a couple about the axis of the shaft. Situations can occur in which the bearing exerts no couples, or exerts no couples and no force parallel to the shaft axis as shown in Fig. 4.7c. Some radial bearings are designed in this way for specific applications.

4.3 Free-Body Diagrams

Free-body diagrams are used to determine forces and moments acting on simple bodies in equilibrium. The beam in Fig. 4.8a has a pin support at the left end A and a roller support at the right end B. The beam is loaded by a force F and a moment M at C. To obtain the free-body diagram first the beam is isolated from its supports. Next, the reactions exerted on the beam by the supports are shown on the free-body diagram, Fig. 4.8. Once the free-body diagram is obtained one can apply the equilibrium equations.

The steps required to determine the reactions on bodies are

1. draw the free-body diagram, isolating the body from its supports and showing the forces and the reactions;
2. apply the equilibrium equations to determine the reactions.

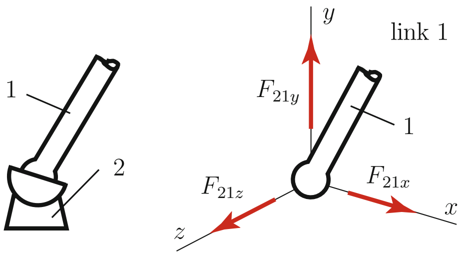


Fig. 4.6 Ball and socket support

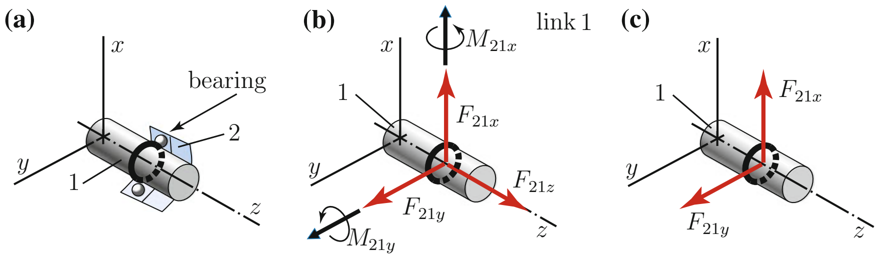


Fig. 4.7 Bearing support

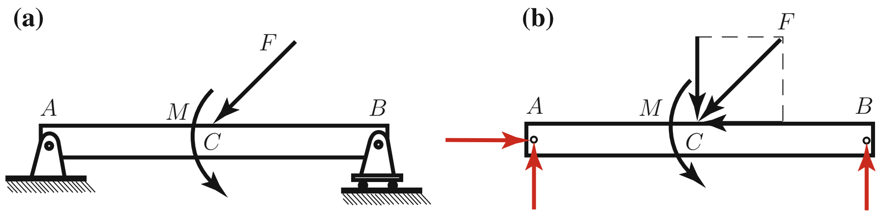


Fig. 4.8 Free-body diagram of a beam

For two-dimensional systems, the forces and moments are related by three scalar equilibrium equations

$$\sum F_x = 0, \tag{4.3}$$

$$\sum F_y = 0, \tag{4.4}$$

$$\sum M_P = 0, \quad \forall P. \tag{4.5}$$

One can obtain more than one equation from Eq. (4.5) by evaluating the sum of the moments about more than one point. The additional equations will not be independent of Eqs. (4.3)–(4.5). One cannot obtain more than three independent equilibrium equations from a two-dimensional free-body diagram, which means one can solve for at most three unknown forces or couples.

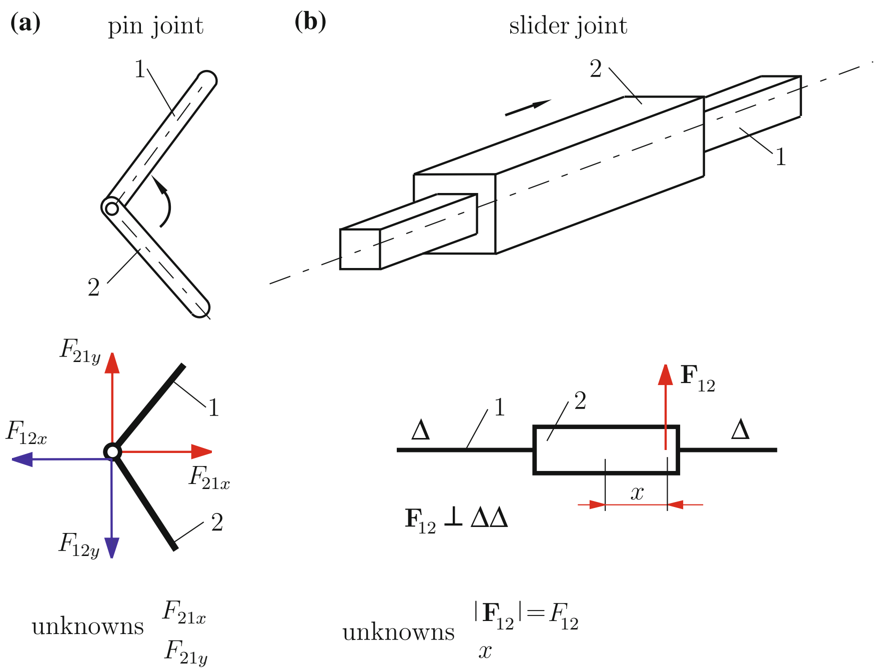


Fig. 4.9 Joint reaction forces

Free-Body Diagrams for Kinematic Chains
A free-body diagram is a drawing of a part of a complete system, isolated in order to determine the forces acting on that rigid body. The vector $\mathbf{F}_{ij}$ represents the force exerted by link i on link j and $\mathbf{F}_{ij} = -\mathbf{F}_{ji}$. Figure 4.9 shows the joint reaction forces for a pin joint, Fig. 4.9a, and a slider joint, Fig. 4.9b. Figure 4.10 shows various free-body diagrams that are considered in the analysis of a slider-crank mechanism Fig. 4.10a. In Fig. 4.10b, the free body consists of the three moving links isolated from the frame 0. The forces acting on the system include an external driven force $\mathbf{F}$, and the forces transmitted from the frame at joint A, $\mathbf{F}_{01}$, and at joint C, $\mathbf{F}_{03}$. Figure 4.10c is a free-body diagram of the two links 1 and 2 and Fig. 4.10d is a free-body diagram of the two links 0 and 1. Figure 4.10e is a free-body diagram of crank 1 and Fig. 4.10f is a free-body diagram of slider 3.

The force analysis can be accomplished by examining individual links or a subsystem of links. In this way the joint forces between links as well as the required input force or moment for a given output load are computed.

For three-dimensional systems, the forces and moments are related by six scalar equilibrium equations

$$\sum F_x = 0, \; \sum F_y = 0, \; \sum F_z = 0, \; \sum M_x = 0, \; \sum M_y = 0, \; \sum M_z = 0. \tag{4.6}$$

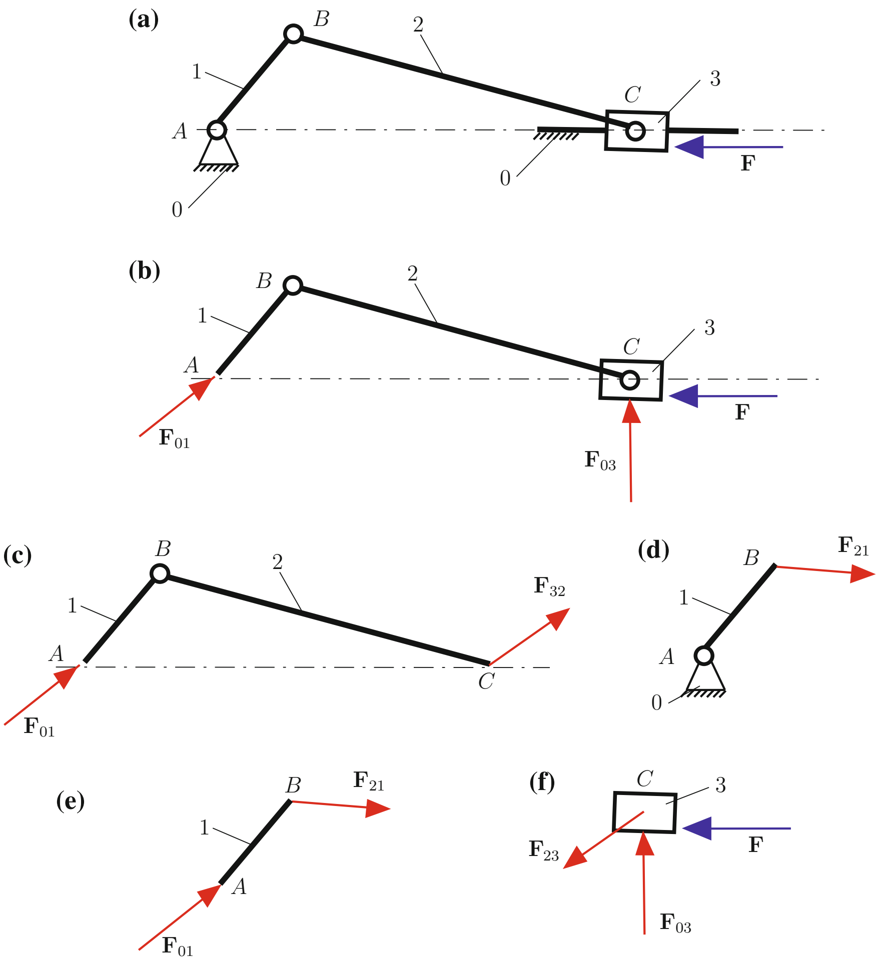


Fig. 4.10 Free-body diagrams of a slider-crank mechanism

One can evaluate the sums of the moments about any point. Although one can obtain other equations by summing the moments about additional points, they will not be independent of these equations. For a three-dimensional free-body diagram one can obtain six independent equilibrium equations and one can solve for at most six unknown forces or couples.

A body has *redundant supports* when the body has more supports than the minimum number necessary to maintain it in equilibrium. Redundant supports are used whenever possible for strength and safety. Each support added to a body results in additional reactions. The difference between the number of reactions and the number of independent equilibrium equations is called the *degree of redundancy*.

A body has *improper supports* if it will not remain in equilibrium under the action of the loads exerted on it. The body with improper supports will move when the loads are applied.

4.4 Two-Force and Three-Force Members

A body is a *two-force member* if the system of forces and moments acting on the body is equivalent to two forces acting at different points.

For example a body is subjected to two forces, $\mathbf{F}_A$ and $\mathbf{F}_B$, at A and B. If the body is in equilibrium, the sum of the forces equals zero only if $\mathbf{F}_A = -\mathbf{F}_B$. Furthermore, the forces $\mathbf{F}_A$ and $-\mathbf{F}_B$ form a couple, so the sum of the moments is not zero unless the lines of action of the forces lie along the line through the points A and B. Thus for equilibrium the two forces are equal in magnitude, are opposite in direction, and have the same line of action. However, the magnitude cannot be calculated without additional information.

A body is a *three-force member* if the system of forces and moments acting on the body is equivalent to three forces acting at different points.

Theorem If a three-force member is in equilibrium, the three forces are coplanar and the three forces are either parallel or concurrent.

Proof Let the forces $\mathbf{F_1}$, $\mathbf{F}_2$, and $\mathbf{F}_3$ acting on the body at A_1, A_2, and A_3. Let π be the plane containing the three points of application A_1, A_2, and A_3. Let $\Delta = A_1A_2$ be the line through the points of application of $\mathbf{F_1}$ and $\mathbf{F}_2$. Since the moments due to $\mathbf{F}_1$ and $\mathbf{F}_2$ about Δ are zero, the moment due to $\mathbf{F}_3$ about Δ must equal zero,

$$[\mathbf{n} \cdot (\mathbf{r} \times \mathbf{F}_3)]\, \mathbf{n} = [\mathbf{F}_3 \cdot (\mathbf{n} \times \mathbf{r})]\, \mathbf{n} = \mathbf{0},$$

where $\mathbf{n}$ is the unit vector of Δ. This equation requires that $\mathbf{F}_3$ be perpendicular to $\mathbf{n} \times \mathbf{r}$, which means that $\mathbf{F}_3$ is contained in π. The same procedure can be used to show that $\mathbf{F}_1$ and $\mathbf{F}_2$ are contained in π, so the forces $\mathbf{F}_1$, $\mathbf{F}_2$, and $\mathbf{F}_3$ are coplanar.

If the three coplanar forces are not parallel, there will be points where their lines of action intersect. Suppose that the lines of action of two forces $\mathbf{F}_1$ and $\mathbf{F}_2$ intersect at a point P. Then the moments of $\mathbf{F}_1$ and $\mathbf{F}_2$ about P are zero. The sum of the moments about P is zero only if the line of action of the third force, $\mathbf{F}_3$, also passes through P. Therefore either the forces are concurrent or they are parallel.

The analysis of a body in equilibrium can often be simplified by recognizing the two-force or three-force member.

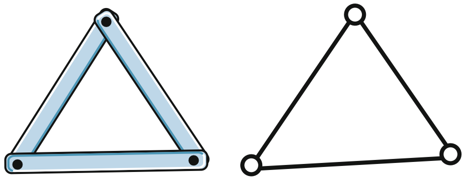


Fig. 4.11 Basic element of a plane truss, a triangle

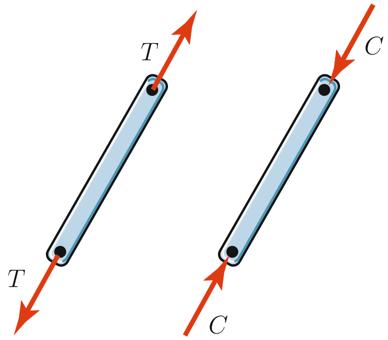


Fig. 4.12 Link in tension (T) and compression (C)

4.5 Plane Trusses

A structure composed of links joined at their ends to form a rigid structure is called a truss. Roof supports and bridges are common examples of trusses. When the links of the truss are in a single plane, the truss is called a plane truss. Three bars linked by pins joints at their ends form a rigid frame or noncollapsible frame. The basic element of a plane truss is the triangle, Fig. 4.11. Four, five or more bars pin-connected to form a polygon of as many sides form a nonrigid frame. A nonrigid frame is made rigid, or stable, by adding a diagonal bars and forming triangles. Frameworks built from a basic triangle are known as simple trusses. The truss is statically indeterminate when more links are present than are needed to prevent collapse. Additional links or supports which are not necessary for maintaining the equilibrium configuration are called redundant.

Several assumptions are made in the force analysis of simple trusses. First, all the links are considered to be two-force members. Each link of a truss is straight and has two nodes as points of application of the forces. The two forces are applied at the ends of the links and are necessarily equal, opposite, and collinear for equilibrium. The link may be in tension (T) or compression (C), as shown in Fig. 4.12.

The weight of the link is small compared with the force it supports. If the weight of the link is not small, the weight W of the member is replaced by two forces, each $W/2$ one force acting at each end of the member. These weight forces are considered as external loads applied to the pin connections. The connection between the links are assumed to be smooth pin joints. All the external forces are applied at the pin connections of the trusses. For large trusses, a roller support is used at one of the

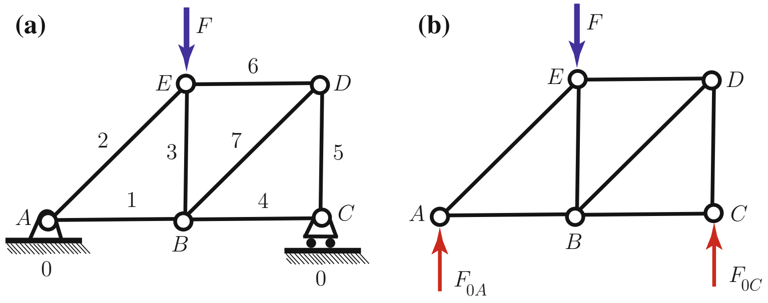


Fig. 4.13 Simple truss

supports to provide for expansion and contraction due to temperature changes and for deformation from applied loads. Trusses and frames in which no such provision is made are statically indeterminate. Two methods for the force analysis of simple trusses will be given. Each method will be explained for the simple truss shown in Fig. 4.13a. The length of the links are $AB = BE = ED = BC = CD = a$. The external force at E is given and has the magnitude of F. The free-body diagram of the truss as a whole is shown in Fig. 4.13b. The external support reactions are usually determined first, by applying the equilibrium equations to the truss as a whole. The reaction force of the ground 0 on the truss at the pin support A is $F_{0A} = F/2$ and the reaction force of the ground 0 on the truss at the roller support C is $F_{0C} = F/2$.

Method of Joints

This method for calculating the forces in the members consists of writing the conditions of equilibrium for the forces acting on the connecting pin of each joint. The method deals with the equilibrium of concurrent forces, and only two independent equilibrium equations are involved. The analysis starts with any joint where at least one known force exists and where not more than two unknown forces are located. For the truss shown in Fig. 4.13 the analysis begins with the pin at A. The force in each link is designated by one letter defining the node and one number defining the member. The proper directions of the forces should be evident by inspection for simple cases. The free-body diagram of the joint A is shown in Fig. 4.14a. Figure 4.14a indicates the process of the action and reaction in the members and joints. The force $\mathbf{F}_{1A}$ is the force of the member 1 (member AB) on the node A and is drawn acting away from the pin A. The force $\mathbf{F}_{A1}$ is the force of node A on the member 1 and $\mathbf{F}_{1A} = -\mathbf{F}_{A1}$, $F_{1A} = F_{A1}$. The tension in member 1 (force $\mathbf{F}_{A1}$) is indicated by an arrow away from the pin A (force $\mathbf{F}_{1A}$).

The force $\mathbf{F}_{2A}$ is the force of the member 2 (member AE) on the node A and is drawn toward the pin A. The force $\mathbf{F}_{A2}$ is the force of node A on the member 2 and $\mathbf{F}_{2A} = -\mathbf{F}_{A2}$, $F_{2A} = F_{A2}$. The compression (force $\mathbf{F}_{2A}$) is indicated by an arrow toward the pin A (force $\mathbf{F}_{2A}$).

The magnitudes of F_{1A} and F_{2A} are obtained from the conditions of equilibrium for the joint A

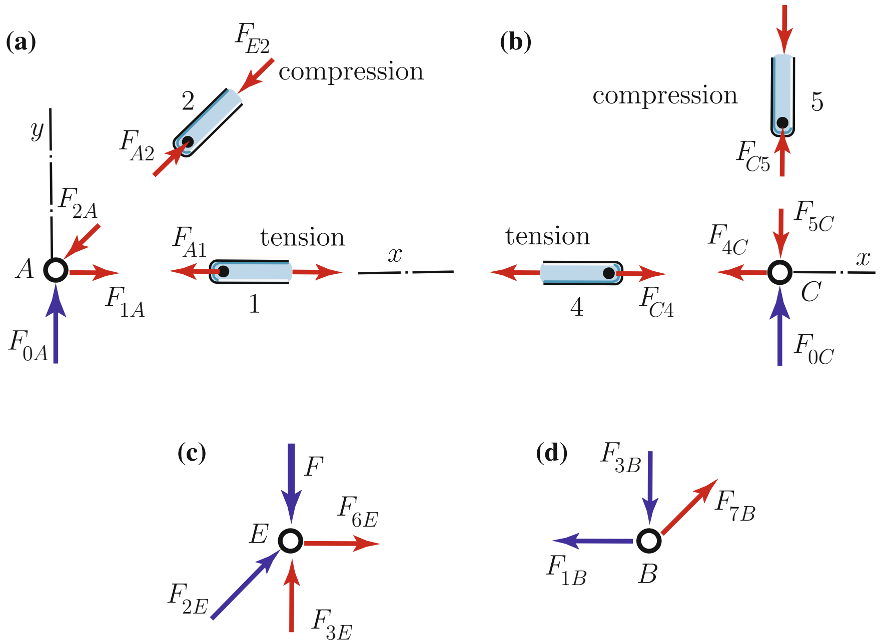


Fig. 4.14 Free-body diagrams of portions of members

$$\sum \mathbf{F}^A = \mathbf{F}_{0A} + \mathbf{F}_{1A} + \mathbf{F}_{2A} = \mathbf{0},$$

or

$$\sum F_x^A = 0 \text{ and } \sum F_y^A = 0,$$

or

$$-F_{2A}\sqrt{2}/2 + F_{1A} = 0 \text{ and } F_{0A} - F_{2A}\sqrt{2}/2 = 0,$$

or

$$F_{2A} = F\sqrt{2}/2 \text{ and } F_{1A} = F/2.$$

Joint C is analyzed next, Fig. 4.14b, since it contains only two unknowns, F_{4C} and F_{5C}

$$F_{4C} = 0 \text{ and } F_{5C} = F_C = F/2.$$

For joint E the force equilibrium conditions give

$$F_{6E} + F_{2E}\sqrt{2}/2 = 0 \text{ and } F_{3E} + F_{2E}\sqrt{2}/2 - F = 0,$$

or

$$F_{6E} = -F/2 \text{ and } F_{3E} = F/2.$$

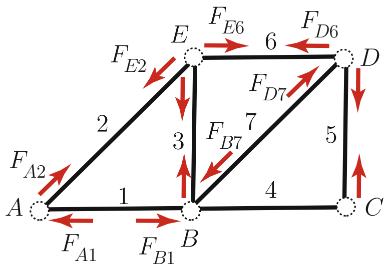


Fig. 4.15 Free-body diagrams of each joint

The force in the member F_{6E} is toward the pin E (compression). For joint B, Fig. 4.14c, the force equilibrium condition for y-axis gives

$$F_{7B}\sqrt{2}/2 - F_{3B} = 0 \text{ or } F_{7B} = F\sqrt{2}/2.$$

The correctness of the analysis is checked with the force equilibrium condition for x-axis

$$F_{7B}\sqrt{2}/2 - F_{1B} = 0.$$

Figure 4.15 shows the free-body diagram of each member. The method of joints for plane trusses employees only two of the three equilibrium equations because the method involves concurrent forces at each joint. A plane truss is statically determinate internally if $n + 3 = 2\,c$, where n is number of its links and c is the number of its joints.

Method of Sections

The method of sections has the advantage that the force in almost any member may be found directly from an analysis of a section which has cut that link. Since there are only three independent equilibrium equations in plane not more than three members whose forces are unknown should be cut. For the truss shown in Fig. 4.13 for ready reference the external reactions are first computed by considering the truss as a whole. The force in the members 6, 7, and 4 will be determined. An imaginary section, indicated by the dashed line, is passed through the truss, cutting it into two parts, Fig. 4.16. This section has cut three links whose forces F_{D6}, F_{D7}, and F_{C4} are initially unknown. The left-hand section is in equilibrium under the action of the external force F at E, the pin support reaction $F_{0A} = F/2$, and the three forces exerted F_{D6}, F_{D7}, and F_{C4} on the cut members by the right-hand section which has been removed. In general, the forces are represented with their proper senses by a visual approximation of the system in equilibrium. The proper senses will also result from the computations. The sum of the moments about point B for the left-hand section (LHS) gives

$$\sum M_B^{\text{LHS}} = F_{D6}\,a + F_{0A}\,a = 0 \text{ or } F_{D6} = F_{0A} = F/2.$$

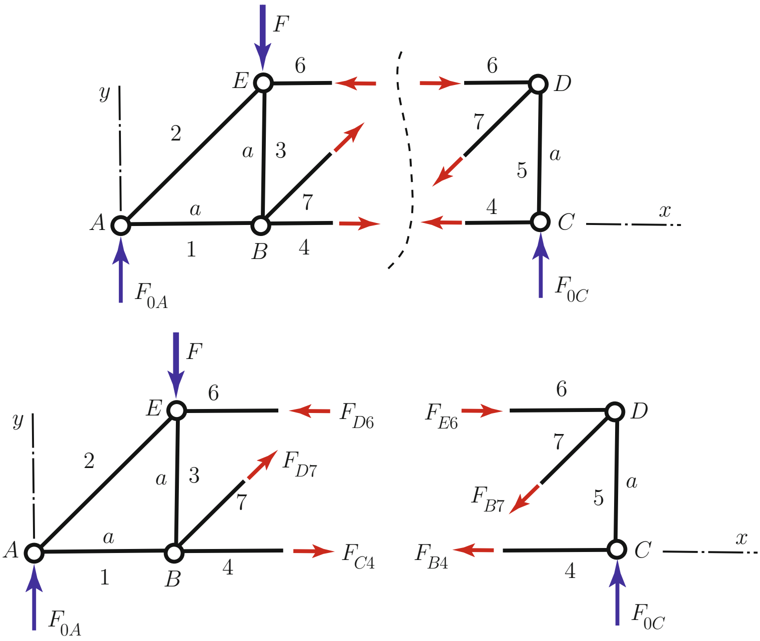


Fig. 4.16 Method of sections

The sum of the moments about point D for the right-hand section gives

$$\sum M_D^{\text{RHS}} = F_{B4}\, a + F_C\,(0) = 0 \;\text{ or }\; F_{B4} = 0.$$

The sum of the forces for right-hand section on x-axis is

$$F_{E6} - F_{B7}\,\sqrt{2}/2 = 0 \;\text{ or }\; F_{B7} = F_{E6}\,\sqrt{2} = F\,\sqrt{2}/2.$$

4.6 Particle on a Smooth Surface and on a Smooth Curve

Consider a particle at rest on a smooth surface. The equation of the surface in a cartesian reference is

$$f(x,\, y,\, z) = 0. \tag{4.7}$$

This is a ideal bilateral constraint for the particle. The mathematical expression for a unilateral constraint would be $f(x,\, y,\, z) \geq 0$. The constraint given by Eq. (4.7) reduces the number of degrees of freedom. The particle has only two degrees of freedom. The reaction of the surface on the particle is perpendicular to the surface and it is a vector collinear to grad $f = \nabla f$ The normal reaction is

$$\mathbf{N} = \lambda \nabla f = \lambda \left(\frac{\partial f}{\partial x} \mathbf{\imath} + \frac{\partial f}{\partial y} \mathbf{\jmath} + \frac{\partial f}{\partial z} \mathbf{k} \right), \quad \lambda \in \mathrm{R}. \tag{4.8}$$

The vectorial equilibrium equation for a particle on a surface is

$$\mathbf{F} + \lambda \nabla f = \mathbf{0}, \tag{4.9}$$

where **F** is the external force on the particle. The scalar equilibrium equations of a particle on a surface are

$$\begin{aligned} F_x + \lambda \frac{\partial f}{\partial x} &= 0, \\ F_y + \lambda \frac{\partial f}{\partial y} &= 0, \\ F_z + \lambda \frac{\partial f}{\partial z} &= 0. \end{aligned} \tag{4.10}$$

The surface equation given by Eq. (4.7) is added to the scalar equilibrium equations. From Eqs. (4.7) and (4.9) the coordinates $x,\ y,\ z$ determine the equilibrium position of the particle and the normal reaction is given by λ.

Consider a particle at rest on a smooth curve. The equation of the smooth curve in a cartesian reference frame is given as the intersection of two surfaces

$$f_1(x,\ y,\ z) = 0 \quad \text{and} \quad f_2(x,\ y,\ z) = 0. \tag{4.11}$$

The constraints given by Eq. (4.11) reduce the number of degrees of freedom of the particle to one. The reaction of the curve on the particle is perpendicular to the curve (normal reaction) and it is a vector calculated with

$$\begin{aligned} \mathbf{N} &= \mathbf{N}_1 + \mathbf{N}_2 = \lambda_1 \nabla f_1 + \lambda_2 \nabla f_2 \\ &= \lambda_1 \left(\frac{\partial f_1}{\partial x} \mathbf{\imath} + \frac{\partial f_1}{\partial y} \mathbf{\jmath} + \frac{\partial f_1}{\partial z} \mathbf{k} \right) + \lambda_2 \left(\frac{\partial f_2}{\partial x} \mathbf{\imath} + \frac{\partial f_2}{\partial y} \mathbf{\jmath} + \frac{\partial f_2}{\partial z} \mathbf{k} \right) \quad \lambda \in \mathrm{R}. \end{aligned} \tag{4.12}$$

The vectorial equilibrium equation for a particle on a smooth curve is

$$\mathbf{F} + \lambda_1 \nabla f_1 + \lambda_2 \nabla f_2 = \mathbf{0}. \tag{4.13}$$

The scalar equilibrium equations of a particle on the curve are

$$\begin{aligned} F_x + \lambda_1 \frac{\partial f_1}{\partial x} + \lambda_2 \frac{\partial f_2}{\partial x} &= 0, \\ F_y + \lambda_1 \frac{\partial f_1}{\partial y} + \lambda_2 \frac{\partial f_2}{\partial y} &= 0, \end{aligned}$$

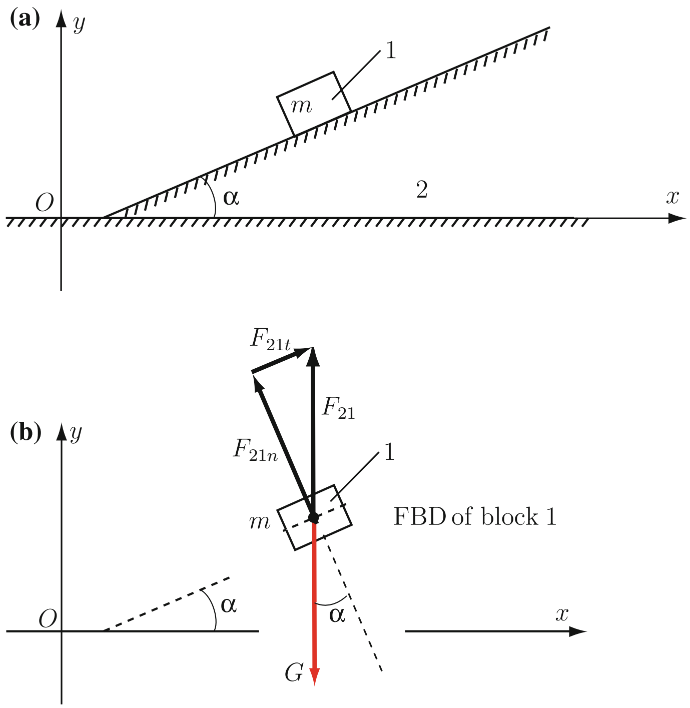


Fig. 4.17 Example 4.1

$$F_z + \lambda_1 \frac{\partial f_1}{\partial z} + \lambda_2 \frac{\partial f_2}{\partial z} = 0. \tag{4.14}$$

The curve equations given by Eq. (4.12) is added to the scalar equilibrium equations. From Eqs. (4.12) and (4.14) the coordinates $x,\ y,\ z$ determine the equilibrium position of the particle and the normal reaction is given by λ_1 and λ_2.

4.7 Examples

Example 4.1 A block 1 with the mass m is on an inclined plane 2 with the angle α with the horizontal, as shown in Fig. 4.17a. Find the normal (perpendicular to the plane) and the tangential (parallel to the plane) components of the reaction force of the inclined 2 plane on the block 1. The dimensions of the block are negligible. Numerical application: $m = 100$ kg, $g = 9.81$ m/s^2, and $\alpha = 30°$.

Solution The free-body diagram of the block 1 is shown in Fig. 4.17b and the reaction force of the inclined plane on the block is $\mathbf{F}_{21}$.

$$\mathbf{F}_{21} = \mathbf{F}_{21n} + \mathbf{F}_{21t}$$

The equilibrium equation for the block 1 is

$$\sum \mathbf{F} = \mathbf{0} \implies \mathbf{G} + \mathbf{F}_{21} = \mathbf{0},$$

or

$$\mathbf{G} + \mathbf{F}_{21n} + \mathbf{F}_{21t} = \mathbf{0}.$$

The normal component of the reaction force $\mathbf{F}_{21n}$ is at an angle of $\alpha = 30°$ with the gravitational force vector $\mathbf{G} = m\mathbf{g}$ and

$$F_{21n} = G \cos\alpha = mg \cos\alpha = 100(9.81)\cos 30° = 849.571 \text{ N}.$$

The parallel component to the plane is

$$F_{21t} = G \sin\alpha = mg \sin\alpha = 100(9.81)\sin 30° = 490.5 \text{ N}.$$

Example 4.2 Figure 4.18a shows a block of mass m supported by two cables AB and AC. The distance BO is a_1, the distance OC is a_2 and the distance AO is a_3. Find the tension in each cable. Numerical application: $m = 10\,\text{kg}$, $g = 9.81\,\text{m/s}^2$, $a_1 = 3\,\text{m}$, $a_2 = 5\,\text{m}$, and $a_3 = 1\,\text{m}$.

Solution The free-body diagram of the knot at A is shown in Fig. 4.18a with mg acting vertically down and the tensions in AC and AB. The force equilibrium equations are

$$\sum F_x = -T_{AB}\sin\theta_1 + T_{AC}\sin\theta_2 = 0, \tag{4.15}$$

$$\sum F_y = T_{AB}\cos\theta_1 + T_{AC}\cos\theta_2 = mg. \tag{4.16}$$

There are two equations with two unknowns. The problem is therefore statically determinate, i.e., it can be solved. From Eq. (4.15), $T_{AC} = \dfrac{\sin\theta_1}{\sin\theta_2} T_{AB}$. Substituting into Eq. (4.16) it results

$$T_{AB}\cos\theta_1 + \frac{\sin\theta_1}{\sin\theta_2} T_{AB}\cos\theta_2 = mg,$$

or

$$T_{AB} = \frac{mg}{\cos\theta_1 + \dfrac{\sin\theta_1}{\sin\theta_2}\cos\theta_2} = \frac{mg\sin\theta_2}{\cos\theta_1\sin\theta_2 + \sin\theta_1\cos\theta_2}.$$

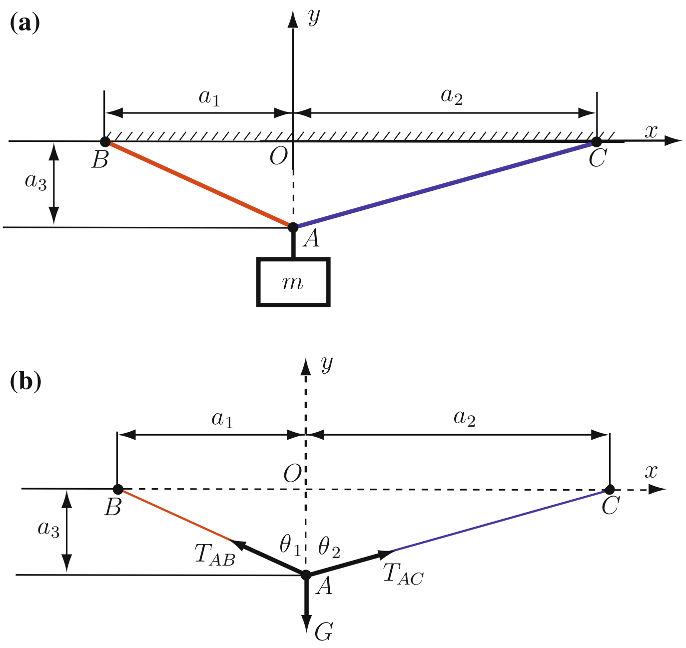


Fig. 4.18 Example 4.2

The trigonometric functions are $\sin\theta_1 = \dfrac{a_1}{l_{AB}}$, $\cos\theta_1 = \dfrac{a_3}{l_{AB}}$, $\sin\theta_2 = \dfrac{a_2}{l_{AC}}$, and $\cos\theta_2 = \dfrac{a_3}{l_{AC}}$, where $l_{AB} = \sqrt{a_1^2 + a_3^2}$ and $l_{AC} = \sqrt{a_2^2 + a_3^2}$.

It results

$$T_{AB} = mg\frac{a_2\sqrt{a_1^2 + a_3^2}}{a_3(a_1 + a_2)} = 10\,(9.81)\frac{5\sqrt{3^2 + 1^2}}{(1)(3 + 5)} = 193.887\text{ N},$$

and in a similar way

$$T_{AC} = mg\frac{a_1\sqrt{a_2^2 + a_3^2}}{a_3(a_1 + a_2)} = 10\,(9.81)\frac{3\sqrt{5^2 + 1^2}}{(1)(3 + 5)} = 187.58\text{ N}.$$

The same solution could also be obtained by writing an equilibrium moment equation with respect to a point that yields to one unknown. Suppose, for example, the moment equation is written about the point B. Then

$$\sum \mathbf{M}_B = \mathbf{r}_{BA} \times (\mathbf{T}_{AB} + \mathbf{T}_{AC} + \mathbf{G}) = \mathbf{r}_{BA} \times (\mathbf{T}_{AC} + \mathbf{G}) = \mathbf{0}, \quad (4.17)$$

where

$$\mathbf{G} = G\mathbf{\jmath} = -mg\mathbf{\jmath}, \quad \mathbf{T}_{AB} = T_{ABx}\mathbf{\imath} + T_{ABy}\mathbf{\jmath}, \quad \mathbf{T}_{AC} = T_{ACx}\mathbf{\imath} + T_{ACy}\mathbf{\jmath},$$

and

$$\mathbf{r}_{BA} \times \mathbf{T}_{AB} = \mathbf{0}. \tag{4.18}$$

The position vectors of the points A, B, and C are

$$\mathbf{r}_A = x_A\mathbf{\imath} + y_A\mathbf{\jmath} = -a_3\mathbf{\jmath}, \quad \mathbf{r}_B = x_B\mathbf{\imath} + y_B\mathbf{\jmath} = -a_1\mathbf{\imath}, \quad \mathbf{r}_C = x_C\mathbf{\imath} + y_C\mathbf{\jmath} = a_2\mathbf{\imath}.$$

Equation (4.17) becomes

$$\begin{aligned}\sum \mathbf{M}_B &= \mathbf{r}_{BA} \times \mathbf{T}_{AC} + \mathbf{r}_{BA} \times \mathbf{G} \\ &= \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ x_A - x_B & y_A - y_B & 0 \\ -T_{AC}\sin\theta_2 & T_{AC}\cos\theta_2 & 0 \end{vmatrix} + \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ x_A - x_B & y_A - y_B & 0 \\ 0 & G & 0 \end{vmatrix} \\ &= ((x_A - x_B)\,T_{AC}\cos\theta_2 - (y_A - y_B)\,T_{AC}\sin\theta_2)\,\mathbf{k} + (x_A - x_B)\,G\mathbf{k} \\ &= \left[(x_A - x_B)\,T_{AC}\cos\theta_2 - (y_A - y_B)\,T_{AC}\sin\theta_2 + (x_A - x_B)\,G\right]\mathbf{k} = \mathbf{0},\end{aligned}$$

or

$$(x_A - x_B)\,T_{AC}\cos\theta_2 - (y_A - y_B)\,T_{AC}\sin\theta_2 + (x_A - x_B)\,G = 0.$$

It results

$$T_{AC} = mg\frac{(x_A - x_B)}{(x_A - x_B)\cos\theta_2 - (y_A - y_B)\sin\theta_2}.$$

The unknown T_{AB} is calculated from a equilibrium moment equation of the system about the point C.

$$\sum \mathbf{M}_C = \mathbf{r}_{CA} \times (\mathbf{T}_{AB} + \mathbf{T}_{AC} + \mathbf{G}) = \mathbf{r}_{CA} \times (\mathbf{T}_{AB} + \mathbf{G}) = \mathbf{0},$$

and from the previous relation the tension $\mathbf{T}_{AB}$ is calculated. The MATLAB program for the second method is given by

```
rB_=[-a1, 0, 0];
rC_=[ a2, 0, 0];
rA_=[0,-a3,0];
SMB_=cross(rA_-rB_,TAC_+G_);
TACs=solve(SMB_(3),'TAC');
SMC_=cross(rA_-rC_,TAB_+G_);
TABs=solve(SMC_(3),'TAB');
```

and the results in MATLAB are:

```
sum M about B = rBA_ x (TAC_+G_) = 0_
TAC =

              2      2 1/2
  a1 g m (a2   + a3 )
  ---------------------
      a3 (a1 + a2)

TAC = 187.58 (N)

sum M about C = rCA_ x (TAB_+G_) = 0_
TAB =

              2      2 1/2
  a2 g m (a1   + a3 )
  ---------------------
      a3 (a1 + a2)

TAB = 193.887 (N)
```

Example 4.3 A particle P of mass m, shown in Fig. 4.19, is at rest on a plane given by the cartesian equation

$$f(x, y, z) = a\,x + b\,y + c\,z + d = 0.$$

The gravity force on the particle is $\mathbf{G} = -mg\,\mathbf{k}$, where g is the gravitational acceleration. A fixed point A of coordinates $x_A = y_A = 0$ and $z_A = h$ is on the $z-$axis. An attraction force proportional with the distance from P to A acts on the particle: $\mathbf{F} = k\,\mathbf{r}_{PA}$ where $k > 0$. Find the equilibrium position of the particle and the reaction force of the plane for this case. For the numerical application use $m = 10$ kg, $g = 9.8$ m/s^2, $a = 1$, $b = 1$, $c = 1$, $d = 1.5$, $h = 5$ m, and $k = 10$ N/m (Fig. 4.19).

Solution The input data and the forces on the particle are given in MATLAB as:

```
syms x y z a b c d h k m g lambda
f = a*x+b*y+c*z+d; % surface equation
rP_ = [x y z]; % position vector of the particle P
rA_ = [0 0 h]; % position vector of fixed point A
F_ = k*(rA_-rP_); % attraction force on the particle
G_ = [0 0 -m*g]; % gravity force on particle P
```

The normal reaction perpendicular to the surface f is

$$\mathbf{N} = \lambda\,\nabla f = \lambda\left(\frac{\partial f}{\partial x}\,\mathbf{\imath} + \frac{\partial f}{\partial y}\,\mathbf{\jmath} + \frac{\partial f}{\partial z}\,\mathbf{k}\right),$$

or in MATLAB:

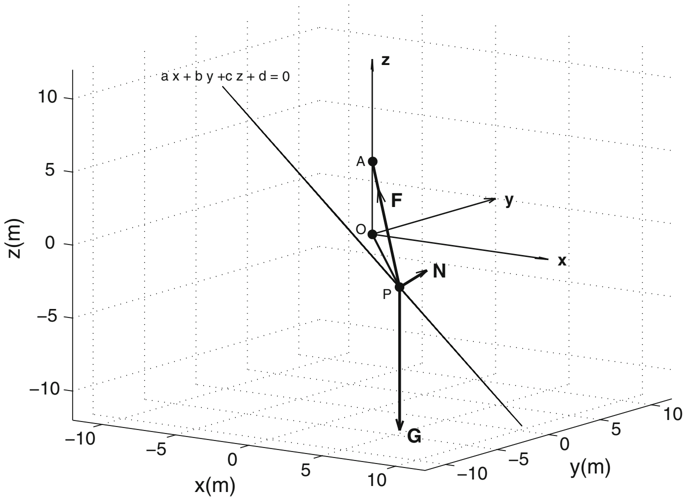


Fig. 4.19 Example 4.3

```
gradf_ = [diff(f,x), diff(f,y), diff(f,z)];
N_ = lambda*gradf_;
```

The sum of all the forces on the particle is: $\sum \mathbf{F} = \mathbf{F} + \mathbf{G} + \mathbf{N}$. The equilibrium equations are

$$\left(\sum \mathbf{F}\right) \cdot \mathbf{\imath} = 0,$$
$$\left(\sum \mathbf{F}\right) \cdot \mathbf{\jmath} = 0,$$
$$\left(\sum \mathbf{F}\right) \cdot \mathbf{k} = 0,$$
$$a\,x + b\,y + c\,z + d = 0,$$

or in MATLAB:

```
a*lambda - k*x = 0
b*lambda - k*y = 0
c*lambda - g*m + k*(h - z) = 0
d + a*x + b*y + c*z = 0
```

The equilibrium position for the particle and the reaction force given by λ are solved in MATLAB:

```
SF_ = F_+G_+N_;
sol=solve...
(SF_(1),SF_(2),SF_(3),f,'x,y,z,lambda');
xe = eval(sol.x);
```

```
ye = eval(sol.y);
ze = eval(sol.z);
lambdae = eval(sol.lambda);
Ne_ = lambdae*gradf_;
Ne = sqrt(simple(Ne_*Ne_.'));
```

and the results are:

```
x =

    a (d k + c h k - c g m)
  - -----------------------
             2    2    2
       k (a  + b  + c )

y =

    b (d k + c h k - c g m)
  - -----------------------
             2    2    2
       k (a  + b  + c )

z =

     2      2                 2    2
  h a  + h b  - c d    g m (a  + b )
  ----------------- - ----------------
     2    2    2           2    2    2
    a  + b  + c        k (a  + b  + c )

normal reaction force N_:

lambda =

    d k + c h k - c g m
  - -------------------
          2    2    2
         a  + b  + c

N_ =

  +-                                                   -+
  |        a #1              b #1              c #1     |
  |  - ------------, - ------------, - ------------     |
  |       2    2    2      2    2    2      2    2    2 |
  |      a  + b  + c      a  + b  + c      a  + b  + c  |
```

```
+-                                                          -+

where

    #1 == d k + c h k - c g m

N = |N_| =

 /                         2 \1/2
 | (d k + c h k - c g m)    |
 | ----------------------   |
 |       2    2    2        |
 \      a  + b  + c         /
```

Example 4.4 A particle P of mass m is on a parabola given by the cartesian equation

$$f = y + ax^2 = 0 \quad \text{and} \quad z = 0.$$

The gravity force on the particle is $\mathbf{G} = -mg\,\mathbf{j}$, where g is the gravitational acceleration. An attraction force proportional with the distance from P to $y-$axis acts on the particle: $\mathbf{F} = -k\,x\,\mathbf{\imath}$ where $k > 0$. Find the equilibrium position of the particle and the reaction force of the curve. For the numerical application use $a = 1\,\text{m}^{-1}$, $g = 9.8\,\text{m/s}^2$, $m = 10\,\text{kg}$, and $k = 10\,\text{N/m}$.

Solution The parabola equation and the forces on the particle are written in MATLAB as:

```
syms x y z a k m g lambda
f = y+a*x^2; % parabola equation
rP_ = [x y 0]; % particle position vector
F_ = -k*[rP_(1) 0 0]; % attraction force on particle
G_ = [0 -m*g 0]; % gravity force on particle P
```

The normal reaction perpendicular to f is

$$\mathbf{N} = \lambda\,\nabla f = \lambda\left(\frac{\partial f}{\partial x}\,\mathbf{\imath} + \frac{\partial f}{\partial y}\,\mathbf{\jmath} + \frac{\partial f}{\partial z}\,\mathbf{k}\right),$$

or in MATLAB:

```
gradf_ = [diff(f,x), diff(f,y), diff(f,z)];
N_ = lambda*gradf_;
```

The sum of all the forces on the particle is:

```
SF_ = F_ + G_ + N_;
```

and the equilibrium equations are:

```
2*a*lambda*x - k*x = 0 (1)
lambda - g*m = 0 (2)
y + a*x^2 = 0 (3)
```

The equilibrium position for the particle and the reaction force are determined in MATLAB:

```
sol=solve...
(SF_(1),SF_(2),f,'x,y,lambda');
xe = eval(sol.x);
ye = eval(sol.y);
lambde = eval(sol.lambda);
```

One equilibrium position, Fig. 4.20, is given by:

```
x = 0
y = 0
lambda = g*m
curve reaction force N_
Nx = 0
Ny = g*m
numerical application
N_ = [0,98.000,0] (N)
```

From the first equilibrium equation `2*a*lambda*x - k*x = 0` the equilibrium condition for any point on the curve is `k = 2*a*g*m`.

Example 4.5 A particle P of mass m is on a circle given by the cartesian equation

$$f = x^2 + y^2 - R^2 = 0 \quad \text{and} \quad z = 0.$$

The gravity force on the particle is $\mathbf{G} = -mg\,\mathbf{J}$, where g is the gravitational acceleration. An reaction force proportional with the distance from P to $y-$axis acts on the particle: $\mathbf{F} = k\,x\,\mathbf{I}$ where $k > 0$. Find the equilibrium position of the particle and the reaction force of the curve. For the numerical application use $R = 1\,\text{m}$, $g = 9.8\,\text{m/s}^2$, $m = 1\,\text{kg}$, and $k = 20\,\text{N/m}$.

Solution The circle equation and the forces on the particle are written in MATLAB as:

```
syms x y z R k m g lambda
f = x^2+y^2-R^2;
rP_ = [x y 0];
F_ = k*[x 0 0];
G_ = [0 -m*g 0];
```

The normal reaction perpendicular to f is:

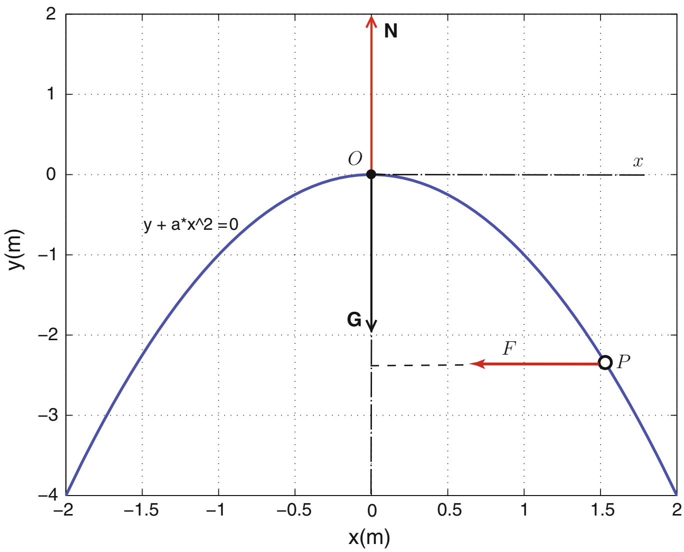


Fig. 4.20 Example 4.4

```
gradf_ = [diff(f,x), diff(f,y), diff(f,z)];
N_ = lambda*gradf_;
```

and the sum of all the forces on the particle is:

```
SF_ = F_ + G_ + N_;
```

The equilibrium equations are:

```
k*x + 2*lambda*x = 0
2*lambda*y - g*m = 0
x^2 - R^2 + y^2 = 0
```

There are four equilibrium positions, as shown in Fig. 4.21:

```
equilibrium position P1:

x = 0
y = R
lambda = (g*m)/(2*R)
Nx = 0
Ny = g*m
Nz = 0

equilibrium position P2:

x = 0
```

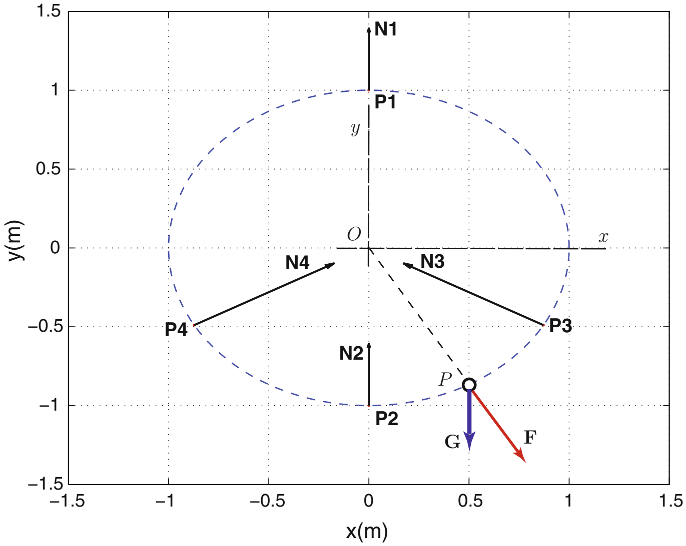


Fig. 4.21 Example 4.5

```
y = -R
lambda = -(g*m)/(2*R)
Nx = 0
Ny = g*m
Nz = 0

equilibrium position P3:

x = ((R*k + g*m)*(R*k - g*m))^(1/2)/k
y = -(g*m)/k
lambda = -k/2
Nx = -((R*k + g*m)*(R*k - g*m))^(1/2)
Ny = g*m
Nz = 0

equilibrium position P4:

x = -((R*k + g*m)*(R*k - g*m))^(1/2)/k
y = -(g*m)/k
lambda = -k/2
Nx = ((R*k + g*m)*(R*k - g*m))^(1/2)
Ny = g*m
Nz = 0
```

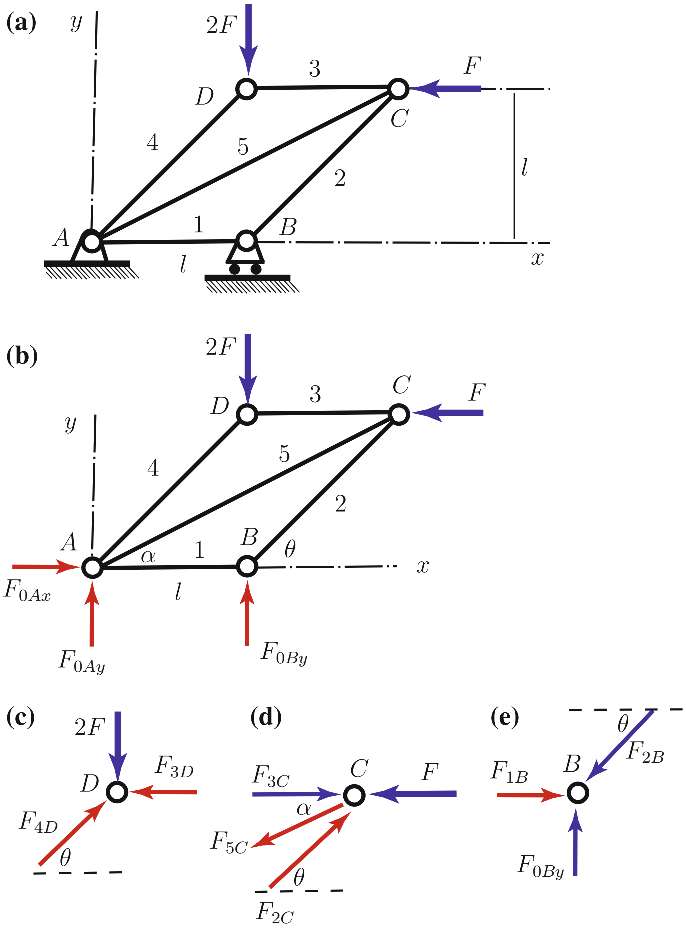


Fig. 4.22 Example 4.6

Example 4.6 Find the force in each member of the truss, shown in Fig. 4.22a, in terms of the external force F. State if the members are in tension or compression. For the numerical application use: $AB = CD = l = 1$ m and $F = 100$ daN.

Solution The free-body diagram of the truss is shown in Fig. 4.22b. For the method of joints the support reactions are not required for determining the member forces. The analysis begins with the pin at D. The free-body diagram of the joint D is shown in Fig. 4.22c. The force in each link is designated by one letter defining the node and one number defining the member. The proper directions of the forces should be evident by inspection for simple cases. The force $\mathbf{F}_{4D}$ is the force of the member 4 (member AD) on the node D. The force $\mathbf{F}_{3D}$ is the force of the member 3 (member

CD) on the node *D*. The magnitudes of F_{4D} and F_{3D} are obtained from the conditions of equilibrium for the joint *D*:

```
Fn =100.; % daN
syms F
theta=atan(1/2);
alpha=atan(1);
% sumFDy = F4D*sin(alpha)- 2*F = 0
F4D = 2*F/sin(alpha);
% sumFDx = F4D*cos(alpha)- F3D = 0
F3D = F4D*cos(alpha);
```

and the results are:

```
% F4D = 2*2^(1/2)*F  = 282.843 (daN) =>
% 4=AD (Compression)
% F3D = 2*F  = 200.000 (daN) =>
% 3=DC (Compression)
```

The free-body diagram of the joint *C* is shown in Fig. 4.22d. The magnitudes of F_{2C} and F_{5C} are obtained from the equilibrium conditions for the joint *C*:

```
syms F2C F5C
F3C = F3D;
% sumFCx = F3C-F5C*cos(theta)+F2C*cos(alpha)-F=0
sumFCx = F3C-F5C*cos(theta)+F2C*cos(alpha)-F;
% sumFCy = -F5C*sin(theta)+F2C*sin(alpha)=0
sumFCy = -F5C*sin(theta)+F2C*sin(alpha);
solC=solve(sumFCx,sumFCy,'F2C','F5C');
F2Cs = solC.F2C;
F5Cs = solC.F5C;
```

The results are:

```
% F2C = 2^(1/2)*F  = 141.421 (daN) =>
% 2=BC (Compression)
% F5C = 5^(1/2)*F  = 223.607 (daN) =>
% 5=AC (Tension)
```

From the free-body diagram of the joint *B*, shown in Fig. 4.22e, the magnitude of F_{1B} is calculated:

```
F2B = F2Cs;
% sumFBx = F1B-F2B*cos(alpha)=0
F1B=F2B*cos(alpha);
```

and the result is:

```
% F1B = F  = 100.000 (daN) =>
% 1=AB (Tension)
```

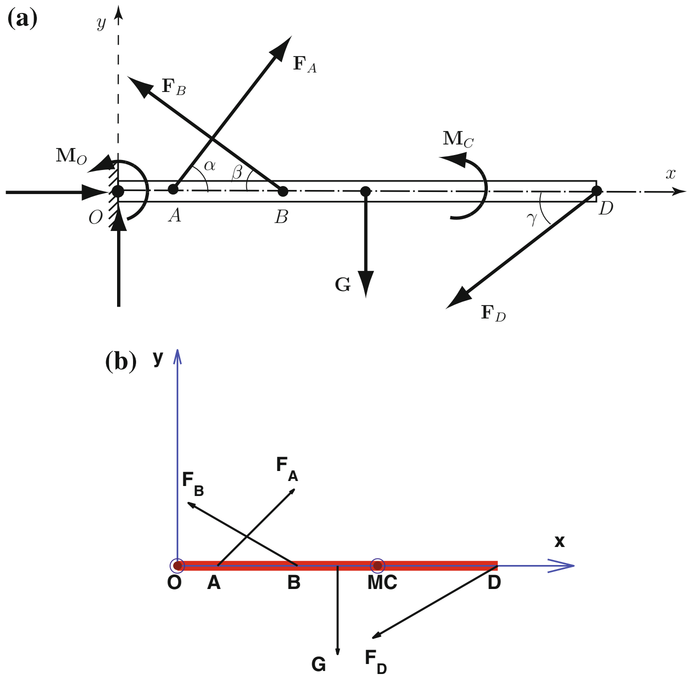


Fig. 4.23 Example 4.7 (**a**) free-body diagram and (**b**) MATLAB figure

Example 4.7 A horizontal uniform cantilever beam OD has the length l, the mass m, and is fixed at O. The forces $\mathbf{F}_A$, $\mathbf{F}_B$, and $\mathbf{F}_D$ act on the beam at the points A, B, and D. The angles of these external forces with the horizontal are α, β, and γ as shown in Fig. 4.23. A counter-clockwise couple $\mathbf{M}_C$ is applied on the beam. Find the reaction force and the reaction moment at the support O. For the numerical application use:

```
% xA=1; xB=3; xD=8; (m)
% FA=3000; FB=3500; FD= 4000; (N)
% alpha=pi/4; beta=pi/6; gamma=pi/6; (rad)
% MC=10*10^3; (N m)
% m=250; (kg)
% g=8.81;(m/s^2)
```

Solution The weight of the beam $\mathbf{G}$ is acting at the midpoint of the beam. In order to determine vertical and horizontal components of the reaction force, the following equations are written. The sum of all the forces acting on the beam is zero, that is:

$$\sum \mathbf{F} = \mathbf{F}_O + \mathbf{F}_A + \mathbf{F}_B + \mathbf{F}_D + \mathbf{G} = \mathbf{0},$$

or equivalent

$$\sum F_x = F_{Ox} + F_{Ax} + F_{Bx} + F_{Dx} + G_x = 0,$$
$$\sum F_y = F_{Oy} + F_{Ay} + F_{By} + F_{Dy} + G_y = 0.$$

The component of $\mathbf{F}_A$ on the $x-$axis is $F_{Ax} = F_A \sin\alpha$ and on the $y-$axis is $F_{1y} = F_A \cos\alpha$. The components of $\mathbf{F}_B$ are $F_{Bx} = -F_B \sin\beta$ and $F_{By} = F_B \cos\beta$. The components of $\mathbf{F}_D$ are $F_{Dx} = -F_D \sin\gamma$ and $F_{Dy} = -F_D \cos\gamma$. The components of weight $\mathbf{G}$ are $G_x = 0$ and $G_y = -mg$. The MATLAB commands for the forces are:

```
syms FA alpha
FAx=FA*cos(alpha);
FAy=FA*sin(alpha);
FA_=[FAx,FAy,0];
syms FB beta
FBx=-FB*cos(beta);
FBy=FB*sin(beta);
FB_=[FBx,FBy,0];
syms FD gamma
FDx=-FD*cos(gamma);
FDy=-FD*sin(gamma);
FD_=[FDx,FDy,0];
syms m g
G_=[0,-m*g,0];
syms FOx FOy
FO_=[FOx,FOy,0];
```

From the equilibrium equations the reaction force at O is

$$F_{Ox} = -F_A \cos\alpha + F_B \cos\beta + F_D \cos\gamma,$$
$$F_{Oy} = -F_A \sin\alpha - F_B \sin\beta + F_D \sin\gamma + mg,$$

or in MATLAB:

```
FO_=-(G_+FA_+FB_+FD_);
FOx = FO_(1);
FOy = FO_(2);
```

The sum of all moments of all forces and moments on the beam about O is zero

$$\sum \mathbf{M} = \mathbf{M}_O + \mathbf{r}_A \times \mathbf{F}_A + \mathbf{r}_G \times \mathbf{G} + \mathbf{r}_B \times \mathbf{F}_B + \mathbf{M}_C + \mathbf{r}_D \times \mathbf{F}_D = \mathbf{0},$$

or

$$\sum M = M_O + x_A F_{Ay} + x_G G_y + x_B F_{By} + M_C + x_D F_{Dy}$$
$$= M_O + x_A F_A \sin\alpha - \frac{l}{2} mg + x_B F_B \sin\beta + M_C - x_D F_D \sin\gamma = 0$$

where $x_G = x_D/2 = l/2$. It results

$$M_O = -x_A F_A \sin\alpha + \frac{l}{2} mg - x_B F_B \sin\beta - M_C + l F_D \sin\gamma.$$

The numerical values for the reaction at O are:

```
FOx =  4.374 (kN)
FOy =  0.581 (kN)
MOz =  8.439 (kN m)
```

Example 4.8 The vertical shaft AB, shown in Fig. 4.24, is mounted through bearings at A and B and is supporting a uniform rectangular plate $ABED$ with mass m and edges length $AB = DE = h$ and $AD = BE = b$. The mass of the shaft is negligible and the mass of the plate is m. The distance between the upper bearing located at B and the lower bearing located at A is equal with h. The bearing at A supports the entire vertical load. A moment of magnitude M_e is applied to shaft in the vertically upward direction. The plate is constrained from rotating about the vertical axis by the action of a cable attached to outside corner of the plate denoted by D. The other end of the cable is attached to a fixed support point P that is in a perpendicular line PA to the pate. The perpendicular distance from the cable attachment point, P, to the plate is equal to $PA = a$. Find the bearing reaction forces and the tension in the cable PD. For the numerical application use $h = 0.6\,\text{m}$, $b = 0.6\,\text{m}$, $a = 0.5\,\text{m}$, $m = 100\,\text{kg}$, $M_e = 120\,\text{N m}$, and $g = 9.81\,\text{m/s}^2$.

Solution Both bearings A and B can exert forces perpendicular to the shaft direction on the shaft but are such that individually they provide no resistance to rotational movements (no couples exerted by the bearings). The bearing forces are contained in a horizontal plane. The plate is uniform and its center of mass is at its geometric center. The weight of the plate is equal to its mass, m, multiplied by the gravitational acceleration g. This force acts vertically downward through the mass center of the plate G.

A reference frame $Oxyz$ with the origin located at the bearing A and having the z-axis directed upward along the shaft is chosen. The plate is contained in the yz plane such that the positive $x-$axis will be directed from the shaft toward the cable tie-down point P as shown in Fig. 4.24.

Considering the mechanical system to be the shaft-plate combination, the bearing force of interest at A and B acts on the mechanical system. The weight of the plate as well as the known couple (moment) acts on the mechanical system. In addition to the

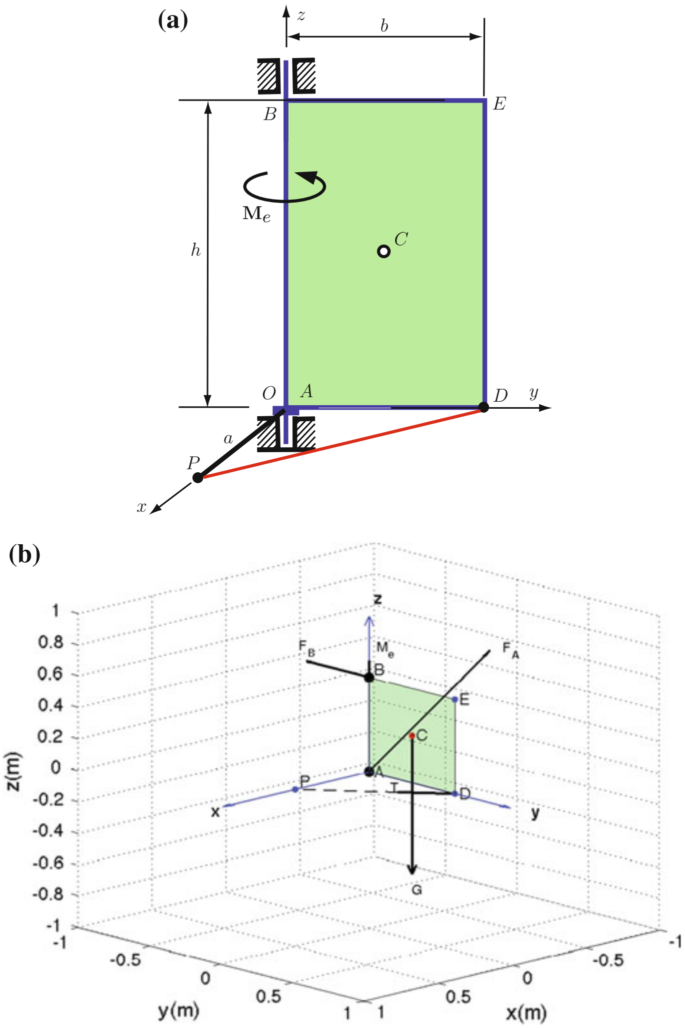


Fig. 4.24 Example 4.8 (**a**) Mechanical system and (**b**) MATLAB figure

two unknown horizontal force components at the bearing B, the mechanical system will also be acted upon by the cable force on the plate, and the three bearing reactions at A (the vertical support on the shaft plus the two horizontal bearing components).

In order to determine vertical and horizontal components of the reaction force, the following equations are written. The sum of all the forces acting on the system is zero, that is

$$\sum \mathbf{F} = \mathbf{F}_A + \mathbf{F}_B + \mathbf{G} + \mathbf{T} = \mathbf{0},$$

or equivalent

$$\sum \mathbf{F} = F_{Ax}\,\mathbf{\imath} + F_{Ay}\,\mathbf{\jmath} + F_{Az}\mathbf{k} + F_{Bx}\,\mathbf{\imath} + F_{By}\,\mathbf{\jmath} + T\,\mathbf{r}_{DP}/|\mathbf{r}_{DP}| - mg\,\mathbf{k} = \mathbf{0},$$

or equivalent

$$\sum F_x = F_{Ax} + F_{Bx} + T_x = 0,$$
$$\sum F_y = F_{Ay} + F_{By} + T_y = 0,$$
$$\sum F_z = F_{Az} - mg = 0,$$

where

$$T_x = T\frac{x_P - x_D}{\sqrt{\left((x_P - x_D)^2 + (y_P - y_D)^2\right)}},$$
$$T_y = T\frac{y_P - y_D}{\sqrt{\left((x_P - x_D)^2 + (y_P - y_D)^2\right)}}.$$

The sum of moments of all forces about point A is zero, that is

$$\sum \mathbf{M}_A = \mathbf{M}_e + \mathbf{r}_{AB} \times \mathbf{F}_B + \mathbf{r}_{AD} \times \mathbf{T} + \mathbf{r}_{AC} \times \mathbf{G} = \mathbf{0},$$

or equivalent

$$\sum \mathbf{M}_A = M_e\mathbf{k} + \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ x_B & y_B & z_B \\ F_{Ax} & F_{Ay} & 0 \end{vmatrix} + \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ x_D & y_D & 0 \\ T_x & T_y & 0 \end{vmatrix} + \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ x_C & y_C & z_C \\ 0 & 0 & -mg \end{vmatrix} = \mathbf{0}.$$

The equilibrium equations in MATLAB are:

```
h=0.6; % m
b=0.6; % m
a=0.5; % m
m = 100.;  % kg
g = 9.81;  % m/s^2
Me = 120.; % N m

xA=0; yA=0; zA=0;     % m
xB=0; yB=0; zB=h;     % m
xD=0; yD=b; zD=0;     % m
xE=0; yE=b; zE=h;     % m
xC=0; yC=b/2;zC=h/2;  % m
xP=a; yP=0; zP=0;     % m

rA_=[xA yA zA];
```

```
rB_=[xB yB zB];
rD_=[xD yD zD];
rE_=[xE yE zE];
rC_=[xC yC zC];
rP_=[xP yP zP];

G_ = [0 0 -m*g];
syms FAx FAy FAz
FA_ = [FAx,FAy,FAz];
syms FBx FBy
FB_ = [FBx,FBy,0];
u_=(rP_-rD_)/norm(rP_-rD_);
syms T
T_ = T*u_;
SF_ = FA_ + FB_ + G_ + T_;

Me_=[0,0,Me];
SMA_ = ...
cross(rC_,G_)+cross(rD_,T_)+cross(rB_,FB_)+Me_;
```

The six equilibrium equations are solved with MATLAB:

```
sol=solve(...
    SF_(1) , SF_(2), SF_(3),...
    SMA_(1),SMA_(2),SMA_(3));
FAxs=eval(sol.FAx);
FAys=eval(sol.FAy);
FAzs=eval(sol.FAz);
FBxs=eval(sol.FBx);
FBys=eval(sol.FBy);
Ts=eval(sol.T);
```

and the results are:

```
FAx= -200.000 (N)
FAy= 730.500 (N)
FAz= 981.000 (N)
FBx=  0.000 (N)
FBy= -490.500 (N)
T= -312.410 (N)
```

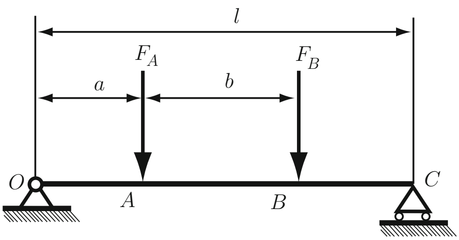


Fig. 4.25 Problem 4.1

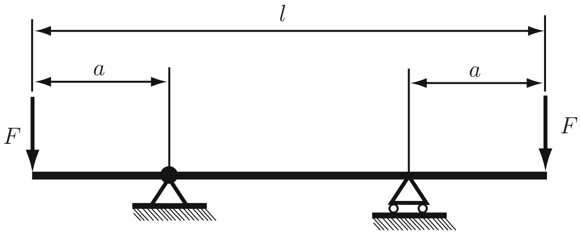
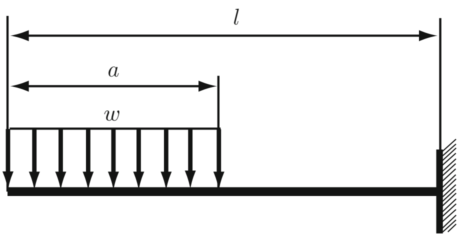


Fig. 4.26 Problem 4.2

Fig. 4.27 Problem 4.3



4.8 Problems

4.1 The beam shown in Fig. 4.25 is loaded with the concentrated forces $F_1 = 100\,\text{N}$ and $F_2 = 500\,\text{N}$. The following dimensions are given: $a = 0.5\,\text{m}$, $b = 0.3\,\text{m}$, and $l = 1\,\text{m}$. Find the reactions at the supports O and C.

4.2 The beam depicted in Fig. 4.26 is loaded with the two concentrated forces with the magnitude $F = 200\,\text{lbs}$. The dimensions of the beam are given: $a = 5$ in and $l = 1\,\text{ft}$. Find the reactions at the supports.

4.3 Consider the cantilever beam of Fig. 4.27, subjected to a uniform load distributed, $w = 100\,\text{N/m}$, over a portion of its length. The dimensions of the beam are: $a = 10\,\text{cm}$ and $l = 1\,\text{m}$. Find the support reaction on the beam.

4.4 A smooth sphere of mass m is resting against a vertical surface and an inclined surface that makes an angle θ with the horizontal, as shown in Fig. 4.28. Find the forces exerted on the sphere by the two contacting surfaces.

Fig. 4.28 Problem 4.4

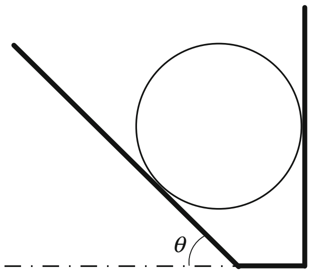
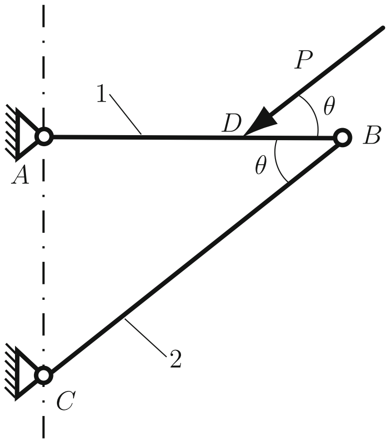


Fig. 4.29 Problem 4.5



Numerical application: (a) $m = 10\,\text{kg}$, $\theta = 30°$, and $g = 9.8\,\text{m/s}^2$; (b) $m = 2$ slugs, $\theta = 60°$, and $g = 32.2\,\text{ft/sec}^2$.

4.5 The links 1 and 2 shown in Fig. 4.29 are each connected to the ground at A and C, and to each other at B using frictionless pins. The length of link 1 is $AB = l$. The angle between the links is $\angle ABC = \theta$. A force of magnitude P is applied at the point D ($AD = 2l/3$) of the link 1. The force makes an angle θ with the horizontal. Find the force exerted by the lower link 2 on the upper link 1. Numerical application: (a) $l = 1\,\text{m}$, $\theta = 30°$, and $P = 1000\,\text{N}$; (b) $l = 2\,\text{ft}$, $\theta = 45°$, and $P = 500\,\text{lb}$.

4.6 The shaft shown in Fig. 4.30 turns in the bearings A and B. The dimensions of the shaft are $a = 6\,\text{in.}$ and $b = 3\,\text{in.}$ The forces on the gear attached to the shaft are $F_t = 900\,\text{lb}$ and $F_r = 500\,\text{lb}$. The gear forces act at a radius $R = 4\,\text{in.}$ from the axis of the shaft. Find the loads applied to the bearings.

4.7 The shaft shown in Fig. 4.31 turns in the bearings A and B. The dimensions of the shaft are $a = 120\,\text{mm}$ and $b = 30\,\text{mm}$. The forces on the gear attached to the shaft are $F_t = 4500\,\text{N}$, $F_r = 2500\,\text{N}$, and $F_a = 1000\,\text{N}$. The gear forces act at a radius $R = 100\,\text{mm}$ from the shaft axis. Determine the bearings loads.

4.8 The dimensions of the shaft shown Fig. 4.32 are $a = 2\,\text{in.}$ and $l = 5\,\text{in.}$ The force on the disk with the radius $r_1 = 5\,\text{in.}$ is $F_1 = 600\,\text{lb}$ and the force on the

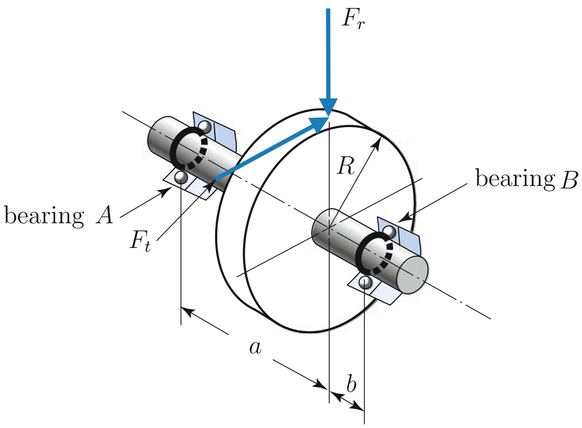
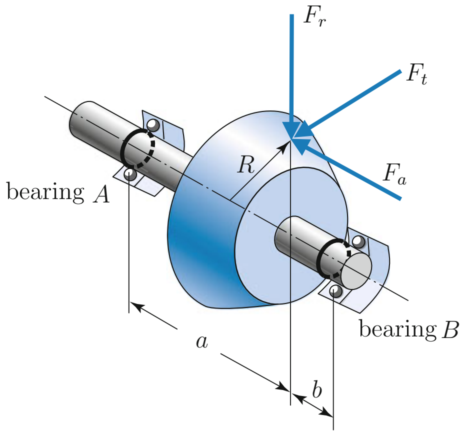


Fig. 4.30 Problem 4.6

Fig. 4.31 Problem 4.7



disk with the radius $r_2 = 2.5$ in. is $F_2 = 1200$ lb. Determine the forces on the bearings at A and B.

4.9 The dimensions of the shaft shown Fig. 4.33 are $a = 50$ mm and $l = 120$ mm. The force on the disk with the radius $r_1 = 50$ mm is $F_1 = 4000$ N and the force on the disk with the radius $r_2 = 100$ mm is $F_2 = 2000$ N. Determine the bearing loads at A and B.

4.10 The force on the gear in Fig. 4.34 is $F = 1.5$ kN and the radius of the gear is $R = 60$ mm. The dimensions of the shaft are $l = 300$ mm and $a = 60$ mm. Determine the bearing loads at A and B.

4.11 A torque (moment) of 24 N m is required to turn the bolt about its axis, as shown in Fig. 4.35, where $d = 120$ mm and $l = 14$ mm. Determine P and the forces between the smooth hardened jaws of the wrench and the corners of A and B

Fig. 4.32 Problem 4.8

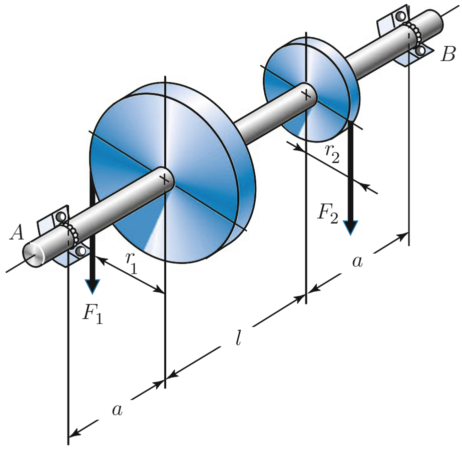
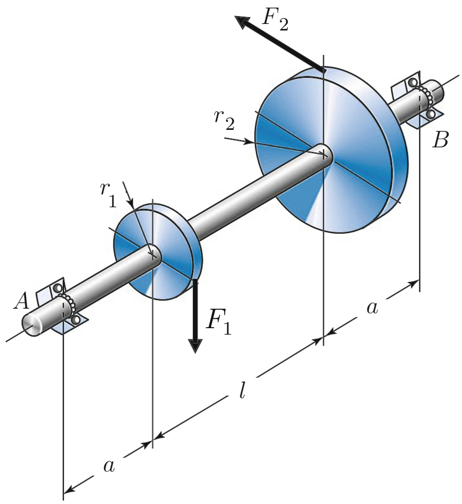


Fig. 4.33 Problem 4.9



of the hexagonal head. Assume that the wrench fits easily on the bolt so that contact is made at corners *A* and *B* only.

4.9 Programs

4.9.1 Program 4.2

```
% example 4.2
% equilibrium of a particle
```

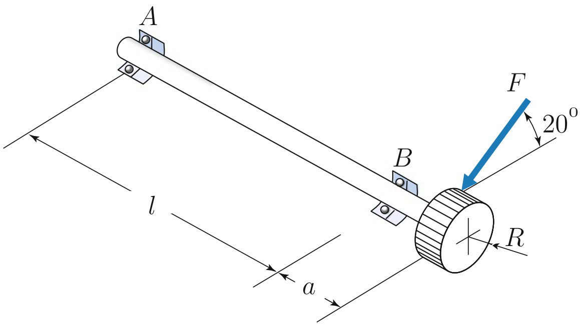
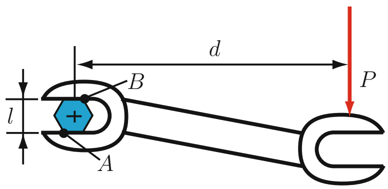


Fig. 4.34 Problem 4.10

Fig. 4.35 Problem 4.11




```
clear all; clc; close

syms a1 a2 a3 TAB TAC m g

list={m, g, a1, a2, a3 };
listn={10, 9.81, 3, 5, 1 };

lAB=sqrt(a1^2+a3^2);
lAC=sqrt(a2^2+a3^2);

stheta1=a1/lAB;
ctheta1=a3/lAB;
stheta2=a2/lAC;
ctheta2=a3/lAC;

TAB_ = [ -TAB*stheta1, TAB*ctheta1, 0];
TAC_ = [  TAC*stheta2, TAC*ctheta2, 0];
G_ = [0, -m*g, 0];

fprintf('Method I \n')
% SF_ = TAB_ + TAC_ + G_ = 0
fprintf('sum forces = TAB_ + TAC_ + G_ = 0_ \n')
```

```
SF_=TAB_+TAC_+G_;
SFx=SF_(1);
SFy=SF_(2);

sol=solve(SFx, SFy,'TAB, TAC');

Tab=eval(sol.TAB);
Tac=eval(sol.TAC);
Tabn=subs(Tab, list, listn);
Tacn=subs(Tac, list, listn);

fprintf('TAB = \n');pretty(simple(Tab));
fprintf('= %g (N) \n', Tabn);

fprintf('\n')
fprintf('TAC = \n');pretty(simple(Tac));
fprintf('= %g (N) \n', Tacn);

fprintf('\n')
fprintf('Method II \n')

rB_=[-a1, 0, 0];
rC_=[ a2, 0, 0];
rA_=[0,-a3,0];

% SMB_ = rBA_ x (TAC_+G_) = 0_
fprintf('sum M about B = rBA_ x (TAC_+G_) = 0_ \n')

SMB_=cross(rA_-rB_,TAC_+G_);

TACs=solve(SMB_(3),'TAC');

fprintf('TAC = \n')
pretty(simple(TACs))
fprintf('\n')
TACn=subs(TACs, list, listn);
fprintf('TAC = %g (N) \n', TACn)

fprintf('\n')
% SMC = rCA_ x (TAB_+G_) = 0_
fprintf('sum M about C = rCA_ x (TAB_+G_) = 0_ \n')

SMC_=cross(rA_-rC_,TAB_+G_);
```

```
TABs=solve(SMC_(3),'TAB');

fprintf('TAB = \n')
pretty(simple(TABs))
fprintf('\n')
TABn=subs(TABs, list, listn);
fprintf('TAB = %g (N) \n', TABn)

% end of program
```

4.9.2 Program 4.3

```
% example 4.3
% equilibrium of a particle on a surface

clear all; clc; close

syms x y z a b c d h k m g lambda
f = a*x+b*y+c*z+d; % surface equation
rP_ = [x y z]; % position vector of the particle P
rA_ = [0 0 h]; % position vector of fixed point A
F_ = k*(rA_-rP_); % attraction force on the particle
G_ = [0 0 -m*g]; % gravity force on particle P
% net force on the particle: F_+G_
% reaction force of the surface on the particle
gradf_ = [diff(f,x), diff(f,y), diff(f,z)];
N_ = lambda*gradf_;
% equilibrium equations
SF_ = F_+G_+N_;
sol=solve...
(SF_(1),SF_(2),SF_(3),f,'x,y,z,lambda');
xe = eval(sol.x);
ye = eval(sol.y);
ze = eval(sol.z);
lambdae = eval(sol.lambda);
Ne_ = lambdae*gradf_;
Ne = sqrt(simple(Ne_*Ne_.'));

fprintf('equilibrium equations: \n')
fprintf('\n')
fprintf('%s = 0 \n',char(SF_(1)))
fprintf('%s = 0 \n',char(SF_(2)))
fprintf('%s = 0 \n',char(SF_(3)))
```

```
fprintf('%s = 0 \n',char(f))
fprintf('\n')

fprintf('equilibrium position: \n')
fprintf('\n')
fprintf('x = \n')
pretty(simple(xe))
fprintf('\n')
fprintf('y = \n')
pretty(simple(ye))
fprintf('\n')
fprintf('z = \n')
pretty(simplify(ze))
fprintf('\n')

fprintf('normal reaction force N_: \n')
fprintf('\n')
fprintf('lambda = \n')
pretty(simple(lambdae))
fprintf('\n')
fprintf('N_ = \n')
pretty(simple(Ne_))
fprintf('\n')
fprintf('N = |N_| = \n')
pretty(Ne)


% numerical application

lists = {a, b, c, d, h, k, m, g};
% numbers for lists
listn = {1,1,1,1.5,5,10,10,9.8};

xn = subs(xe,lists,listn);
yn = subs(ye,lists,listn);
zn = subs(ze,lists,listn);
ln = subs(lambdae,lists,listn);

rPn_ = [xn yn zn];
rAn_ = subs(rA_,lists,listn);
rPAn_ = rAn_-rPn_;
Gn_ = subs(G_,lists,listn);
Fn_ = subs(k*rPAn_,lists,listn);
Nn_ = subs(Ne_,lists,listn);
```

```
quiver3(0,0,0,xn,yn,zn,1,...
    'Color','k','LineWidth',1.5)
hold on
quiver3(xn,yn,zn, rPAn_(1),rPAn_(2),rPAn_(3),1,...
    'Color','k','LineWidth',1.5)
hold on
ff=10;
quiver3(xn,yn,zn,Nn_(1)/ff,Nn_(2)/ff,Nn_(3)/ff,1,...
    'Color','k','LineWidth',1.5)
hold on
quiver3(xn,yn,zn,Gn_(1)/ff,Gn_(2)/ff,Gn_(3)/ff,1,...
    'Color','k','LineWidth',1.5)
hold on

% surface equation
mg = 6;
[X,Y] = meshgrid(-mg:1:mg);
Z = (-a*X - b*Y -d)/c;
Z = subs(Z,lists,listn);
surf(X,Y,Z)
mesh(X,Y,Z,'EdgeColor','black')

xlabel('x(m)'), ylabel('y(m)'), zlabel('z(m)')
sf=12;
axis([-sf sf -sf sf -sf sf])

text(0,0,0,' O','HorizontalAlignment','right')
text(xn,yn,zn,' P','HorizontalAlignment','right')
text(rAn_(1),rAn_(2),rAn_(3),' A',...
    'HorizontalAlignment','right')
text(xn+Nn_(1)/ff,yn+Nn_(2)/ff,zn+Nn_(3)/ff,...
 ' N','fontsize',14,'fontweight','b')
text(xn+Gn_(1)/ff,yn+Gn_(2)/ff,zn+Gn_(3)/ff,...
 ' G','fontsize',14,'fontweight','b')

% cartesian axes
quiver3(0,0,0,sf,0,0,1,'Color','k','LineWidth',1)
quiver3(0,0,0,0,sf,0,1,'Color','k','LineWidth',1)
quiver3(0,0,0,0,0,sf,1,'Color','k','LineWidth',1)

text(sf,0,0,'  x','fontsize',12,'fontweight','b')
text(0,sf,0,'  y','fontsize',12,'fontweight','b')
text(0,0,sf,'  z','fontsize',12,'fontweight','b')

AZ = 35;
```

```
EL = 14;
view(AZ,EL)

% end of program
```

4.9.3 Program 4.4

```
% example 4.4
% equilibrium of a particle on a curve

clear all; clc; close

syms x y z a k m g lambda
% parabola equation: y+a*x^2=0 & z=0
f = y+a*x^2; % a>0
% particle position vector
rP_ = [x y 0];
% attraction force on particle
% proportional to the y-axis distance
% F_ = -k*[rP_(1),0,0]
F_ = -k*[rP_(1) 0 0];
% gravity force on particle P
G_ = [0 -m*g 0];
% reaction force of the surface on the particle
gradf_ = [diff(f,x), diff(f,y), diff(f,z)];
N_ = lambda*gradf_;
% equilibrium equations
SF_ = F_ + G_ + N_;
fprintf('equilibrium equations: \n')
fprintf('\n')
fprintf('%s = 0 (1)\n',char(SF_(1)))
fprintf('%s = 0 (2)\n',char(SF_(2)))
fprintf('%s = 0 (3)\n',char(f))
fprintf('\n')
sol=solve...
(SF_(1),SF_(2),f,'x,y,lambda');
xe = eval(sol.x);
ye = eval(sol.y);
lambde = eval(sol.lambda);

list  ={x,y,lambda};
liste ={xe,ye,lambde};
Ne_ = subs(N_,list,liste);
```

```
Ne = sqrt(simple(Ne_*Ne_.'));

fprintf('I. one equilibrium position is \n')
fprintf('x = %d \n',xe)
fprintf('y = %d \n',ye)
fprintf('lambda = %s \n',char(lambde))
fprintf('curve reaction force N_ \n')
fprintf('Nx = %s \n',char(Ne_(1)))
fprintf('Ny = %s \n',char(Ne_(2)))
fprintf(' \n')
% numerical application
lists = {a, k, m, g};
% numbers for lists
listn = {1, 10, 10, 9.8};

xn = subs(xe,lists,listn);
yn = subs(ye,lists,listn);
ln = subs(lambde,lists,listn);
Gn_ = subs(G_,lists,listn);
rPn_ = [xn yn 0];
Nn_ = subs(Ne_,lists,listn);
fprintf('numerical application \n')
fprintf('G_ = [%g,%6.3f,%g] (N)\n',Gn_)
fprintf('N_ = [%g,%6.3f,%g] (N)\n',Nn_)
fprintf(' \n')

fprintf('II. from Eq.(1)=> \n')
fprintf...
('equilibrim condition for any point on the curve\n')
sole=solve...
(SF_(1),SF_(2),'k,lambda');
ke = eval(sole.k);
lame = eval(sole.lambda);
fprintf('k = %s \n',char(ke))

% plot the curve
x = -2:1/10:2;
y = -x.^2;
plot(x,y,'-','LineWidth',2)
hold on
ff = 50;
quiver(xn,yn,Gn_(1)/ff,Gn_(2)/ff,1,...
    'Color','k','LineWidth',1.5)
quiver(xn,yn,Nn_(1)/ff,Nn_(2)/ff,1,...
    'Color','r','LineWidth',1.5)
```

```
grid on
xlabel('x(m)'), ylabel('y(m)')
text(xn+Nn_(1)/ff,yn+Nn_(2)/ff,...
 ' N','fontsize',14,'fontweight','b')
text(xn+Gn_(1)/ff,yn+Gn_(2)/ff,...
 ' G','fontsize',14,'fontweight','b')

% end of program
```

4.9.4 Program 4.5

```
% example 4.5
% equilibrium of a particle on a circle

clear all; clc; close

syms x y z R k m g lambda
% circle equation
f = x^2+y^2-R^2;
% position vector of the particle P
rP_ = [x y 0];
% reaction force on the particle
% F_ = k*[x 0 0]
F_ = k*[x 0 0];
% gravity force on particle P
G_ = [0 -m*g 0];
% external force on the particle: F_+G_
% reaction force of the surface on the particle
gradf_ = [diff(f,x), diff(f,y), diff(f,z)];
N_ = lambda*gradf_;
% equilibrium equations
SF_ = F_ + G_ + N_;
fprintf('equilibrium equations: \n')
fprintf('\n')
fprintf('%s = 0 \n',char(SF_(1)))
fprintf('%s = 0 \n',char(SF_(2)))
fprintf('%s = 0 \n',char(f))
fprintf('\n')

sol=solve...
(SF_(1),SF_(2),f,'x,y,lambda');
xe = eval(sol.x);
ye = eval(sol.y);
```

```
lambde = eval(sol.lambda);
list  ={x,y,lambda};

fprintf('equilibrium position P1: \n\n')
fprintf('x = %s \n',char(xe(1)))
fprintf('y = %s \n',char(ye(1)))
fprintf('lambda = %s \n',char(lambde(1)))
list1 ={xe(1),ye(1),lambde(1)};
N1_ = subs(N_,list,list1);
fprintf('Nx = %s \n',char(N1_(1)))
fprintf('Ny = %s \n',char(N1_(2)))
fprintf('Nz = %s \n',char(N1_(3)))
fprintf('\n')

fprintf('equilibrium position P2: \n\n')
fprintf('x = %s \n',char(xe(2)))
fprintf('y = %s \n',char(ye(2)))
fprintf('lambda = %s \n',char(lambde(2)))
list2 ={xe(2),ye(2),lambde(2)};
N2_ = subs(N_,list,list2);
fprintf('Nx = %s \n',char(N2_(1)))
fprintf('Ny = %s \n',char(N2_(2)))
fprintf('Nz = %s \n',char(N2_(3)))
fprintf('\n')

fprintf('equilibrium position P3: \n\n')
fprintf('x = %s \n',char(xe(3)))
fprintf('y = %s \n',char(ye(3)))
fprintf('lambda = %s \n',char(lambde(3)))
list3 ={xe(3),ye(3),lambde(3)};
N3_ = subs(N_,list,list3);
fprintf('Nx = %s \n',char(N3_(1)))
fprintf('Ny = %s \n',char(N3_(2)))
fprintf('Nz = %s \n',char(N3_(3)))
fprintf('\n')

fprintf('equilibrium position P4: \n\n')
fprintf('x = %s \n',char(xe(4)))
fprintf('y = %s \n',char(ye(4)))
fprintf('lambda = %s \n',char(lambde(4)))
list4 ={xe(4),ye(4),lambde(4)};
N4_ = subs(N_,list,list4);
fprintf('Nx = %s \n',char(N4_(1)))
fprintf('Ny = %s \n',char(N4_(2)))
fprintf('Nz = %s \n',char(N4_(3)))
```

```
fprintf('\n')

% numerical application
lists = {R, k, m, g};
% numbers for lists
listn = {1, 20, 1, 9.8};

x1 = subs(xe(1),lists,listn);
y1 = subs(ye(1),lists,listn);
N1n_= subs(N1_,lists,listn);

x2 = subs(xe(2),lists,listn);
y2 = subs(ye(2),lists,listn);
N2n_= subs(N2_,lists,listn);

x3 = subs(xe(3),lists,listn);
y3 = subs(ye(3),lists,listn);
N3n_= subs(N3_,lists,listn);

x4 = subs(xe(4),lists,listn);
y4 = subs(ye(4),lists,listn);
N4n_= subs(N4_,lists,listn);

xC = 0; yC = 0; R = 1;
phi=0:0.01:2*pi;
xp=R*cos(phi);
yp=R*sin(phi);
plot(xC+xp,yC+yp,'--','LineWidth',1)
hold on

plot(x1,y1,'r.')
plot(x2,y2,'r.')
plot(x3,y3,'r.')
plot(x4,y4,'r.')

ff = 25;
quiver...
(x1,y1,N1n_(1)/ff,N1n_(2)/ff,1,...
'Color','k','LineWidth',1.5)
quiver...
(x2,y2,N2n_(1)/ff,N2n_(2)/ff,1,...
'Color','k','LineWidth',1.5)
quiver...
(x3,y3,N3n_(1)/ff,N3n_(2)/ff,1,...
'Color','k','LineWidth',1.5)
```

```
quiver...
(x4,y4,N4n_(1)/ff,N4n_(2)/ff,1,...
'Color','k','LineWidth',1.5)

text...
(x1+N1n_(1)/ff,y1+N1n_(2)/ff,...
' N1','fontsize',14,'fontweight','b')
text...
(x2+N2n_(1)/ff,y2+N2n_(2)/ff,...
' N2','fontsize',14,'fontweight','b')
text...
(x3+N3n_(1)/ff,y3+N3n_(2)/ff,...
' N3','fontsize',14,'fontweight','b')
text...
(x4+N4n_(1)/ff,y4+N4n_(2)/ff,...
' N4','fontsize',14,'fontweight','b')

xlabel('x(m)'), ylabel('y(m)')
sf=1.5;
axis([-sf sf -sf sf])
grid on

text(x1,y1,' P1','fontsize',14,'fontweight','b')
text(x2,y2,' P2','fontsize',14,'fontweight','b')
text(x3,y3,' P3','fontsize',14,'fontweight','b')
text(x4,y4,' P4','fontsize',14,'fontweight','b')

% end of program
```

4.9.5 Program 4.6

```
% example 4.6
% method of joints:
% support reactions are not required
% for determining the member forces

clear all; clc; close all

Fn =100.; % daN
syms F
theta=atan(1/2);
alpha=atan(1);
```

```
% joint D
% sumFDy = F4D*sin(alpha)- 2*F = 0
F4D = 2*F/sin(alpha);
% sumFDx = F4D*cos(alpha)- F3D = 0
F3D = F4D*cos(alpha);

fprintf('F4D = %s ',char(F4D))
fprintf(' = %6.3f (daN) \n',subs(F4D,F,Fn))
fprintf('F3D = %s ',char(F3D))
fprintf(' = %6.3f (daN) \n',subs(F3D,F,Fn))
fprintf('\n')

% F4D = 2*2^(1/2)*F  = 282.843 (daN) =>
% 4=AD (Compression)

% F3D = 2*F  = 200.000 (daN) =>
% 3=DC (Compression)

syms F2C F5C
% joint C
F3C = F3D;
% sumFCx = F3C-F5C*cos(theta)+F2C*cos(alpha)-F=0
sumFCx = F3C-F5C*cos(theta)+F2C*cos(alpha)-F;
% sumFCy = -F5C*sin(theta)+F2C*sin(alpha)=0
sumFCy = -F5C*sin(theta)+F2C*sin(alpha);
solC=solve(sumFCx,sumFCy,'F2C','F5C');
F2Cs = solC.F2C;
F5Cs = solC.F5C;
fprintf('F2C = %s ',char(F2Cs))
fprintf(' = %6.3f (daN) \n',subs(F2Cs,F,Fn))
fprintf('F5C = %s ',char(F5Cs))
fprintf(' = %6.3f (daN) \n',subs(F5Cs,F,Fn))
fprintf('\n')

% F2C = 2^(1/2)*F  = 141.421 (daN) =>
% 2=BC (Compression)

% F5C = 5^(1/2)*F  = 223.607 (daN) =>
% 5=AC (Tension)

% joint B
F2B = F2Cs;
% sumFBx = F1B-F2B*cos(alpha)=0
F1B=F2B*cos(alpha);
% sumFBy = -F2B*sin(alpha)+F0By=0
```

```
F0By = F2B*sin(alpha);
fprintf('F1B = %s ',char(F1B))
fprintf(' = %6.3f (daN) \n',subs(F1B,F,Fn))

% F1B = F  = 100.000 (daN) =>
% 1=AB (Tension)

% end of program
```

4.9.6 Program 4.7

```
% example 4.7

clear all; clc; close all

syms xO yO xA yA xB yB xC yC xD yD
rO_=[xO,yO,0];
rA_=[xA-xO,yA-yO,0];
rB_=[xB-xO,yB-yO,0];
rC_=[xC-xO,yC-yO,0];
rD_=[xD-xO,yC-yO,0];

syms FA alpha
FAx=FA*cos(alpha);
FAy=FA*sin(alpha);
FA_=[FAx,FAy,0];
fprintf('FA_=[%s,%s] \n\n',...
    char(FAx),char(FAy))

syms FB beta
FBx=-FB*cos(beta);
FBy=FB*sin(beta);
FB_=[FBx,FBy,0];
fprintf('FB_=[%s,%s]\n\n',...
    char(FBx),char(FBy))

syms FD gamma
FDx=-FD*cos(gamma);
FDy=-FD*sin(gamma);
FD_=[FDx,FDy,0];
fprintf('FD_=[%s,%s]\n\n',...
    char(FDx),char(FDy))
```

```
syms m g
G_=[0,-m*g,0];
fprintf('G_ =[%s,%s] \n\n',...
    char(G_(1)),char(G_(2)))

syms FOx FOy
FO_=[FOx,FOy,0];

sumF_ = FO_+G_+FA_+FB_+FD_;
SFx = sumF_(1);
SFy = sumF_(2);

fprintf('sum of forces on x:\n\n')
fprintf('sumFx = %s \n\n', char(SFx))
fprintf('sum of forces on y:\n\n')
fprintf('sumFy = %s \n\n', char(SFy))

FOx=solve(SFx, FOx);
FOy=solve(SFy, FOy);
fprintf('reaction FOx is \n\n')
fprintf('FOx = %s \n\n', char(FOx))
fprintf('reaction FOy is \n\n')
fprintf('FOy = %s \n\n', char(FOy))

syms MOz MCz
MO_=[0,0,MOz];
MC_=[0,0,MCz];

sumMO_=MO_+MC_...
+cross(rA_,FA_)+cross(rB_,FB_)+...
cross(0.5*rD_,G_)+cross(rD_,FD_);
SMO=sumMO_(3);
fprintf('sum of moments about O:\n\n')
fprintf('SMO = %s \n\n', char(SMO));

MOz=solve(SMO, MOz);
fprintf('The moment MO is \n\n')
fprintf('MOz = %s \n\n', char(MOz));

lists = {xO,yO,xA,yA,xB,yB,xC,yC,xD,yD,...
         FA,FB,FD,alpha,beta,gamma,m,g,MCz};
listt = {0,0,1,0,3,0,5,0,8,0,...
3000,3500,4000,45*pi/180,30*pi/180,...
30*pi/180,250,9.81,10*10^3};
```

```
FOx=subs(FOx,lists,listt);
FOy=subs(FOy,lists,listt);
MOz=subs(MOz ,lists,listt);
fprintf(' FOx = %g (kN)\n\n', FOx/1000);
fprintf(' FOy = %g (kN)\n\n', FOy/1000);
fprintf(' MOz = %g (kN m)\n\n', MOz/1000);

a = 7000;
axis([-a/2 a -a a])
hold on
axis equal

ax=0; ay=8000;
bx=0; by=0;
A = [ax ay];
B = [bx by];
line(A,B,'LineStyle','-',...
    'Color','r','LineWidth',6)

ax=11000;
ay=6000;
quiver(0,0,ax,0,...
'Color','b','LineWidth',1.0);
text(ax,0,' x',...
'fontsize',12,'fontweight','b');
quiver(0,0,0,ay,...
'Color','b','LineWidth',1.0);
text(0,ay,' y',...
'fontsize',12,'fontweight','b');

x_O=0;y_O=0;
yy=400;
text(x_O-yy,y_O-yy,' O',...
'fontsize',12,'fontweight','b');
x_A=1;y_A=0;
text(x_A*1000-yy,y_A-yy,' A',...
'fontsize',12,'fontweight','b');
x_B=3;y_B=0;
text(x_B*1000-yy,y_B-yy,' B',...
'fontsize',12,'fontweight','b');
x_C=5;y_C=0;
text(x_C*1000-yy,y_C-yy,' C',...
'fontsize',12,'fontweight','b');
x_D=8;y_D=0;
text(x_D*1000-yy,y_D-yy,' D',...
```

```
'fontsize',12,'fontweight','b');

x_A=1;
FA_=subs(FA_,lists,listt);
quiver(x_A*1000,0,FA_(1),FA_(2),...
'Color','k','LineWidth',1.25)
text(FA_(2)+200,FA_(1)+200,' F_A',...
'fontsize',12,'fontweight','b');

x_B=3;
FB_=subs(FB_,lists,listt);
quiver(x_B*1000,0,FB_(1),FB_(2),...
'Color','k','LineWidth',1.25);
text(x_B*1000+FB_(1),FB_(2)+200,' F_B',...
'fontsize',12,'fontweight','b');

x_D=8;
FD_=subs(FD_,lists,listt);
quiver(x_D*1000,0,FD_(1),FD_(2),...
'Color','k','LineWidth',1.25);
text(x_D*1000+FD_(1),FD_(2)-500,' F_D',...
'fontsize',12,'fontweight','b');

x_G=x_D/2;
G_=subs(G_,lists,listt);
quiver(x_G*1000,0,0,G_(2),...
'Color','k','LineWidth',1.25)
text(x_G*1000-800,G_(2)+1,' G ',...
'fontsize',12,'fontweight','b')

XMO1 = 0;
YMO1 = 0;
x_C = 5;
XMB1 = x_C*1000;
YMB1 = 0;
scatter(XMO1,YMO1,80,2)
scatter(XMO1,YMO1,30,50,'filled')
scatter(XMB1,YMB1,80,2)
scatter(XMB1,YMB1,30,50,'filled')

% end of program
```

4.9.7 Program 4.8

```
% example 4.8
% equilibrium 3D

clear all; clc; close all

h=0.6; % m
b=0.6; % m
a=0.5; % m
m = 100.;  % kg
g = 9.81;  % m/s^2
Me = 120.; % N m

xA=0; yA=0; zA=0;    % m
xB=0; yB=0; zB=h;    % m
xD=0; yD=b; zD=0;    % m
xE=0; yE=b; zE=h;    % m
xC=0; yC=b/2;zC=h/2; % m
xP=a; yP=0; zP=0;    % m

rA_=[xA yA zA];
rB_=[xB yB zB];
rD_=[xD yD zD];
rE_=[xE yE zE];
rC_=[xC yC zC];
rP_=[xP yP zP];

G_ = [0 0 -m*g];
syms FAx FAy FAz
FA_ = [FAx,FAy,FAz];
syms FBx FBy
FB_ = [FBx,FBy,0];
u_=(rP_-rD_)/norm(rP_-rD_);
syms T
T_ = T*u_;

SF_ = FA_ + FB_ + G_ + T_;

Me_=[0,0,Me];
SMA_ = ...
cross(rC_,G_)+cross(rD_,T_)+cross(rB_,FB_)+Me_;

Fx=vpa(SF_(1),3);
```

```
fprintf('Fx : %s = 0 \n',char(Fx))
Fy=vpa(SF_(2),3);
fprintf('Fy : %s = 0 \n',char(Fy))
Fz=vpa(SF_(3),3);
fprintf('Fz : %s = 0 \n',char(Fz))

Mx=vpa(SMA_(1),3);
fprintf('Mx : %s = 0 \n',char(Mx))
My=vpa(SMA_(2),3);
fprintf('My : %s = 0 \n',char(My))
Mz=vpa(SMA_(3),3);
fprintf('Mz : %s = 0 \n',char(Mz))

sol=solve(...
    SF_(1) , SF_(2), SF_(3),...
    SMA_(1),SMA_(2),SMA_(3));
FAxs=eval(sol.FAx);
FAys=eval(sol.FAy);
FAzs=eval(sol.FAz);
FBxs=eval(sol.FBx);
FBys=eval(sol.FBy);
Ts=eval(sol.T);

fprintf('=>\n')
fprintf('FAx= %6.3f (N)\n',FAxs)
fprintf('FAy= %6.3f (N)\n',FAys)
fprintf('FAz= %6.3f (N)\n',FAzs)
fprintf('FBx= %6.3f (N)\n',FBxs)
fprintf('FBy= %6.3f (N)\n',FBys)
fprintf('T= %6.3f (N)\n',Ts)

FAs_ = [FAxs FAys FAzs];
FBs_ = [FBxs FBys 0];
Ts_ = Ts*u_;

as=1;
hold on
axis([-as as -as as -as as])
grid on
view(136,18);

quiver3(0,0,0,as+0.1,0,0,...
'Color','b','LineWidth',1.0);
text(as+0.1,0,0,' x',...
'fontsize',12,'fontweight','b');
```

```
quiver3(0,0,0,0,as+0.1,0,...
'Color','b','LineWidth',1.0);
text(0,as+0.1,0,' y',...
'fontsize',12,'fontweight','b');
quiver3(0,0,0,0,0,as+0.1,...
'Color','b','LineWidth',1.0);
text(0,0,as+0.1,' z',...
'fontsize',12,'fontweight','b');

% scatter3(x,y,z,S,C) displays
% colored circles at (x,y,z)
% S area of the marker
% C color of the marker

scatter3(xA,yA,zA,60,'k','filled')
scatter3(xB,yB,zB,60,'k','filled')
scatter3(xD,yD,zD,30,'b','filled')
scatter3(xE,yE,zE,30,'b','filled')
scatter3(xC,yC,zC,30,'r','filled')
scatter3(xP,yP,zP,30,'b','filled')

text(xA,yA,zA,' A','fontsize',12);
text(xB,yB,zB+0.05,' B','fontsize',12);
text(xD,yD,zD,' D','fontsize',12);
text(xE,yE,zE,' E','fontsize',12);
text(xC,yC,zC,' C','fontsize',12);
text(xP,yP,zP+0.05,' P','fontsize',12);

%input the vertices
vert = ...
[xA yA zA; xB yB zB; xE yE zE; xD yD zD];
%input the faces
fac = [1 2 3 4];
%draw using patch function
prism=patch('Faces',fac,...
'Vertices',vert,'FaceColor','g');

 line([xP xD],[yP yD],[zP zD],...
'LineStyle','--','Color','k','LineWidth',1)

fs = 1000;
quiver3(xD,yD,zD,Ts_(1)/fs,Ts_(2)/fs,Ts_(3)/fs,...
'Color','k','LineWidth',2);
text...
(xD+Ts_(1)/fs,yD+Ts_(2)/fs,zD+0.03+Ts_(3)/fs,'T');
```

```
quiver3(xA,yA,zA,FAs_(1)/fs,FAs_(2)/fs,FAs_(3)/fs,...
'Color','k','LineWidth',2);
text...
(xA+FAs_(1)/fs,yA+FAs_(2)/fs,...
zA-0.1+FAs_(3)/fs,'F_A');

quiver3(xB,yB,zB,FBs_(1)/fs,FBs_(2)/fs,FBs_(3)/fs,...
'Color','k','LineWidth',2);
text...
(xB+FBs_(1)/fs,yB+FBs_(2)/fs,...
zB+0.05+FBs_(3)/fs,'F_B');

quiver3(xC,yC,zC,G_(1)/fs,G_(2)/fs,G_(3)/fs,...
'Color','k','LineWidth',2);
text...
(xC+G_(1)/fs,yC+G_(2)/fs,zC+G_(3)/fs,'G');

quiver3(xB,yB,zB,Me_(1)/fs,Me_(2)/fs,Me_(3)/fs,...
'Color','k','LineWidth',2);
text...
(xB+Me_(1)/fs,yB+0.05+Me_(2)/fs,...
 zB+0.05+Me_(3)/fs,'M_e');

light('Position',[1 3 2]);
alpha(prism,0.3);
%alpha sets one of three transparency properties
%depending on the specified arguments
xlabel('x(m)'); ylabel('y(m)');zlabel('z(m)');

% end of program
```

References

1. P. Appell, *Traité de mécanique rationnelle* (Gauthier-Villars, Paris, 1955)
2. M. Atanasiu, *Mechanics* (EDP, Bucharest, 1973)
3. H. Baruh, *Analytical Dynamics* (WCB/McGraw-Hill, Boston, 1999)
4. G. Baumann, *Mathematica for Theoretical Physics: Classical Mechanics and Nonlinear Dynamics* (Springer, New York, 2005)
5. G. Baumann, *Mathematica for Theoretical Physics: Electrodynamics, Quantum Mechanics, General Relativity and Fractals* (Springer, New York, 2005)
6. F.P. Beer, E.R. Johnston, *Vector Mechanics for Engineers: Statics and Dynamics 5/e* (McGraw-Hill, New York, 1988)
7. F.P. Beer, E.R. Johnston, D.F. Mazurek, *Vector Mechanics for Engineers: Statics, 10/e*(McGraw-Hill, New York, 2012)

8. A.M. Bedford, W. Fowler, K.M. Liechti, *Statics and Mechanics of Materials* (Prentice Hall, Upper Saddle River, 2002)
9. A.M. Bedford, W. Fowler, *Engineering Mechanics: Statics, 5/e* (Prentice Hall, Upper Saddle River, 2007)
10. A.P. Boresi, R.J. Schmidt, *Engineering Mechanics: Statics* (PWS Publishing Company, Boston, 2000)
11. M.I. Buculei, *Mechanics* (University of Craiova Press, Craiova, 1974)
12. M.I. Buculei, D. Bagnaru, G. Nanu, D.B. Marghitu, *Analysis of Mechanisms with Bars* (Scrisul romanesc, Craiova, 1986)
13. I. Stroe et al., *Analytical Mechanics Problems* (University Politehnica of Bucharest, Romania, 1997)
14. V. Ceausu, N. Enescu, F. Ceausu, *Mechanics Problems* (Printech, Bucharest, 1999)
15. S.J. Chapman, *MATLAB Programming for Engineers* (Thomson Learning, Pacific Grove, CA, 2002)
16. S. Attaway, *MATLAB: A Practical Introduction to Programming and Problem Solving* (Butterworth-Heinemann, Elsevier, Amsterdam, 2012)
17. D.M. Etter, D.C. Kuncicky, *Introduction to MATLAB for Engineers and Scientists* (Prentice Hall, Upper Saddle River, 1996)
18. C. Iacob, *Theoretical Mechanics* (EDP, Bucharest, 1980)
19. J.H. Ginsberg, *Advanced Engineering Dynamics* (Cambridge University Press, Cambridge, 1995)
20. D.T. Greenwood, *Principles of Dynamics* (Prentice-Hall, Englewood Cliffs, 1998)
21. L.E. Goodman, W.H. Warner, *Statics* (Dover Publications, New York, 2001)
22. R.C. Hibbeler, *Engineering Mechanics: Statics and Dynamics 13/e* (Prentice-Hall, Upper Saddle River, 2013)
23. T.R. Kane, *Analytical Elements of Mechanics*, vol. 1 (Academic Press, New York, 1959)
24. T.R. Kane, *Analytical Elements of Mechanics*, vol. 2 (Academic Press, New York, 1961)
25. T.R. Kane, P.W. Likins, D.A. Levinson, *Spacecraft Dynamics* (McGraw-Hill, New York, 1983)
26. T.R. Kane, D.A. Levinson, *Dynamics* (McGraw-Hill, New York, 1985)
27. R. Maeder, *Programming in Mathematica* (Addison-Wesley, Redwood City, 1990)
28. N.H. Madsen, *Statics and Dynamics*, class notes, http://www.eng.auburn.edu/users/nmadsen/
29. D.B. Marghitu, *Mechanical Engineer's Handbook* (Academic Press, San Diego, 2001)
30. D.B. Marghitu, M.J. Crocker, *Analytical Elements of Mechanisms* (Cambridge University Press, Cambridge, 2001)
31. D.B. Marghitu, *Kinematic Chains and Machine Component Design* (Elsevier, Amsterdam, 2005)
32. D.B. Marghitu, *Mechanisms and Robots Analysis with MATLAB* (Springer, New York, 2009)
33. D.B. Marghitu, M. Dupac, *Advanced Dynamics: Analytical and Numerical Calculations with MATLAB* (Springer, New York, 2012)
34. D.B. Marghitu, *Statics and Dynamics*, class notes, http://www.eng.auburn.edu/users/marghitu/
35. D.J. McGill, W.W. King, *Engineering Mechanics: Statics and an Introduction to Dynamics* (PWS Publishing Company, Boston, 1995)
36. J.L. Meriam, L.G. Kraige, *Engineering Mechanics: Statics, 7/e* (Wiley, New York, 2011)
37. R.L. Mott, *Machine Elements in Mechanical Design* (Prentice Hall, Upper Saddle River, 1999)
38. R.L. Norton, *Machine Design* (Prentice-Hall, Upper Saddle River, 1996)
39. L.A. Pars, *A Treatise on Analytical Dynamics* (Wiley, New York, 1965)
40. M. Plesha, G. Gray, F. Costanzo, *Engineering Mechanics: Statics, 2/e* (McGraw-Hill, New York, 2012)
41. M. Radoi, E. Deciu, *Mechanics* (EDP, Bucharest, 1981)
42. W.F. Riley, L.D. Sturges, *Engineering Mechanics: Statics, 2/e* (Wiley, New York, 1995)
43. A. Ruina, R. Pratap, *Introduction to Statics and Dynamics* (Oxford University Press, Oxford, 2002)
44. A. Ripianu, P. Popescu, B. Balan, *Technical Mechanics* (EDP, Bucharest, 1979)
45. I.H. Shames, *Engineering Mechanics Statics, 4/e* (Prentice Hall, New Jersey, 1996)

46. S.D. Sheppard, B.H. Tongue, *Statics: Analysis and Design of Systems in Equilibrium* (Wiley, New York, 2005)
47. D. Smith, *Engineering Computation with MATLAB* (Pearson Education, Upper Saddle River, 2008)
48. R.W. Soutas-Little, D.J. Inman, *Engineering Mechanics: Statics and Dynamics* (Prentice-Hall, Upper Saddle River, 1999)
49. R.W. Soutas-Little, D.J. Inman, D. Balint, *Engineering Mechanics: Statics* (Cengage Learning, Independence, KY, 2007)
50. S. Staicu, *Theoretical Mechanics* (EDP, Bucharest, 1998)
51. A. Stan, M. Grumazescu, *Mechanics Problems* (EDP, Bucharest, 1973)
52. J. Sticklen, M.T. Eskil, *An Introduction to Technical Problem Solving with MATLAB* (Great Lakes Press, Wildwood, 2006)
53. A. Stoenescu, G. Silas, *Theoretical Mechanics* (ET, Bucharest, 1957)
54. J.H. Jackson, H.G. Wirtz, *Schaum's Outline of Theory and Problems of Statics and Strength of Materials* (McGraw-Hill, New York, 1983)
55. The MathWorks: https://www.mathworks.com/
56. Statics eBook : https://ecourses.ou.edu/
57. R. Voinea, D. Voiculescu, V. Ceausu, *Mechanics* (EDP, Bucharest, 1983)
58. V. Valcovici, S. Balan, R. Voinea, *Theoretical Mechanics* (ET, Bucharest, 1959)
59. K.J. Waldron, G.L. Kinzel, *Kinematics, Dynamics, and Design of Machinery* (Wiley, New York, 1999)
60. H.B. Wilson, L.H. Turcotte, and D. Halpern, *Advanced Mathematics and Mechanics Applications Using MATLAB* (Chapman & Hall/CRC, Boca Raton, FL, 2003)
61. J.H. Williams Jr, *Fundamentals of Applied Dynamics* (Wiley, New York, 1996)
62. S. Wolfram, *Mathematica* (Wolfram Media/Cambridge University Press, Cambridge, 1999)

Chapter 5
Friction

5.1 Introduction

Friction (from Latin fricare, to rub), is the term given to the resistance caused by the moving of the surfaces of bodies over each other. The resistance is due to the roughness of the surfaces. The first experiments on friction were made by Guillaume Amontons (1699) when he published his rediscovery of the laws of friction first presented by Leonardo da Vinci. Leonardo da Vinci (1452–1519) studied screws, gears, mechanisms, wear, bearings, friction, and lubrication. At Rochefort in 1781, Charles-Augustin de Coulomb verified the laws friction. The laws of dry friction are: 1. friction is directly proportional to the normal force between the surfaces of contact (Amontons 1st Law); 2. friction is independent of the apparent area of contact (Amontons 2nd Law); 3. friction is independent of the velocity with which the surfaces slide one on the other (Coulomb's Law). Arthur Jules Morin confirmed and extended Coulomb's work on friction (1830–1834). He build an experimental apparatus under the supervision of Jean-Victor Poncelet. He developed an apparatus to study the laws of falling bodies presented an accurate experimental proof of Galileo's result that distances travelled by a falling body increase as the square of the time.

If a body rests on an incline plane, the friction force exerted on it by the surface prevents it from sliding down the incline. The question is, what is the steepest incline on which the body can rest?

A body is placed on a horizontal surface. The body is pushed with a small horizontal force F. If the force F is sufficiently small, the body does not move.

Figure 5.1 shows the free-body diagram of the body, where the force W is the weight force of the body, and N is the normal force exerted by the surface on the body. The force F is the horizontal force, and F_f is the friction force exerted by the surface. Friction force arises in part from the interactions of the roughness, or asperities, of the contacting surfaces. The body is in equilibrium and $F_f = F$.

The force F is slowly increased. As long as the body remains in equilibrium, the friction force F_f must increase correspondingly, since it equals the force F. The body slips on the surface. The friction force, after reaching the maximum value, cannot

D. B. Marghitu et al., *Statics with MATLAB®*,
DOI: 10.1007/978-1-4471-5110-4_5, © Springer-Verlag London 2013

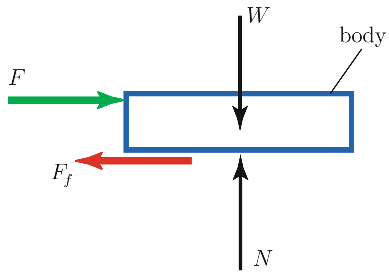


Fig. 5.1 Free-body diagram of the body

maintain the body in equilibrium. The force applied to keep the body moving on the surface is smaller than the force required to cause it to slip. Why more force is required to start the body sliding on a surface than to keep it sliding is explained in part by the necessity to break the asperities of the contacting surfaces before sliding can begin.

The theory of dry friction, or *Coulomb friction*, predicts:

- the maximum friction forces that can be exerted by dry, contacting surfaces that are stationary relative to each other;
- the friction forces exerted by the surfaces when they are in relative motion, or sliding.

5.2 Static Coefficient of Friction

The magnitude of the maximum friction force, F_f, that can be exerted between two plane dry surfaces in contact is

$$F_f = \mu_s N, \tag{5.1}$$

where μ_s is a constant, the *static coefficient of friction*, and N is the normal component of the contact force between the surfaces. The value of the static coefficient of friction, μ_s, depends on:

- the materials of the contacting surfaces;
- the conditions of the contacting surfaces namely smoothness and degree of contamination.

Typical values of μ_s for various materials are shown in Table 5.1.

Equation (5.1) gives the maximum friction force that the two surfaces can exert without causing it to slip. If the static coefficient of friction μ_s between the body and the surface is known, the largest value of F one can apply to the body without causing it to slip is $F = F_f = \mu_s N$. Equation (5.1) determines the magnitude of the maximum friction force but not its direction. The friction force resists the impending motion.

Table 5.1 Typical values of the static coefficient of friction

Materials	μ_s
Metal on metal	0.15–0.20
Metal on wood	0.20–0.60
Metal on masonry	0.30–0.70
Wood on wood	0.25–0.50
Masonry on masonry	0.60–0.70
Rubber on concrete	0.50–0.90

5.3 Kinetic Coefficient of Friction

The magnitude of the friction force between two plane dry contacting surfaces that are in motion relative to each other is

$$F_f = \mu_k N, \tag{5.2}$$

where μ_k is the *kinetic coefficient of friction* and N is the normal force between the surfaces. The value of the kinetic coefficient of friction is generally smaller than the value of the static coefficient of friction, μ_s.

To keep the body in Fig. 5.1 in uniform motion (sliding on the surface) the force exerted must be $F = F_f = \mu_k N$. The friction force resists the relative motion, when two surfaces are sliding relative to each other.

The body *RB* shown in Fig. 5.2a is moving on the fixed surface 0.

The direction of motion of *RB* is the positive axis x. The friction force on the body *RB* acts in the direction opposite to its motion, and the friction force on the fixed surface is in the opposite direction as shown in Fig. 5.2b.

5.4 Angle of Friction

The *angle of friction*, θ, is the angle between the friction force, $F_f = |\mathbf{F}_f|$, and the normal force to the surface $N = |\mathbf{N}|$, as shown in Fig. 5.3.

The magnitudes of the normal force and friction force, and θ are related by

$$F_f = R \sin\theta,$$
$$N = R \cos\theta,$$

where $R = |\mathbf{R}| = |\mathbf{N} + \mathbf{F}_f|$.

The value of the angle of friction when slip is impending is called the *static angle of friction*, θ_s,

$$\tan\theta_s = \mu_s.$$

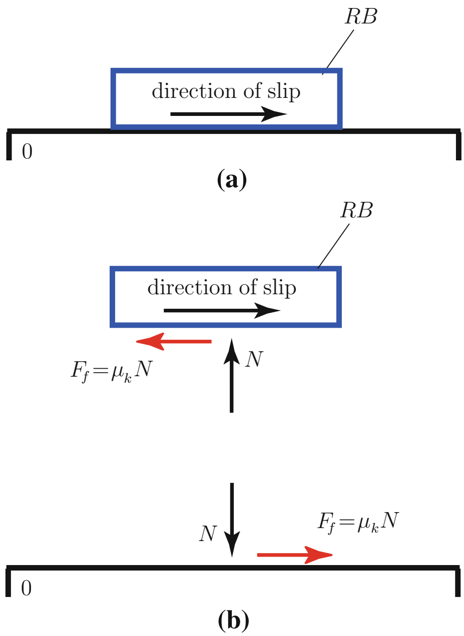


Fig. 5.2 Directions of the friction forces

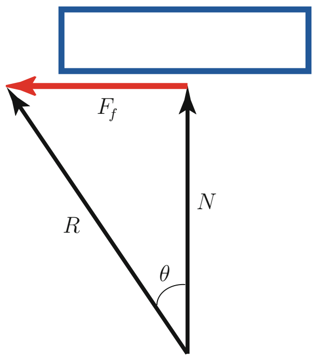


Fig. 5.3 Angle of friction, θ

The value of the angle of friction when the surfaces are sliding relative to each other is called the *kinetic angle of friction*, θ_k,

$$\tan\theta_k = \mu_k.$$

Example 5.1 The prism 1 of mass m_1 makes an angle α with the horizontal and can slide along the horizontal surface as shown Fig. 5.4. The slider 2 of mass m_2 is prevented from horizontal movement and can slide down on the inclined prism 1. The coefficients of static friction between the prism 1 and the slider 2, between the prism 1 and the horizontal surface 0, between the slider 2 and the vertical support 0

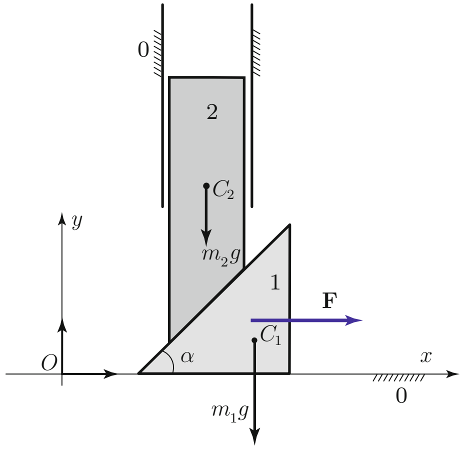


Fig. 5.4 Example 5.1

are equal to μ. The friction is sufficient to prevent the prism from moving without the application of any force.

Determine the greatest value of the horizontal force **F** that acts on the prism 1 without causing the motion of the system.

For the numerical example use $m_1 = 10$ kg, $m_2 = 5$ kg, $\alpha = 10°$, $\mu = 0.2$, and $g = 9.81$ m/s^2.

Solution A reference frame xy having the y-axis directed upward and the x-axis directed along the horizontal surface was considered, as shown Fig. 5.5.

Considering the mechanical system to be the slider-prism combination, the weight of the slider and the external force **F** will act on the prism making the prism to move. In addition to the weights and external force, the mechanical system will also be acted upon reaction and friction forces as shown in Fig. 5.5. The sum of all the forces acting on slider 2 can be expressed as:

$$\sum \mathbf{F}^{(2)} = \mathbf{F}_{02} + \mathbf{F}_{f02} + \mathbf{F}_{12} + \mathbf{F}_{f12} + \mathbf{G}_2,$$

where $\mathbf{F}_{02}$ is the of force of the ground (vertical support) on slider 2, $\mathbf{F}_{02} \perp y-$axis, $\mathbf{F}_{f02}$ is the friction force between the ground 0 and slider 2, $F_{f02} = \mu F_{02}$, $\mathbf{F}_{12}$ is the force of prism 1 on slider 2, $\mathbf{F}_{f12}$ is friction force between 1 and 2, $F_{f12} = \mu F_{12}$ and $\mathbf{G}_2$ is the weight of body 2. The frictional force is equal to the product of the static coefficient of friction with the normal force between the bodies in contact. As the direction of the frictional force must resist the tendency to slip. The MATLAB commands for the forces on slider 2 are:

```
F02_ = [-F02 0 0];
Ff02_ = [0 mu*F02 0];
F12_ = [-F12*sin(alpha) F12*cos(alpha) 0];
Ff12_ = [mu*F12*cos(alpha) mu*F12*sin(alpha) 0];
G2_ = [0 -m2*g 0];
```

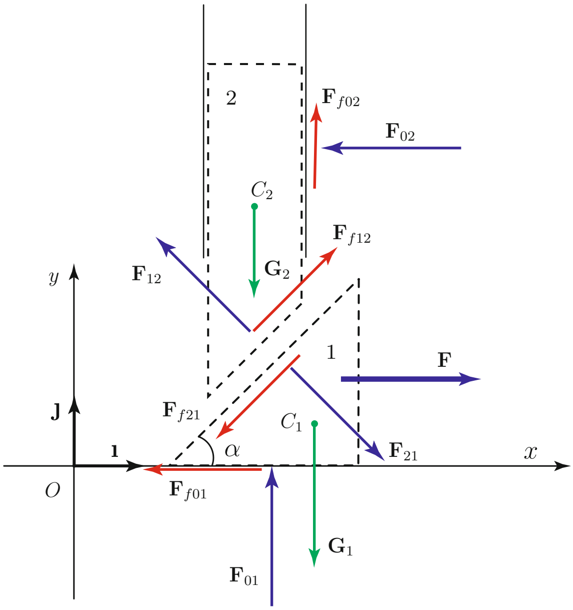


Fig. 5.5 Example 5.1 Free-Body diagrams

```
F2_  = F02_+Ff02_+F12_+Ff12_+G2_;
F2x = F2_(1);
F2y = F2_(2);
```

The two scalar equilibrium equations on $x-$axis and $y-$axis are:

```
Fx on 2:
F12*mu*cos(alpha) - F12*sin(alpha) - F02 = 0
Fy on 2:
F02*mu - g*m2 + F12*cos(alpha) + F12*mu*sin(alpha)=0
```

The joint forces reaction forces F_{02} and F_{12} are

```
F02 = (g*m2*(mu - tan(alpha)))/(mu^2 + 1)
F12 = (g*m2)/(cos(alpha)*(mu^2 + 1))
```

The sum of all the forces acting on the inclined prism 1 can be expressed as

$$\sum \mathbf{F}^{(1)} = \mathbf{F}_{01} + \mathbf{F}_{f01} + \mathbf{F}_{21} + \mathbf{F}_{f21} + \mathbf{G}_1 + \mathbf{F},$$

where $\mathbf{F}_{01}$ is the of force of the ground (vertical support) on wedge 1, $\mathbf{F}_{f01}$ is the friction force between the ground 0 and 1, $F_{f01} = \mu F_{01}$, $\mathbf{F}_{21} = -\mathbf{F}_{12}$ is the force of 2 on 1, $\mathbf{F}_{f21} = -\mathbf{F}_{f12}$ is friction force between 1 and 2, $\mathbf{G}_1$ is the weight of body 1 and $\mathbf{F}$ is the horizontal external force on 1. The MATLAB exressions for the forces on prism 1 are:

```
sol2 = solve(F2x,F2y,F02,F12);
F21_ = ...
-[-F12s*sin(alpha) F12s*cos(alpha) 0];
Ff21_ = ...
-[mu*F12s*cos(alpha) mu*F12s*sin(alpha) 0];
F01_ = [0 F01 0];
Ff01_ = -mu*[F01 0 0];
G1_ =  [0 -m1*g 0];
F_ = [F 0 0];
```

The equilibrium equations for the prism 1 are:

```
F1_ = F01_+Ff01_+F21_+Ff21_+G1_+F_;
F1x = F1_(1);
F1y = F1_(2);
```

or

```
Fx on 1: 0 =

              g m2 mu      g m2 sin(alpha)
  F - F01 mu - ------- + --------------------
                 2                      2
               mu  + 1   cos(alpha) (mu  + 1)
Fy on 1: 0 =

               g m2       g m2 mu sin(alpha)
  F01 - g m1 - ------- - --------------------
                 2                      2
               mu  + 1   cos(alpha) (mu  + 1)
```

The joint force reaction force F_{01} and the external force F are:

```
F01 =

         g m2 (mu tan(alpha) + 1)
  g m1 + ------------------------
                   2
                 mu  + 1
F =

                                 2 g m2 (mu - tan(alpha))
 g (m1 mu+m2 tan(alpha)) + ------------------------
                                          2
                                        mu  + 1
```

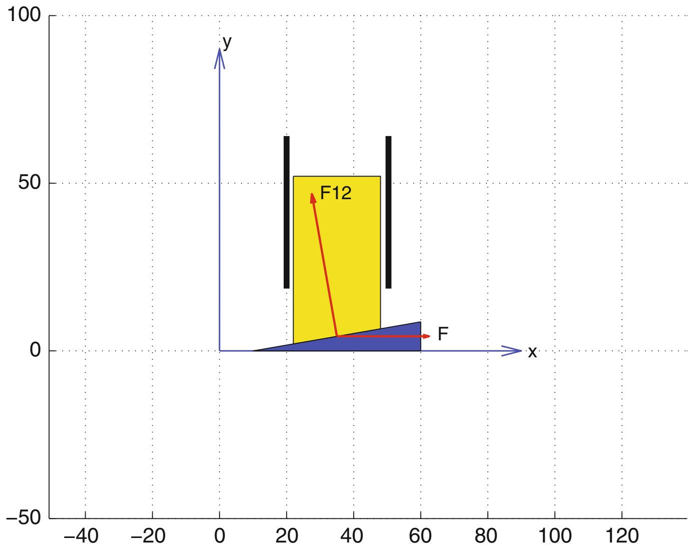


Fig. 5.6 Example 5.1 MATLAB representation of the mechanical system

The numerical results are for the joint forces are:

```
F02 =  1.117 (N), F01 = 146.927 (N), F12 = 47.891 (N)
```

and the greatest value of the horizontal force that acts on the prism 1 without causing the motion of the system is `F = 30.502 (N)`. The MATLAB representation of the mechanical system is shown in Fig. 5.6.

The MATLAB commands for the graphics are:

```
% system plot
aa = 100;
axis([-aa/2 aa -aa/2 aa])
grid on, hold on
axis equal

quiver(0,0,aa,0,...
'Color','b','LineWidth',1.0);
text(aa-8,0,'x','fontsize',12);
quiver(0,0,0,aa,...
'Color','b','LineWidth',1.0);
text(0,aa-8,'y','fontsize',12);

a=60; b=25;c=10;
alpha=10*pi/180;
x_F=c; y_F=0;
```

```
x_H=a; y_H=0;
x_E=x_H; y_E=(x_H-x_F)*sin(alpha);
x_O=0; y_O=0;
% slider vertices
vert = [x_F y_F 0; x_E y_E 0; x_H y_H 0;];
% slider faces
fac = [ 2 1 3];
% draw the slider
slider=patch...
('Faces',fac,'Vertices',vert,'FaceColor','b');

h=50;f=12;
x_A=x_F+f; y_A=(x_A-x_F)*sin(alpha);
x_B=x_A; y_B=y_A+h;
x_D=x_H-f; y_D=(x_D-x_F)*sin(alpha);
x_C=x_D; y_C=y_B;
% prism vertices
vert = ...
[x_A y_A 0; x_B y_B 0; x_D y_D 0; x_C y_C 0;];
% prism faces
fac = [ 2 1 3 4];
% draw the prism
prism=patch...
('Faces',fac,'Vertices',vert,'FaceColor','y');

% draw body left wall
s1x=x_A-f/6; s1y=x_A-f/6;
s2x=y_B+f; s2y=y_D+f;
s1 = [s1x s1y];
s2 = [s2x s2y];
line(s1,s2,'LineStyle','-',...
    'Color','k','LineWidth',4)

% draw body right wall
s3x=x_D+f/5; s3y=x_D+f/5;
s4x=y_B+f; s4y=y_D+f;
s3 = [s3x s3y];
s4 = [s4x s4y];
line(s3,s4,'LineStyle','-',...
    'Color','k','LineWidth',4)

quiver(x_A+(x_D-x_A)/2,y_A+(y_D-y_A)/2,...
F12n*cos(alpha+pi/2),F12n*sin(alpha+pi/2),...
'Color','r','LineWidth',1.5);
t0=text(abs(F12n*cos(alpha+pi/2))+20,...
```

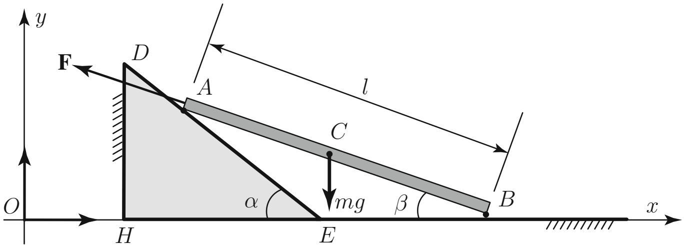


Fig. 5.7 Example 5.2

```
F12n*sin(alpha+pi/2), 0,' F12','fontsize',12);

quiver(x_A+(x_D-x_A)/2,y_A+(y_D-y_A)/2,Fn,0,...
'Color','r','LineWidth',1.5);
t0=text(abs(Fn)+33,5, 0,' F','fontsize',12);
```

Example 5.2 The rod *AB* of length l shown in Fig. 5.7 is supported at end B on a horizontal surface and at the other end (end A) by a inclined surface which makes an angle α with the horizontal. The coefficient of static friction between the rod and the inclined surface and the horizontal surface is μ. The weight of the rod is $G = mg$. The end A of the rod (supported by the inclined surface) is positioned in such a way that the angle between the rod and the horizontal supporting surface is equal to β. The end A of the rod can slid down on the inclined plane. A force F which has its directions parallel with the rod is applied at the left end A of the rod as shown in Fig. 5.7. Determine the greatest value of F without causing the motion of the rod. For the numerical example use: $l = 1$ m, $m = 10$ kg, $\mu = 0.2$, $\beta = \pi/6$ rad, $\alpha = \pi/4$ rad, and $g = 9.81$ m/s^2.

Solution The frictional force at the contact point of the end of the rod with the inclined plane is equal to the product of the static coefficient of friction with the normal force between that same end of the rod and the inclined surface. The normal force is always perpendicular at the contact point to the inclined plane, while the friction force is parallel to the inclined plane. As the direction of the frictional force must resist the tendency to slip, the frictional force must be acting "up" the inclined plane. At the other end of the rod resting on a horizontal surface. The force exerted by the horizontal surface must be vertical and the frictional force is equal to the product of the static coefficient of friction with the normal force as shown in Fig. 5.8.

The weight **G** is also a vertical force. The sum of all the forces acting on the left end A can be expressed as

$$\sum \mathbf{F} = \mathbf{N}_A + \mathbf{F}_{fA} + \mathbf{F} + \mathbf{N}_B + \mathbf{F}_{fB} + \mathbf{G} = \mathbf{0},$$

where $\mathbf{N}_A$ is the of force of the ground (prism) on the rod at A, $\mathbf{F}_{fA}$ is the friction force between the prism and the rod at A, $F_{fA} = \mu N_A$, **F** is the external force

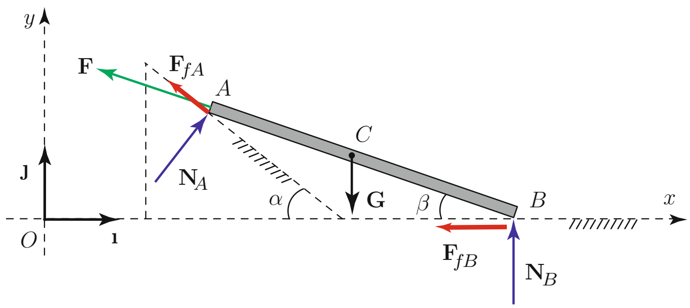


Fig. 5.8 Example 5.2 Free-Body diagram

along the rod at A, $\mathbf{N}_B$ is the of force of the horizontal surface (ground) on the rod at B, $\mathbf{N}_B \perp x-$axis, $\mathbf{F}_{fB}$ is the friction force between the ground and the rod at B, $F_{fB} = \mu N_B$, and $\mathbf{G}_2$ is the weight of the rod. The MATLAB commands for the forces on the rod are:

```
syms NA NB F
NA_=[NA*sin(alpha) NA*cos(alpha) 0];
FfA=mu*NA;
FfA_=[-FfA*cos(alpha) FfA*sin(alpha) 0];
F_=[-F*cos(beta) F*sin(beta) 0];
NB_=[0 NB 0];
FfB=mu*NB;
FfB_=[-FfB 0 0];
G_=[0 -m*g 0];
eqF_=NA_+FfA_+F_+NB_+FfB_+G_;
eqFx=eqF_(1);
eqFy=eqF_(2);
```

The sum of the moments of all forces about the left end A of the rod can be expressed as

$$\sum M_A = -\frac{l}{2}mg\ \cos\beta + l\, N_B \cos\beta = 0.$$

The scalar equilibrium equations in MATLAB are:

```
sum Fx:
NA*sin(alpha)-F*cos(beta)-NB*mu-NA*mu*cos(alpha)=0
sum Fy:
NB-g*m+NA*cos(alpha)+F*sin(beta)+NA*mu*sin(alpha)=0
moment about A:
NB*l*cos(beta)-(g*l*m*cos(beta))/2=0
```

The normal reactions are:

```
NA =

        g m (cos(beta) + mu sin(beta))
   ------------------------------------------
   2 (cos(alpha - beta) + mu sin(alpha - beta))

NA = 46.560 (N)

NB =

   g m
   ---
    2

NB = 49.050 (N)
```

and the greatest value of F without causing the motion of the rod is

```
 F =
                         2
  g m (sin(alpha) mu + 2 cos(alpha) mu - sin(alpha))
- --------------------------------------------------
     2 (cos(alpha-beta) + mu sin(alpha-beta))

 F = 19.085 (N)
```

The MATLAB representation of the mechanical system is shown in Fig. 5.9.

5.5 Technical Applications of Friction: Screws

A screw thread is a uniform wedge-shaped section in the form of a helix on the external or internal surface of a cylinder (straight thread) or a cone (taper thread). The basic arrangement of a helical thread wound around a cylinder is illustrated in Fig. 5.10.

The terminology of an external screw threads is:

- *pitch* denoted by p is the distance, parallel to the screw axis, between corresponding points on adjacent thread forms having uniform spacing.
- *major diameter* denoted by d is the largest (outside) diameter of a screw thread.
- *minor diameter* denoted by d_r or d_1, is the smallest diameter of a screw thread.
- *pitch diameter* denoted by d_m or d_2 is the imaginary diameter for which the width of the threads and the grooves are equal.

The *lead* denoted by l is the distance the nut moves parallel to the screw axis when the nut is given one turn (distance a threaded section moves axially in one revolution).

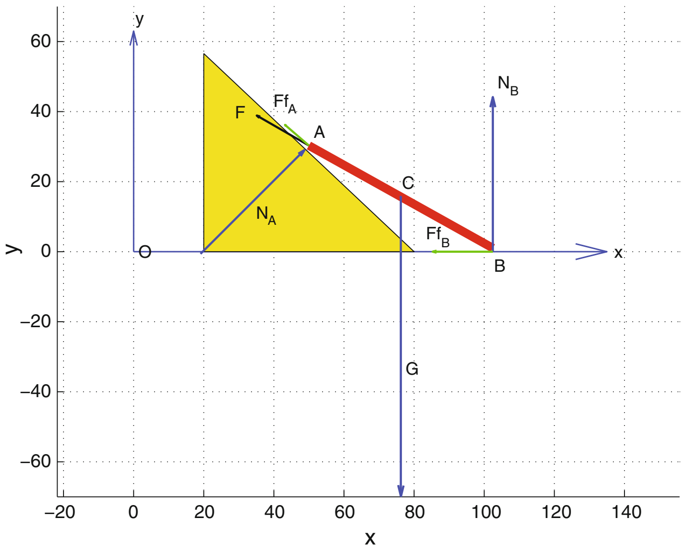
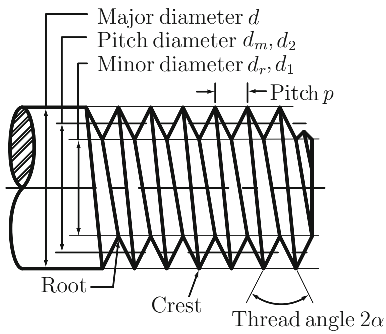


Fig. 5.9 Example 5.2 MATLAB representation of the mechanical system

Fig. 5.10 Screw thread



A screw with two or more threads cut beside each other is called a *multiple-threaded* screw. The lead is equal to twice the pitch for a double-threaded screw, and up to 3 times the pitch for a triple-threaded screw. The pitch p, lead l, and lead angle λ are represented in Fig. 5.11.

Figure 5.11a shows a single thread right-hand screw and Fig. 5.11b shows a double-threaded left-hand screw. If a thread traverses a path in a clockwise and receding direction when viewed axially, it is a *right-hand thread*. All threads are assumed to be right-hand, unless otherwise specified.

Metric threads are specified by the letter M preceding the nominal major diameter in millimeters and the pitch in millimeters per thread. For example: M 14 × 2, M is the SI thread designation, 14 mm is the outside (major) diameter, and the pitch is 2 mm per thread. Screw size in the Unified system is designated by the size number

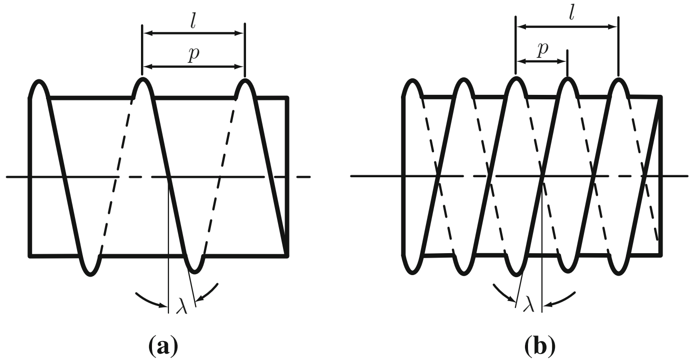
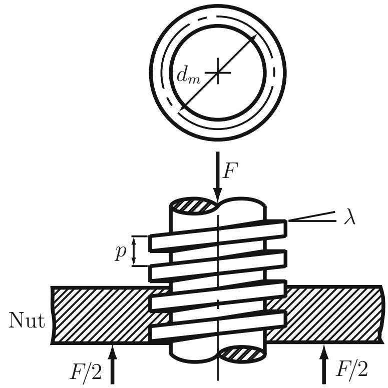


Fig. 5.11 Pitch p, lead l, and lead angle λ. **a** single thread-right hand, **b** double thread-left hand

Fig. 5.12 Power screw with a single thread



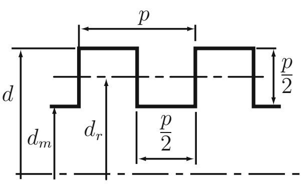
for major diameter (in.), the number of treads per in., and the thread form and series, like this: $\frac{5''}{8} - 18$, UNF $\frac{5''}{8}$ is the outside (major) diameter where the double tick marks mean inches, and 18 threads per in.

5.5.1 Power Screws

Power screws are used to convert rotary motion to linear motion of the meeting member along the screw axis. These screws are used to lift weights (screw-type jacks) or exert large forces (presses, tensile testing machines). The power screws can also be used to obtain precise positioning of the axial movement.

A square-threaded power screw with a single thread having the pitch diameter d_m, the pitch p, and the helix angle λ is considered in Fig. 5.12. A square thread profile is shown in Fig. 5.13.

Fig. 5.13 Square thread



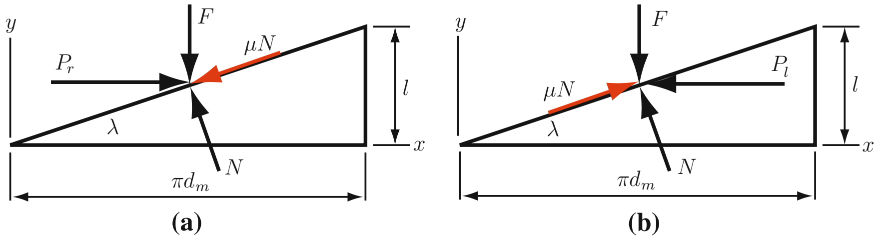


Fig. 5.14 **a** Force diagram for lifting the load and **b** force diagram for lowering the load

Consider that a single thread of the screw is unrolled for exactly one turn. The edge of the thread is the hypotenuse of a right triangle and the height is the lead. The base of the right triangle is the circumference of the pitch diameter circle (Fig. 5.14). The lead angle λ is the helix angle of the thread. The screw is loaded by an axial compressive force F (Figs. 5.12 and 5.14). The force diagram for lifting the load is shown in Fig. 5.14a, (the force P_r is positive). The force diagram for lowering the load is shown in Fig. 5.14b, (the force P_l is negative). The friction force is

$$F_f = \mu N,$$

where μ is the coefficient of dry friction and N is the normal force. The friction force is acting opposite to the motion. The equilibrium of forces for raising the load gives

$$\sum F_x = P_r - N \sin\lambda - \mu N \cos\lambda = 0, \tag{5.3}$$

$$\sum F_y = F + \mu N \sin\lambda - N \cos\lambda = 0. \tag{5.4}$$

Similarly, for lowering the load one may write the equations

$$\sum F_x = -P_l - N \sin\lambda + \mu N \cos\lambda = 0, \tag{5.5}$$

$$\sum F_y = F - \mu N \sin\lambda - N \cos\lambda = 0. \tag{5.6}$$

Eliminating N and solving for P_r

$$P_r = \frac{F(\sin\lambda + \mu \cos\lambda)}{\cos\lambda - \mu \sin\lambda}, \tag{5.7}$$

and for lowering the load

$$P_l = \frac{F\ (\mu\ \cos\lambda - \sin\lambda)}{\cos\lambda + \mu\ \sin\lambda}. \tag{5.8}$$

Using the relation

$$\tan\lambda = l/(\pi d_m),$$

and dividing the equations by $\cos\lambda$ one may obtain

$$P_r = \frac{F\ [(l\ \pi\ d_m) + \mu]}{1 - (\mu\ l\ \pi\ d_m)}, \tag{5.9}$$

$$P_l = \frac{F\ [\mu - (l\ \pi\ d_m)]}{1 + (\mu\ l\ \pi\ d_m)}. \tag{5.10}$$

The moment required to overcome the thread friction and to raise the load is

$$M_r = P_r \frac{d_m}{2} = \frac{F\ d_m}{2}\left(\frac{l + \pi\ \mu\ d_m}{\pi\ d_m - \mu\ l}\right). \tag{5.11}$$

The moment required to lower the load (and to overcome a part of the friction) is

$$M_l = \frac{F\ d_m}{2}\left(\frac{\pi\ \mu\ d_m - l}{\pi\ d_m + \mu\ l}\right). \tag{5.12}$$

When the lead, l, is large or the friction, μ, is low the load will lower itself. In this case the screw will spin without any external effort, and the moment M_l in Eq. (5.12) will be negative or zero. When the moment is positive, $M_l > 0$, the screw is said to be *self-locking*. The condition for self-locking is

$$\pi\ \mu\ d_m > l.$$

Dividing both sides of this inequality by $\pi\ d_m$, and using $l/(\pi\ d_m) = \tan\lambda$, yields

$$\mu > \tan\lambda. \tag{5.13}$$

The self-locking is obtained whenever the coefficient of friction is equal to or greater than the tangent of the thread lead angle.

The moment, M_0, required only to raise the load when the friction is zero, $\mu = 0$, is obtained from Eq. (5.11):

$$M_0 = \frac{F\ l}{2\ \pi}. \tag{5.14}$$

The screw efficiency e can be defined as

$$e = \frac{M_0}{M_r} = \frac{F\,l}{2\,\pi\,M_r}. \tag{5.15}$$

For square threads the normal thread load, F, is parallel to the axis of the screw. The preceding equations can be applied for square threads.

5.5.2 Force Analysis for a Square-Threaded Screw

Consider a square-threaded jack under the action of a axial load F and a moment M about the axis of the screw, Fig. 5.15. The screw has the mean radius r_m and the lead l. The force exerted by the frame thread on the screw thread is R. The angle θ made by R with the normal to the thread is the angle of friction (see Fig. 5.15)

$$\tan\theta = \mu = \frac{F_f}{N}.$$

The unwrapped thread of the screw shown in Fig. 5.15 is for lifting the load. The force equilibrium equation in the axial direction is

$$F = R\,\cos(\lambda + \theta),$$

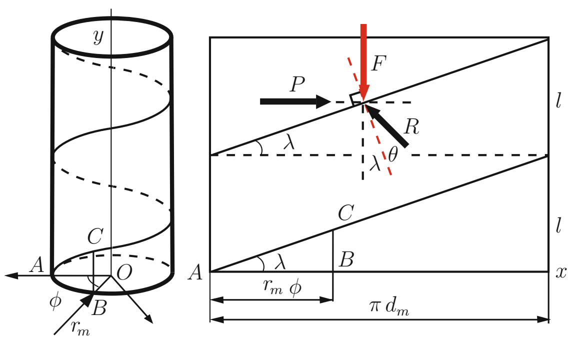
where λ is the helix angle, $\tan\lambda = l/(2\,\pi\,r_m)$. The moment of R about the vertical axis of the screw is $R\,r_m\,\sin(\lambda+\theta)$. The moment equilibrium equation for the screw becomes

$$M = R\,r_m\,\sin(\lambda + \theta).$$

Combining the expression for F and M gives

$$M = M_r = F\,r_m\,\tan(\lambda + \theta). \tag{5.16}$$

Fig. 5.15 Force diagram for a square-fhreaded screw



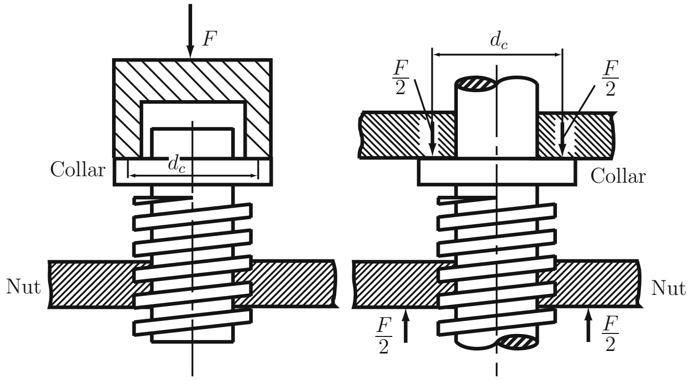


Fig. 5.16 Thrust collar

The force required to push the thread up is $P = M/r_m$. The moment required to lower the load by unwinding the screw is obtained in a similar manner:

$$M = M_l = F\, r_m\, \tan(\theta - \lambda). \tag{5.17}$$

If $\theta < \lambda$ the screw will unwind by itself.

In general, when the screw is loaded axially, a thrust bearing or thrust collar may be used between the rotating and stationary links to carry the axial component (Fig. 5.16). The load is concentrated at the mean collar diameter d_c. The moment required is

$$M_c = \frac{F\, \mu_c\, d_c}{2}, \tag{5.18}$$

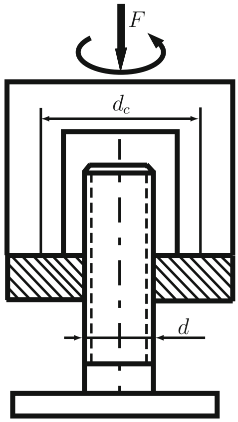
where μ_c is the coefficient of collar friction.

Example 5.3 A square-thread power screw, as shown in Fig. 5.17 has the major diameter $d = 38$ mm and the pitch $p = 6$ mm. The coefficient of friction of the thread is $\mu = 0.08$ and the coefficient of collar friction is $\mu_c = 0.1$. The mean collar diameter is $d_c = 45$ mm. The external load on the screw is $F = 9$ kN. Find: (a) the lead, the pitch (mean) diameter and the minor diameter; (b) the moment required to raise the load; (c) the moment required to lower the load; (d) the efficiency of the device.

Solution (a) From Fig. 5.13:
the minor diameter is $d_r = d - p$
the pitch (mean) diameter is $d_m = d - p/2$
the lead is$l = p$,
or:

```
l = p =  6.000 (mm)
dr = d - p = 32.000 (mm)
```

Fig. 5.17 Example 5.1




```
dm = d - p/2 = 35.000 (mm)
```

(b) The moment required to raise the load is

$$M_r = \frac{F d_m}{2}\left(\frac{l + \pi\mu d_m}{\pi d_m - \mu l}\right) + \frac{F\mu_c d_c}{2},$$

and:

```
Mr =0.5*F*dm*(pi*mu*dm+l)/(pi*dm-mu*l)...
    +0.5*F*dc*muc;
Mr = 41.537 (kN m)
```

(c) The moment required to lower the load is

$$M_l = \frac{F d_m}{2}\left(\frac{\pi\mu d_m - l}{\pi d_m + \mu l}\right) + \frac{F\mu_c d_c}{2},$$

and:

```
Ml =0.5*F*dm*(pi*mu*dm-l)/(pi*dm+mu*l)...
    +0.5*F*dc*muc;
Ml = 24.238 (kN m)
```

The sef-locking condition:

```
%  (pi*mu*dm - l) > 0
sf = (pi*mu*dm - l);
fprintf('sf = %6.3f \n',sf)
if sf > 0
fprintf('sf>0 => screw is sef-locking\n\n')
else
```

```
fprintf('sf<0 => screw is not sef-locking\n\n')
end
```

The screw is self-locking `sf = 2.796`.

(d) The overall efficiency is calculated with

$$e = \frac{F l}{2\pi M_r},$$

and:

```
e = F*l/(2*pi*Mr);
e =  0.207
```

5.6 Problems

5.1 Find the orientation angle θ of the force P for the smallest possible force P that can be applied so that the body shown in Fig. 5.18 is on the verge of moving. The body has weight the mass m and the coefficient of static friction at the surface is $\mu_s = 0.4$.

5.2 The car shown in Fig. 5.19 has the mass m, the center of mass at C, and travels along a track with a constant speed. Find the greatest slope θ of the track without causing the car to tip or to slip. The coefficient of static friction between the road and the car is μ. For the numerical example use: $m = 2000$ kg, $\mu = 0.3$, $l = 1.75$ m, $h = 0.5$ m, and $g = 9.81$ m/s^2.

5.3 A uniform bar of mass m and length l is placed on a rough wall at A and on a smooth floor at B at shown in Fig. 5.20. The coefficient of static friction between the bar and the wall is μ. The distance $OB = d$ is given. Find if the bar will remain in the initial position when it is released. For the numerical example use: $\mu = 0.4$, $l = 1$ m, $d = OB = 0.3$ m, and $g = 9.81$ m/s^2.

5.4 The block 1, shown in Fig. 5.21, has the mass m and is placed on a rough wall. Find the minimum force F required to move the block of mass m. The coefficient of static friction is μ. The angle of the two wedges 2 and 3 is α. For the numerical example use: $m = 80$ kg, $\mu = 0.4$, $\alpha = \pi/180$ rad, and $g = 9.81$ m/s^2.

5.5 The wedge 3, shown in Fig. 5.22, has the mass $2\,m$. The wedges 2 has the mass m. Find the minimum force F required to move the wedge 3. The coefficient of

Fig. 5.18 Problem 5.1

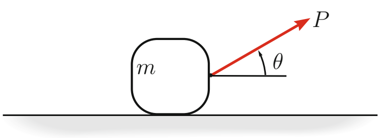


Fig. 5.19 Problem 5.2

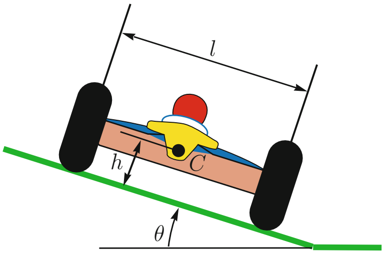


Fig. 5.20 Problem 5.3

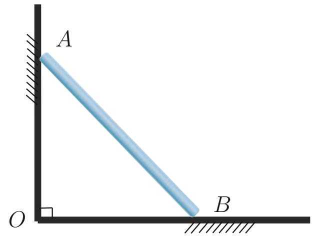


Fig. 5.21 Problem 5.4

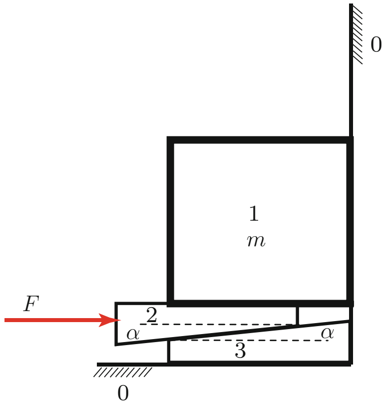
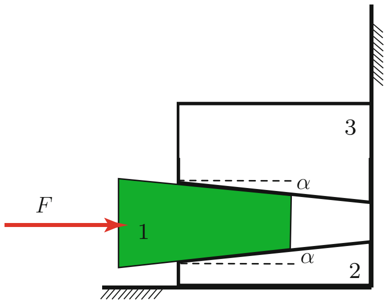


Fig. 5.22 Problem 5.5



static friction is μ. The angle of the wedges is α. For the numerical example use: $m = 10\,\text{kg}$, $\mu = 0.2$, $\alpha = 15°$ rad, and $g = 9.81\,\text{m/s}^2$.

5.6 A double square-thread power screw has a pitch (mean) diameter of 30 mm and a pitch of 6 mm. The coefficient of friction of the thread is 0.1 and the coefficient of collar friction is also 0.2. The mean collar diameter is 40 mm. The external load on the screw is 8 kN. Determine the moment required to lower the load and the overall efficiency.

5.7 A power screw has a double square thread with a mean diameter of 50 mm and a pitch of 8 mm. The coefficient of friction in the thread is 0.15. Determine if the screw is self-locking.

5.8 A triple-thread screw is used in a jack to raise a load of 3000 lb. The major diameter of the screw is 4 in. A plain thrust collar is used. The mean diameter of the collar is 5 in. The coefficient of friction of the thread is 0.08 and the coefficient of collar friction is 0.2. Determine: (a) the screw pitch, lead, thread depth, mean pitch diameter, and helix angle; (b) the starting moment for raising and for lowering the load; (c) the efficiency of the jack.

5.7 Programs

5.7.1 Program 5.1

```
% example 5.1
clear all; clc; close all

syms F02 F12 F01 F
syms m1 m2 g alpha mu

list ={m1,m2,g,alpha,mu};
listn={10,5,9.81,10*pi/180,0.2};

% slider 2
% force of ground (vertical support) on body 2
F02_ = [-F02 0 0];
% friction force between 0 and 2
Ff02_ = [0 mu*F02 0];
% force of prism 1 on body 2
F12_ = ...
[-F12*sin(alpha) F12*cos(alpha) 0];
% friction force between 1 and 2
Ff12_ = ...
[mu*F12*cos(alpha) mu*F12*sin(alpha) 0];
% weight of body 2
```

```
G2_ = [0 -m2*g 0];
% sum of forces on body 2
F2_ = F02_+Ff02_+F12_+Ff12_+G2_;
F2x = F2_(1);
F2y = F2_(2);

fprintf('Fx on 2:\n')
fprintf(' %s = 0 \n',char(F2x))
fprintf('Fy on 2:\n')
fprintf(' %s = 0 \n',char(F2y))

sol2 = solve(F2x,F2y,F02,F12);
F02s = simple(sol2.F02);
F12s = simple(sol2.F12);
fprintf('=>\n')
fprintf('F02 = %s\n',char(F02s))
fprintf('F12 = %s\n',char(F12s))
fprintf('\n')

% prism 1
% force of body 2 on prism 1
F21_ = ...
-[-F12s*sin(alpha) F12s*cos(alpha) 0];
% friction force between 2 and 1
Ff21_ = ...
-[mu*F12s*cos(alpha) mu*F12s*sin(alpha) 0];

% force of ground on prism 1
F01_ = [0 F01 0];
% friction force between 0 prism 1
Ff01_ = -mu*[F01 0 0];
% weight of prism 1
G1_ =  [0 -m1*g 0];
% external force of prism 1
F_ = [F 0 0];

% sum of forces on prism 1
F1_ = F01_+Ff01_+F21_+Ff21_+G1_+F_;
F1x = F1_(1);
F1y = F1_(2);

fprintf('Fx on 1: 0 = \n')
pretty(F1x)
fprintf('Fy on 1: 0 = \n')
pretty(F1y)
```

```
sol1 = solve(F1x,F1y,F01,F);
F01s = simple(sol1.F01);
Fs = simple(sol1.F);
fprintf('=>\n')
fprintf('F01 = \n')
pretty(F01s)
fprintf('F = \n')
pretty(Fs)

F02n = subs(F02s,list,listn);
F01n = subs(F01s,list,listn);
F12n = subs(F12s,list,listn);
Fn = subs(Fs,list,listn);

fprintf('\n\n')
fprintf('F02 = %6.3f (N) \n',F02n)
fprintf('F01 = %6.3f (N) \n',F01n)
fprintf('F12 = %6.3f (N) \n',F12n)
fprintf('F = %6.3f (N) \n',Fn)

% system plot
aa = 100;
axis([-aa/2 aa -aa/2 aa])
grid on, hold on
axis equal

quiver(0,0,aa,0,...
'Color','b','LineWidth',1.0);
text(aa-8,0,'x','fontsize',12);
quiver(0,0,0,aa,...
'Color','b','LineWidth',1.0);
text(0,aa-8,'y','fontsize',12);

a=60; b=25;c=10;
alpha=10*pi/180;
x_F=c; y_F=0;
x_H=a; y_H=0;
x_E=x_H; y_E=(x_H-x_F)*sin(alpha);
x_O=0; y_O=0;
% slider vertices
vert = [x_F y_F 0; x_E y_E 0; x_H y_H 0;];
% slider faces
fac = [ 2 1 3];
% draw the slider
```

```
slider=patch...
('Faces',fac,'Vertices',vert,'FaceColor','b');

h=50;f=12;
x_A=x_F+f; y_A=(x_A-x_F)*sin(alpha);
x_B=x_A; y_B=y_A+h;
x_D=x_H-f; y_D=(x_D-x_F)*sin(alpha);
x_C=x_D; y_C=y_B;
% prism vertices
vert = ...
[x_A y_A 0; x_B y_B 0; x_D y_D 0; x_C y_C 0;];
% prism faces
fac = [ 2 1 3 4];
% draw the prism
prism=patch...
('Faces',fac,'Vertices',vert,'FaceColor','y');

% draw body left wall
s1x=x_A-f/6; s1y=x_A-f/6;
s2x=y_B+f; s2y=y_D+f;
s1 = [s1x s1y];
s2 = [s2x s2y];
line(s1,s2,'LineStyle','-',...
    'Color','k','LineWidth',4)

% draw body right wall
s3x=x_D+f/5; s3y=x_D+f/5;
s4x=y_B+f; s4y=y_D+f;
s3 = [s3x s3y];
s4 = [s4x s4y];
line(s3,s4,'LineStyle','-',...
    'Color','k','LineWidth',4)

quiver(x_A+(x_D-x_A)/2,y_A+(y_D-y_A)/2,...
F12n*cos(alpha+pi/2),F12n*sin(alpha+pi/2),...
'Color','r','LineWidth',1.5);
t0=text(abs(F12n*cos(alpha+pi/2))+20,...
F12n*sin(alpha+pi/2), 0,' F12','fontsize',12);

quiver(x_A+(x_D-x_A)/2,y_A+(y_D-y_A)/2,Fn,0,...
'Color','r','LineWidth',1.5);
t0=text(abs(Fn)+33,5, 0,' F','fontsize',12);
% end of program
```

5.7.2 Program 5.2

```
% example 5.2
clear all; clc; close all

syms l m g mu alpha beta
syms NA NB F

NA_=[NA*sin(alpha) NA*cos(alpha) 0];

FfA=mu*NA;
FfA_=[-FfA*cos(alpha) FfA*sin(alpha) 0];

F_=[-F*cos(beta) F*sin(beta) 0];

NB_=[0 NB 0];

FfB=mu*NB;
FfB_=[-FfB 0 0];

G_=[0 -m*g 0];

eqF_=NA_+FfA_+F_+NB_+FfB_+G_;

eqFx=eqF_(1);
eqFy=eqF_(2);

eqMA=-l/2*m*g*cos(beta)+l*NB*cos(beta);

fprintf('sum Fx:\n')
fprintf(' %s = 0 \n',char(eqFx))
fprintf('sum Fy:\n')
fprintf(' %s = 0  \n',char(eqFy))
fprintf('moment about A:\n');
fprintf(' %s = 0 \n\n',char(eqMA))

sol=solve(eqFx,eqFy,eqMA,F,NA,NB);

NAs=simplify(sol.NA);
NBs=sol.NB;
Fs=simplify(sol.F);

list={l,m,g,mu,beta,alpha};
listn={1,10,9.81,0.2,pi/6,pi/4};
```

```
NAn=subs(NAs,list,listn);
NBn=subs(NBs,list,listn);
Fn=subs(Fs,list,listn);

fprintf('NA =\n')
pretty(NAs);
fprintf('\n')
fprintf('NA = %6.3f (N)\n\n',NAn)
fprintf('NB =\n')
pretty(NBs);
fprintf('\n')
fprintf('NB = %6.3f (N)\n\n',NBn)
fprintf('F =\n')
pretty(Fs);
fprintf('\n')
fprintf('F = %6.3f (N)\n\n',Fn)

% graphic
a = 70;
axis([-a/2 a -a a])
grid on, hold on
axis equal
xlabel('x'), ylabel('y')

quiver(0,0,2*a+10,0,...
'Color','b','LineWidth',1.0);
text(2*a,0,' x','fontsize',12);
quiver(0,0,0,a,...
'Color','b','LineWidth',1.0);
text(0,a,' y','fontsize',12);

a=80; b=35;c=20;
alpha=45*pi/180;
x_O=0; y_O=0;
x_F=c; y_F=0;
x_E=a; y_E=0;
x_D=c; y_D=(x_E)*sin(alpha);

t0=text(x_O, y_O, 0,' O','fontsize',12);
t1=text(x_F-1, y_F-4, 0,' F','fontsize',12);
t2=text(x_D-1, y_D+2, 0,' D','fontsize',12);
t3=text(x_E-1, y_E-4, 0,' E','fontsize',12);
% prism vertices
vert = [x_F y_F 0; x_D y_D 0; x_E y_E 0;];
```

```
% prism faces
fac = [ 2 1 3];
% draw the prism
prism=patch...
('Faces',fac,'Vertices',vert,'FaceColor','y');

offset=1;
beta=30*pi/180;
x_A=x_F+(x_E-x_F)/2; y_A=y_D/2+2*offset;
x_B=x_A+y_A/tan(beta); y_B=offset;
A = [x_A x_B];
B = [y_A y_B];
x_C=x_A+(x_B-x_A)/2;
y_C=(y_A)/2+offset/2;
C = [x_C y_C];

t4=text...
(x_B-1, y_B-5, 0,' B','fontsize',12);
t5=text...
(x_A, y_A+4, 0,' A','fontsize',12);
t6=text...
(x_C-1, y_C+4, 0,' C','fontsize',12);
% draw the rod
line(A,B,'LineStyle','-',...
    'Color','r','LineWidth',6)
scatter(x_C,y_C,3,5,'filled','b')

quiver(....
x_A-NAn*cos(alpha)+2*offset,...
y_A-NAn*sin(alpha)+2*offset,...
NAn*cos(alpha),NAn*sin(alpha),...
'Color','b','LineWidth',1.5);
t0=text(...
x_A-NAn*cos(alpha)/2,...
y_A-NAn*sin(alpha)/2-5*offset,...
0,' N_A','fontsize',12);

quiver(...
x_A,y_A,Fn*cos(beta+pi),...
Fn*sin(beta),...
'Color','k','LineWidth',1.5);
t0=text(...
x_A-Fn*cos(alpha)-9*offset,...
y_A+Fn*sin(alpha)-4*offset,...
0,' F','fontsize',12);
```

```
FfA=subs(FfA,list,listn);
FfA=subs(FfA,NAn,NA);
quiver(...
x_A,y_A,...
FfA*cos(alpha+pi)-offset,...
FfA*sin(alpha),...
'Color','g','LineWidth',1.5);
t0=text(...
x_A-Fn*cos(alpha)+2*offset,...
y_A+Fn*sin(alpha)-3.0*offset,...
0,' Ff_A','fontsize',12);

quiver(x_B,y_B-offset,0,NBn,...
'Color','b','LineWidth',1.5);
t0=text(...
x_B,y_B+NBn-4*offset,...
0,' N_B','fontsize',12);

quiver(x_B,y_B-offset,-Fn,0,...
'Color','g','LineWidth',1.5);
t0=text(...
x_B-Fn,y_B+2*offset, 0,...
'Ff_B','fontsize',12);

G=subs(G_,list,listn);
quiver(x_C,y_C,0,G(2),...
'Color','b','LineWidth',1.5);
t0=text(...
x_C,y_C+G(2)/2, 0,...
' G','fontsize',12);
% end of program
```

5.7.3 Program 5.3

```
% example 5.3
clear all; clc; close all

% major diameter d
d = 38; % mm
% screw pitch p
p = 6;  % mm
% coefficient of friction for thread
mu = 0.08;
```

```
% coefficient of collar friction
muc = 0.1;
% mean collar diameter dc
dc = 45; % mm
% external load F
F = 9;   % kN

% lead l
l = p;
% minor diameter dr
dr = d-p;
% (pitch) mean diameter dm
dm = d-p/2;

fprintf('l = p = %6.3f (mm)\n',l)
fprintf('dr = d - p = %6.3f (mm)\n',dr)
fprintf('dm = d - p/2 = %6.3f (mm)\n\n',dm)

% momemnt required to raise the load
Mr =0.5*F*dm*(pi*mu*dm+l)/(pi*dm-mu*l)...
    +0.5*F*dc*muc;
fprintf('Mr = %6.3f (kN m)\n\n',Mr)

% moment required to lower load
Ml =0.5*F*dm*(pi*mu*dm-l)/(pi*dm+mu*l)...
    +0.5*F*dc*muc;
fprintf('Ml = %6.3f (kN m)\n\n',Ml)

% sef-locking condition: (pi*mu*dm - l) > 0
sf = (pi*mu*dm - l);
fprintf('sf = %6.3f \n',sf)

if sf > 0
fprintf('sf>0 => screw is sef-locking\n\n')
else
fprintf('sf<0 => screw is not sef-locking\n\n')
end

% If  sf > 0 => the screw is sef-locking
% If  sf <= 0 => the screw is not sef-locking

% efficiency
e = F*l/(2*pi*Mr);
fprintf('e = %6.3f \n\n',e)
% end of program
```

References

1. P. Appell, *Traité de mécanique rationnelle* (Gauthier-Villars, Paris, 1955)
2. M. Atanasiu, *Mechanics* (EDP, Bucharest, 1973)
3. H. Baruh, *Analytical Dynamics* (WCB/McGraw-Hill, Boston, 1999)
4. G. Baumann, *Mathematica for Theoretical Physics: Classical Mechanics and Nonlinear Dynamics* (Springer, New York, 2005)
5. G. Baumann, *Mathematica for Theoretical Physics: Electrodynamics, Quantum Mechanics, General Relativity and Fractals* (Springer, New York, 2005)
6. F.P. Beer, E.R. Johnston, *Vector Mechanics for Engineers: Statics and Dynamics* 5th edn. (McGraw-Hill Publishing Company, New York, 1988)
7. F.P. Beer, E.R. Johnston, D.F. Mazurek, *Vector Mechanics for Engineers: Statics* 10th edn. McGraw-Hill Publishing Company, New York, 2012)
8. A.M. Bedford, W. Fowler, K.M. Liechti, *Statics and Mechanics of Materials* (Prentice Hall, Inc., Upper Saddle River, NJ, 2002)
9. A.M. Bedford, W. Fowler, *Engineering Mechanics: Statics* 5th edn. (Prentice Hall, Inc., Upper Saddle River, NJ, 2007)
10. A.P. Boresi, R.J. Schmidt, *Engineering Mechanics: Statics* (PWS Publishing Company, Boston, 2000)
11. M.I. Buculei, *Mechanics* (University of Craiova Press, Craiova, 1974)
12. M.I. Buculei, D. Bagnaru, G. Nanu, D.B. Marghitu, *Analysis of Mechanisms with Bars* (Scrisul romanesc, Craiova, 1986)
13. I. Stroe et al., *Analytical Mechanics Problems* (University Politehnica of Bucharest, Bucharest, 1997)
14. V. Ceausu, N. Enescu, F. Ceausu, *Mechanics Problems* (Printech, Bucharest, 1999)
15. S.J. Chapman, *MATLAB Programming for Engineers* (Thomson Engineering, 2007).
16. S. Attaway, *MATLAB: A Practical Introduction to Programming and Problem Solving* (Butterworth-Heinemann, Elsevier, Amsterdam, 2012)
17. D.M. Etter, D.C. Kuncicky, *Introduction to MATLAB for Engineers and Scientists* (Prentice Hall, Inc., Upper Saddle River, NJ, 1996)
18. C. Iacob, *Theoretical Mechanics* (EDP, Bucharest, 1980)
19. J.H. Ginsberg, *Advanced Engineering Dynamics* (Cambridge University Press, Cambridge, 1995)
20. D.T. Greenwood, *Principles of Dynamics* (Prentice-Hall, Inc., Englewood Cliffs, NJ, 1998)
21. L.E. Goodman, W.H. Warner, *Statics* (Dover Publications, Inc., New York, 2001)
22. R.C. Hibbeler, *Engineering Mechanics: Statics and Dynamics* 13th edn. (Prentice-Hall, Inc., Upper Saddle River, NJ, 2013)
23. T.R. Kane, *Analytical Elements of Mechanics*, vol. 1 (Academic Press, New York, 1959)
24. T.R. Kane, *Analytical Elements of Mechanics*, vol. 2 (Academic Press, New York, 1961)
25. T.R. Kane, P.W. Likins, D.A. Levinson, *Spacecraft Dynamics* (McGraw-Hill, New York, 1983)
26. T.R. Kane, D.A. Levinson, *Dynamics* (McGraw-Hill, New York, 1985)
27. R. Maeder, *Programming in Mathematica* (Addison-Wesley Publishing Company, Redwood City, CA, 1990)
28. N.H. Madsen, Statics and dynamics, class notes, available at http://www.eng.auburn.edu/users/nmadsen/
29. D.B. Marghitu, *Mechanical Engineer's Handbook* (Academic Press, San Diego, CA, 2001)
30. D.B. Marghitu, M.J. Crocker, *Analytical Elements of Mechanisms* (Cambridge University Press, Cambridge, 2001)
31. D.B. Marghitu, *Kinematic Chains and Machine Component Design* (Elsevier, Amsterdam, 2005)
32. D.B. Marghitu, *Mechanisms and Robots Analysis with MATLAB* (Springer, New York , 2009)
33. D.B. Marghitu, M. Dupac, *Advanced Dynamics: Analytical and Numerical Calculations with MATLAB* (Springer, New York, 2012)

34. D.B. Marghitu, Statics and dynamics, class notes, available at http://www.eng.auburn.edu/users/marghitu/
35. D.J. McGill, W.W. King, *Engineering Mechanics: Statics and an Introduction to Dynamics* (PWS Publishing Company, Boston, 1995)
36. J.L. Meriam, L.G. Kraige, *Engineering Mechanics: Statics* 7th edn. (John Wiley & Sons, New York, 2011)
37. R.L. Mott, *Machine Elements in Mechanical Design* (Prentice Hall, Upper Saddle River, NJ, 1999)
38. R.L. Norton, *Machine Design* (Prentice-Hall, Upper Saddle River, NJ, 1996)
39. L.A. Pars, *A Treatise on Analytical Dynamics* (John Wiley & Sons, New York, 1965)
40. M. Plesha, G. Gray, F. Costanzo, *Engineering Mechanics: Statics* 2nd edn. (McGraw-Hill, New York, 2012)
41. M. Radoi, E. Deciu, *Mechanics* (EDP, Bucharest, 1981)
42. W.F. Riley, L.D. Sturges, *Engineering Mechanics: Statics* 2nd edn. (John Wiley & Sons, Inc., New York, 1995)
43. A. Ruina, R. Pratap, *Introduction to Statics and Dynamics* (Oxford University Press, Oxford, 2002)
44. A. Ripianu, P. Popescu, B. Balan, *Technical Mechanics* (EDP, Bucharest, 1979)
45. I.H. Shames, *Engineering Mechanics Statics* 4th edn. (Prentice Hall, Inc., New Jersey, 1996)
46. S.D. Sheppard, B.H. Tongue, *Statics: Analysis and Design of Systems in Equilibrium* (John Wiley & Sons, New York, 2005)
47. D. Smith, *Engineering Computation with MATLAB* (Pearson Education, Upper Saddle River, NJ, 2008)
48. R.W. Soutas-Little, D.J. Inman, *Engineering Mechanics: Statics and Dynamics* (Prentice-Hall, Upper Saddle River, NJ, 1999)
49. R.W. Soutas-Little, D.J. Inman, D. Balint, *Engineering Mechanics: Statics* (Cengage Learning, Independence, KY, 2007)
50. S. Staicu, *Theoretical Mechanics* (EDP, Bucharest, 1998)
51. A. Stan, M. Grumazescu, *Mechanics Problems* (EDP, Bucharest, 1973)
52. J. Sticklen, M.T. Eskil, *An Introduction to Technical Problem Solving with MATLAB* (Great Lakes Press, Wildwood, MO, 2006)
53. A. Stoenescu, G. Silas, *Theoretical Mechanics* (ET, Bucharest, 1957)
54. J.H. Jackson, H.G. Wirtz, *Schaum's Outline of Theory and Problems of Statics and Strength of Materials* (McGraw-Hill, New York, 1983)
55. The MathWorks: https://www.mathworks.com/
56. Statics eBook : https://ecourses.ou.edu/
57. R. Voinea, D. Voiculescu, V. Ceausu, *Mechanics* (EDP, Bucharest, 1983)
58. V. Valcovici, S. Balan, R. Voinea, *Theoretical Mechanics* (ET, Bucharest, 1959)
59. K.J. Waldron, G.L. Kinzel, *Kinematics, Dynamics, and Design of Machinery* (John Wiley & Sons, New York, 1999)
60. H.B. Wilson, L.H. Turcotte, D. Halpern, *Advanced Mathematics and Mechanics Applications Using MATLAB* (Chapman & Hall/CRC, Boca Raton, 2003)
61. J.H. Williams Jr, *Fundamentals of Applied Dynamics* (John Wiley & Sons, New York, 1996)
62. S. Wolfram, *Mathematica* (Wolfram Media/Cambridge University Press, Cambridge, 1999)

Chapter 6
Virtual Work and Stability

6.1 Virtual Displacement and Virtual Work

A particle in static equilibrium position is considered, Fig. 6.1a. The static equilibrium position of the particle is determined by the forces that act on it. The *virtual displacement*, $\delta\mathbf{r}$, is any arbitrary small displacement away from this natural position and consistent with the system constraints. The term virtual is used to indicate that thedisplacement does not really exist but only is assumed to exist. The *virtual work* is the work done by any force $\mathbf{F}$ acting on the particle during the virtual displacement $\delta\mathbf{r}$:

$$\delta U = \mathbf{F} \cdot \delta\mathbf{r} = F\,\delta r \cos\alpha,$$

where α is the angle between $\mathbf{F}$ and $\delta\mathbf{r}$ ($|\delta\mathbf{r}| = \delta r$). The actual infinitesimal change in position $d\mathbf{r}$ can be integrated and the infinitesimal virtual or assumed movement $\delta\mathbf{r}$ cannot be integrated. Mathematically, both quantities are first-order differentials. The force $\mathbf{F}$ is constant during any infinitesimal virtual displacement.

Consider a particle in equilibrium position as a result of the forces $\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_n$. For an assumed virtual displacement $\delta\mathbf{r}$ of the particle away from its equilibrium position, the total virtual work done on the particle is

$$\delta U = \Sigma\mathbf{F} \cdot \delta\mathbf{r} = \Sigma F_x\,\delta x + \Sigma F_y\,\delta y + \Sigma F_z\,\delta z = 0.$$

The sum is zero, since $\Sigma\mathbf{F} = \mathbf{0}$. The equation $\delta U = 0$ is therefore an alternative statement of the equilibrium conditions for a particle. This condition of zero virtual work for equilibrium is both necessary and sufficient.

The principle of virtual work for a single particle can be extended to a rigid body treated as a system of small elements or particles rigidly attached to one another. Because the virtual work done on each particle of the body in equilibrium is zero, it results that the virtual work done on the entire rigid body is zero.

All the internal forces appear in pairs of equal, opposite, and collinear forces, and the net work done by these forces during any movement is zero. Only the virtual

D. B. Marghitu et al., *Statics with MATLAB®*,
DOI: 10.1007/978-1-4471-5110-4_6,

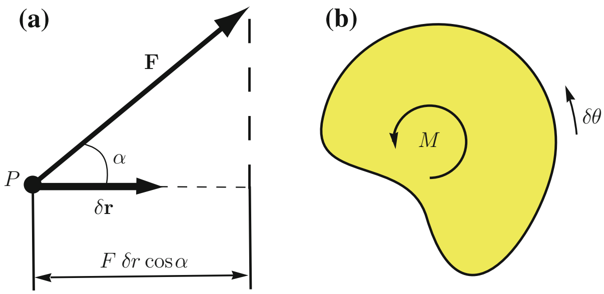


Fig. 6.1 **a** Force acting on a particle and **b** couple acting on an object

work done by external forces are taken into account in the evaluation of $\delta U = 0$ for the entire body. A virtual displacement may also be a rotation $\delta\theta$ of a body as shown in Fig. 6.1b. The virtual work done by a couple M during a virtual angular displacement $\delta\theta$ is $\delta U = M\delta\theta$. The force **F** or couple M remain constant during any infinitesimal virtual displacement.

The principle of virtual work will be extended to the equilibrium of an interconnected ideal system of rigid bodies. The *ideal systems* are systems composed of two or more rigid bodies linked together by mechanical connections which are incapable of absorbing energy through elongation or compression, and in which friction is small enough to be neglected. There are two types of forces which act in such an interconnected system:

- active forces are external forces capable of doing virtual work during possible virtual displacements;
- joint forces are forces in the connections between members. During any possible movement of the system or its parts, the net work done by the joint forces at the connections is zero, because the joint forces always exist in pairs of equal and opposite forces.

Principle of Virtual Work: The work done by external active forces on an ideal mechanical system in equilibrium is zero for any and all virtual displacements consistent with the constraints.

Mathematically, the principle can be expressed as

$$\delta U = 0. \tag{6.1}$$

The advantage of the method of virtual work is that relations between the active forces can be determined directly without reference to the joint forces. The method is useful in determining the position of equilibrium of a system under known forces. The method of virtual work cannot be applied for the system where the internal friction in a mechanical system is appreciable (the work done by internal friction should be included).

6.2 Elastic Potential Energy

The work done by any force **F** acting on the particle due to a differential displacement $d\mathbf{r}$ of the force application point is

$$\mathrm{d}U = \mathbf{F} \cdot \mathrm{d}\mathbf{r}.$$

A function of position V is the *potential energy* associated with the force **F** if for any $\mathrm{d}\mathbf{r}$

$$\mathrm{d}V = -\mathbf{F} \cdot \mathrm{d}\mathbf{r} = -\mathrm{d}U.$$

The force **F** (for which a potential function exists) is called a *conservative force*.

The work done on an elastic body is accumulated in the body in the form of elastic potential energy, V_e. The potential energy can do work on other body during the compression or extension. A spring can store and release potential energy. A spring with the elastic constant or stiffness k is attached to a particle P as shown in Fig. 6.2. The spring is connected to a fixed support. A force **F** acts on the particle and the spring is compressed. The spring exerts a force F_s on the particle. The spring is linear elastic and the force F_s is directly proportional to its deflection x

$$F_s = kx.$$

The work done by the elastic force of the spring, F_s, on the particle is calculated from

$$\mathrm{d}U = -F_s\,\mathrm{d}x,$$

where dx is the differential displacement. The work done is negative since the spring exerts a force F_s on the particle that is opposite to the particle displacement dx

$$U = -\int_0^x F_s\,\mathrm{d}x = -\int_0^x k\,x\mathrm{d}x = -\frac{1}{2}kx^2.$$

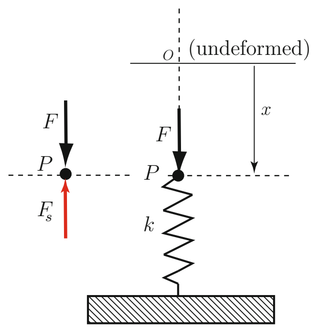


Fig. 6.2 Force acting on a spring

The elastic potential energy of the spring on the attached particle for the compression x is

$$V_e = \int_0^x F_s \,\mathrm{d}x = \int_0^x kx\mathrm{d}x = \frac{1}{2}\,kx^2.$$

For an increase in the compression of the spring from x_1 to x_2 the change in elastic potential energy is

$$\Delta V_e = \int_{x_1}^{x_2} kx\mathrm{d}x = \frac{1}{2}k(x_2^2 - x_1^2).$$

During a virtual displacement δx, the virtual change in elastic potential energy is

$$\delta V_e = F\delta x = kx\delta x.$$

If the spring is elongated, the work and energy relations are the same as those for compression, where x is the stretch of the spring.

For a torsional spring the elastic moment is

$$M_s = K\theta,$$

where K is the torsional stiffness. The potential energy becomes

$$V_e = \int_0^\theta K\theta\mathrm{d}\theta = \frac{1}{2}\,K\theta^2,$$

which is analogous to the expression for the linear extension spring. The torsional spring resists the rotaten. The units of elastic potential energy are the same as those of work and are expressed in joules (J) in SI units and in foot-pounds (ft·lb) in U.S. customary units. The force developed by a an elastic spring is a conservative force.

For springs in parallel having individual spring rates, k_i, Fig. 6.3a, the spring rate k is

$$k = k_1 + k_2 + k_3. \tag{6.2}$$

For springs in series, with individual spring rates, k_i, Fig. 6.3b, the spring rate k is

$$k = \frac{1}{\dfrac{1}{k_1} + \dfrac{1}{k_2} + \dfrac{1}{k_3}}. \tag{6.3}$$

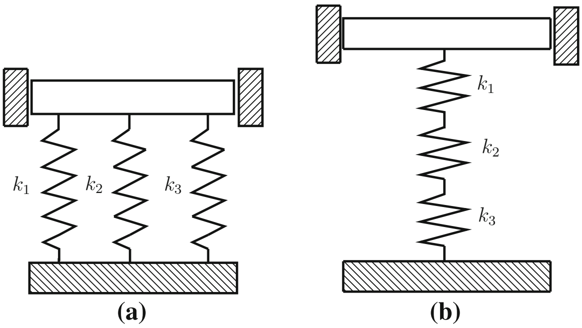


Fig. 6.3 Springs in **a** parallel and **b** series

6.3 Gravitational Potential Energy

Consider a body P of weight $G = mg$. The body is initially at P_1 and is moved down at the position P along an arbitrary path **I**, as shown in Fig. 6.4. The magnitude of the displacement in the **G** direction is $\mathrm{d}h = \mathrm{d}s \cos\alpha$, where $\mathrm{d}s$ is the displacement along the path. The displacement and the force are in the same direction and the work is $\mathrm{d}U = \mathbf{G} \cdot \mathrm{d}\mathbf{s}$. It results

$$U = \int_s G \cos\alpha \mathrm{d}s = \int_0^h mg\mathrm{d}y = mgh.$$

The work done by the weight **G** when the body moves up from P to position P_1 along the arbitrary path **II** is the negative work

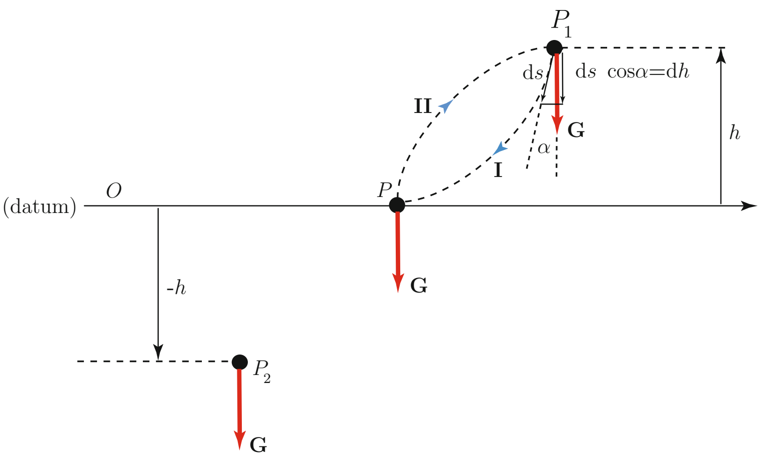


Fig. 6.4 Gravitational potential energy

$$U = -mgh.$$

The work done by the weight depends only on the vertical displacement h and is independent of the path of the body.

The gravitational potential energy V_g of a body is defined as the work done on the body by a force equal and opposite to the weight in bringing the body to the position under consideration from an arbitrary datum where the potential energy is defined to be zero. The potential energy is the negative of the work done by the weight. At the datum $h = 0$ and $V_g = 0$. At a height h above the datum plane, the gravitational potential energy of the body is $V_g = mgh$. If the body is a distance h below the datum the gravitational potential energy is $V_g = -mgh$.

Remarks:

1. the datum for zero potential energy is arbitrary because only the change in potential energy is considered;
2. the gravitational potential energy is independent of the path followed in arriving at a particular level h.

The virtual change in gravitational potential energy is

$$\delta V_g = mg\delta h,$$

where δh is the upward virtual displacement of the mass center of the body. The units of gravitational potential energy are the same as those for work and elastic potential energy, joules (J) in SI units and foot-pounds (ft·lb) in U.S. customary units.

Consider a linear spring attached to a body of mass m. The work done by the linear spring on the body is the negative of the change in elastic potential energy. The work done by the gravitational force is the negative of the change in gravitational potential energy.

The total virtual work δU is the sum of the work δU_a done by the active forces (other than spring forces and gravitational forces) and the work done by the spring forces and gravitational forces. For this case the virtual work equation $\delta U = 0$ is

$$\delta U_a - (\delta V_e + \delta V_g) = 0 \quad \text{or} \quad \delta U_a = \delta V, \tag{6.4}$$

where $V = V_e + V_g$ is the total potential energy of the system.

6.4 Stability of Equilibrium

For a system the sum of the work done by the active forces other than spring forces and gravitational forces is considered zero ($\delta U_a = 0$). With $\delta U_a = 0$ and Eq. (6.4) the relation of the virtual work is

$$\delta(V_e + V_g) = 0, \tag{6.5}$$

or

$$\delta V = 0. \tag{6.6}$$

Equation (6.6) expresses the principle of virtual work for conservative forces. Consider a mechanical system in equilibrium. If the forces that do work are conservative the total potential energy V of the system has a stationary value.

For a system with one degree of freedom the configuration is described by a single independent variable q. The equilibrium condition is $\delta V = 0$ or

$$\delta V = \frac{\mathrm{d}V}{\mathrm{d}q}\delta q = 0,$$

or

$$\frac{\mathrm{d}V}{\mathrm{d}q} = 0. \tag{6.7}$$

A mechanical system is in equilibrium when the derivative of its total potential energy is zero. For systems with multiple degrees of freedom, the partial derivative of V with respect to each independent coordinate q_i must be zero for equilibrium.

The following three cases are considered for Eq. (6.7), as shown in Fig. 6.5:

- the total potential energy is a minimum (stable equilibrium),
- the total potential energy is a maximum (unstable equilibrium),
- the total potential energy is constant (neutral equilibrium).

For a continuous function with continuous derivatives, the second derivative is positive at a minimum point of the function and negative at a maximum point of the function. The mathematical conditions for equilibrium and stability of the one degree of freedom system are:

$$\text{Equilibrium}: \frac{\mathrm{d}V}{\mathrm{d}q} = 0;$$
$$\text{stable}: \frac{\mathrm{d}^2V}{\mathrm{d}q^2} > 0;$$
$$\text{unstable}: \frac{\mathrm{d}^2V}{\mathrm{d}q^2} < 0.$$

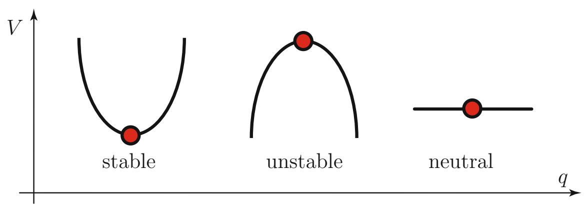


Fig. 6.5 Equilibrium positions

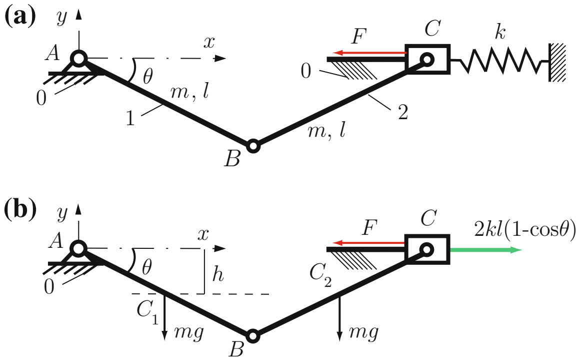


Fig. 6.6 Example 6.1

6.5 Examples

Example 6.1 In a vertical plane two uniform links, each of mass m and length l, are connected and constrained as shown in Fig. 6.6a. The spring is not stretched when the links are horizontal ($\theta = 0$). The angle θ increases with the application of the known horizontal force **F**. Determine the spring stiffness k which will produce equilibrium at a given angle θ. For the numerical values use $l = 5\,\text{m}$, $\theta = \pi/6\,\text{rad}$, $m = 1\,\text{kg}$, $F = 500\,\text{N}$, and $g = 9.81\,\text{m/s}^2$.

Solution The mechanical system has one degree of freedom. The position of each link can be expressed in terms of the angle θ. The MATLAB commands for the positions A, B, C, and D are:

```
syms l theta m F g k
rA_=[0,0,0];
xB=l*cos(theta);
yB=-l*sin(theta);
rB_=[xB,yB,0];
xC=2*l*cos(theta);
yC=0;
rC_=[xC,yC,0];
xD=2*l;
yD=0;
rD_=[xD,yD,0];
```

The point D represents the unstretched position of C when $\theta = 0$. The spring deflection is

$$x = 2\,l - 2\,l\cos\theta = 2\,l(1 - \cos\theta),$$

or with MATLAB:

```
xS=xD-xC;
```

The force diagram is shown in Fig. 6.6b. The joint forces are not included in the diagram. The elastic potential energy of the spring is

$$V_e = \frac{1}{2}kx^2 = 2\,kl^2\,(1 - \cos\theta)^2,$$

or with MATLAB:

```
Ve=1/2*k*xS^2;
```

The virtual change in elastic potential energy is

$$\delta V_e = \delta\left[2\,kl^2(1-\cos\theta)^2\right] = 2\,kl^2\delta(1-\cos\theta)^2 = 4\,kl^2(1-\cos\theta)\sin\theta\delta\theta,$$

or in MATLAB:

```
dVe=diff(Ve,theta);
```

The gravitational potential energy is

$$V_g = -2\,mgh = -2\,mg\left(\frac{l}{2}\sin\theta\right) = -mgl\sin\theta.$$

The datum for zero gravitational potential energy was taken through the support at A. The gravitational potential energy in MATLAB for the two links 1 and 2 is:

```
h1=l*sin(theta)/2;
h2=h1;
m1=m;
m2=m;
Vg=-m1*g*h1-m2*g*h2;
```

The virtual change in gravitational potential energy is

$$\delta V_g = \delta(-mgl\sin\theta) = -mgl\cos\theta\delta\theta$$

or with MATLAB:

```
dVg=diff(Vg,theta);
```

The virtual work done by the active external force **F** is

$$\delta U_a = F\delta = F\delta\,[2\,l(1-\cos\theta)] = 2\,Fl\delta(1-\cos\theta) = 2\,Fl\sin\theta\delta\theta.$$

```
dUa=F*diff(xS,theta);
```

The virtual work equation $\delta U_a = \delta V_e + \delta V_g$ gives

$$2\,Fl\sin\theta\delta\theta = 4\,kl^2(1-\cos\theta)\sin\theta\delta\theta - mgl\cos\theta\delta\theta.$$

The stiffness of the spring is

$$k = \frac{2\,F\sin\theta + mg\cos\theta}{4\,l(1-\cos\theta)\sin\theta}.$$

The MATLAB commands for the calculation of the stiffness of the spring are:

```
dW = dUa - (dVe + dVg);
ks = solve(dW, k);
```

The numerical value for the stiffness of the spring is calculated with:

```
lists = {l, theta, m, F, g};
listt = {5, pi/6, 1, 500, 9.81};
ks=subs(ks,lists,listt);
```

and the results is: k = 379.546 (N/m). The MATLAB figure of the system is shown in Fig. 6.7.

Example 6.2 Figure 6.8 shows a uniform bar of mass m and length l that moves in the vertical and horizontal directions. The spring has the stiffness k and is not compressed when the bar is vertical. Find the equilibrium positions and examine the stability. For the numerical values use $l = 5$ m, $\theta = \pi/6$ rad, $m = 1$ kg, $F = 500$ N, and $g = 9.81$ m/s^2.

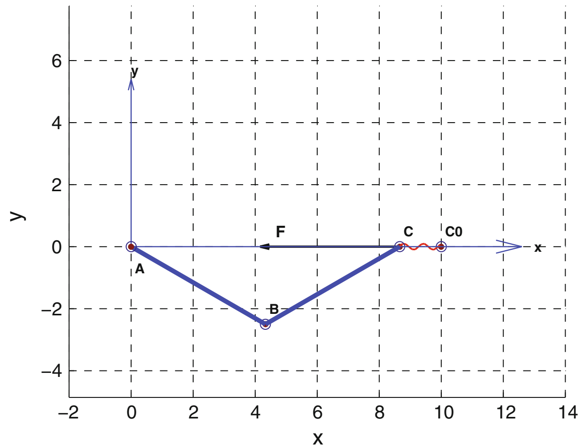


Fig. 6.7 Example 6.1 MATLAB figure of the system

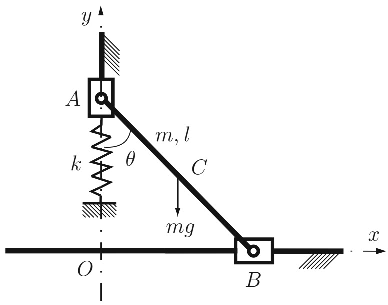


Fig. 6.8 Example 6.2

Solution The displacement of the bar shown in Fig. 6.8 can be expressed in term of the angle θ. The spring is undeformed when $\theta = 0$. The datum for zero gravitational potential energy is the horizontal x-axis. The positions of A, B, and the mass center C are given in MATLAB by:

```
xO=0;
yO=0;
rO_=[xO,yO,0];
xA=0;
yA=l*cos(theta);
rA_=[xA,yA,0];
xB=l*sin(theta);
yB=0;
rB_=[xB,yB,0];
xC=(xA+xB)/2;
yC=(yA+yB)/2;
rC_=[xC,yC,0];
```

The spring deflection is

$$y = l - l\cos\theta = l(1 - \cos\theta),$$

or with MATLAB:

```
yS=l-yA;
```

The elastic potential energy of the spring is

$$V_e = \frac{1}{2}ky^2 = \frac{1}{2}kl^2(1 - \cos\theta)^2,$$

The gravitational potential energy is

$$V_g = mgh = mg\left(\frac{l}{2}\cos\theta\right) = \frac{1}{2}mgl\cos\theta.$$

The total potential energy is

$$V = V_e + V_g = \frac{1}{2}kl^2(1-\cos\theta)^2 + \frac{1}{2}mgl\cos\theta.$$

The elastic potential energy, the gravitational potential energy, and the total potential energy with MATLAB are:

```
Ve=1/2*k*yS^2;
Vg= m*g*yC;
V=Ve+Vg;
```

The equilibrium position is obtained by differentiating the total potential energy and setting it to zero

$$\frac{dV}{d\theta} = kl^2(1-\cos\theta)\sin\theta - \frac{mgl\sin\theta}{2} = l\sin\theta\left[kl(1-\cos\theta) - \frac{mg}{2}\right] = 0.$$

The solutions to this equation are the equilibrium positions:

$$\sin\theta = 0 \quad \text{and} \quad \cos\theta = 1 - \frac{mg}{2kl}.$$

The MATLAB program for calculating the solutions is:

```
dVdtheta = diff(V,theta);
thetas=solve(dVdtheta, theta);
theta1=thetas(1);
theta2=thetas(2);
theta3=thetas(3);
```

and the MATLAB results are:

```
theta1 =
  0

theta2 =
            /   g m       \
  pi + acos| ----- - 1 |
            \ 2 k l       /

theta3 =
            /   g m       \
  pi - acos| ----- - 1 |
            \ 2 k l       /
```

For the numerical data:

```
list = {l,k,m,g};
listn = {5,100,10,9.8};
theta1n = subs(theta1,list,listn);
theta2n = subs(theta2,list,listn);
theta3n = subs(theta3,list,listn);
```

the solutions are:

```
theta1 =       0 (rad) =      0 (deg)
theta2 =    5.84 (rad) =    334 (deg)
theta3 =   0.446 (rad) =   25.6 (deg)
```

The sign of the second derivative of the potential energy for each of the two equilibrium positions will determine the stability of the system. The second derivative of the total potential energy is

$$\frac{d^2V}{d\theta^2} = kl^2 \sin^2\theta + kl^2(1-\cos\theta)\cos\theta - \frac{mgl\cos\theta}{2}.$$

Solution 1: $\sin\theta = 0$, $\theta = 0 \implies$

$$\frac{d^2V}{d\theta^2} = 0 + kl^2(1-1)(1) - \frac{mgl}{2} = -\frac{mgl}{2} < 0.$$

Equilibrium for $\theta = 0$ is never stable.

Solutions 2 and 3: $\cos\theta = 1 - \dfrac{mg}{2kl} \implies$

$$\frac{d^2V}{d\theta^2} = mg\left(l - \frac{mg}{4k}\right).$$

For $l > mg/(4k) \implies d^2V/d\theta^2 > 0$ the equilibrium position is stable.

For $l < mg/(4k) \implies d^2V/d\theta^2 < 0$ the equilibrium position is unstable. For the given numerical data it results:

```
for theta1 => d^2(V)/d(theta)^2 =   -245
theta1 is unstable equilibrium position

for theta2 => d^2(V)/d(theta)^2 =    466
theta2 is stable equilibrium position

for theta3 => d^2(V)/d(theta)^2 =    466
theta3 is stable equilibrium position
```

The MATLAB figure of the system and the three equilibrium positions A_1B_1, A_2B_2, and A_3B_3 are shown in Fig. 6.9.

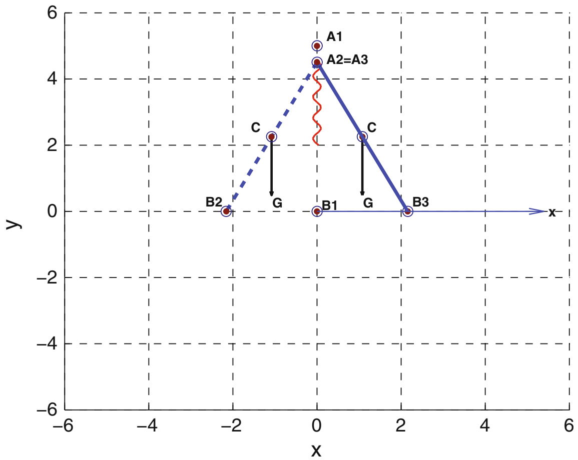


Fig. 6.9 Example 7.2 MATLAB figure of the system with equilibrium positions: $A_1 B_1$, $A_2 B_2$, and $A_3 B_3$

Example 6.3 A particle P of mass m is constrained to a helical trajectory as shown in the Fig. 6.10. The parametric equations of the particle P are given by

$$x = a\cos\theta,\ \ y = a\sin\theta,\ \ z = b\theta,$$

where a and b are constants and θ is the angle to the helix axes. The particle is attracted by the origin O with a force $F = kr_{PO}$. Find the equilibrium positions of the particle. For the numerical values use $a = 3\,\text{m}$, $b = 0.5\,\text{m}$, $k = 10\,\text{N/m}$, $m = 1\,\text{kg}$, and $g = 9.81\,\text{m/s}^2$.

Solution The force acting on the particle with respect to the force center O can be expressed as

$$F = -k\left(x\mathbf{\imath} + y\mathbf{\jmath} + z\mathbf{k}\right) = -ka\left(\cos\theta\mathbf{\imath} + \sin\theta\mathbf{\imath} + \frac{b}{a}\theta\mathbf{k}\right),$$

where $\mathbf{r} = x\mathbf{\imath} + y\mathbf{\jmath} + z\mathbf{k} = a\cos\theta\mathbf{\imath} + a\sin\theta\mathbf{\jmath} + b\theta\mathbf{k}$ represents the position vector of the particle. The sum of the forces acting on the particle are

$$\mathbf{R} = \mathbf{F} + \mathbf{G} = -ka\left(\cos\theta\mathbf{\imath} + \sin\theta\mathbf{\jmath}\right) + \left(-mg - kb\theta\right)\mathbf{k},$$

where $\mathbf{G}$ is the force of gravity. The curl of the vector force $\mathbf{R} = R_x\imath + R_y\jmath + R_z\mathbf{k}$ in cartesian coordinates is

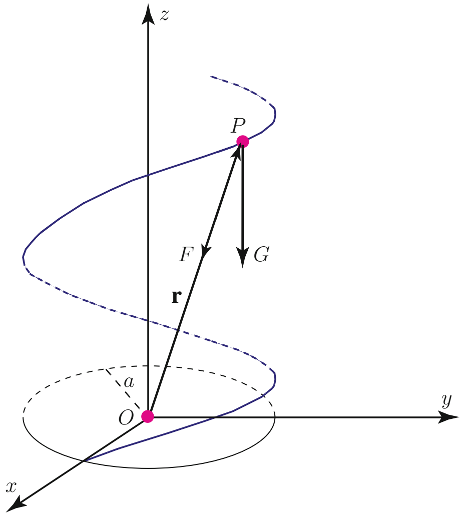


Fig. 6.10 Example 6.3

$$\nabla \times \mathbf{R} = \begin{vmatrix} \mathbf{\imath} & \mathbf{\jmath} & \mathbf{k} \\ \dfrac{\partial}{\partial x} & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial z} \\ R_x & R_y & R_z \end{vmatrix}.$$

The MATLAB calculations for the curl of **R** are:

```
syms x y z a b theta k m g
r_ = [x y z];
F_ = -k*r_;
G_ = [0 0 -m*g];
R_ = F_ + G_;
rotR = curl(R_,[x y z]);
```

and the results is:

```
curl(R_) =
  +-    -+
  |  0   |
  |  0   |
  |  0   |
  +-    -+
```

Because $\nabla \times \mathbf{R} = \mathbf{0}$ the force **R** is conservative.

The infinitesimal change in position d**r** is

$$\mathrm{d}\mathbf{r} = a\mathrm{d}(\cos\theta)\,\mathbf{\imath} + a\mathrm{d}(\sin\theta)\,\mathbf{\jmath} + b\mathrm{d}\theta\mathbf{k}.$$

The potential energy of the particle is

$$V = -\int \mathbf{R}\cdot \mathrm{d}\mathbf{r} = -\int \left[-ka^2\cos\theta \mathrm{d}\left(\cos\theta\right) - ka^2\sin\theta \mathrm{d}\left(\sin\theta\right) - mgb\mathrm{d}\theta - kb^2\theta\mathrm{d}\theta\right].$$

The partial derivative of the potential energy V with respect to θ is calculated as

$$\frac{\partial V}{\partial\theta} = -mgb - kb^2\theta.$$

The equilibrium position is calculated from

$$\frac{\partial V}{\partial\theta} = 0 \Rightarrow mgb + kb^2\theta = 0 \Rightarrow \theta b = -\frac{mg}{k}.$$

It results

$$z = \theta b = -\frac{mg}{k} < 0,$$

there is only one equilibrium position. Since

$$\frac{\partial^2 V}{\partial\theta^2} = kb^2 > 0,$$

the equilibrium position is stable. The MATLAB program for the equilibrium position is:

```
x = a*cos(theta);
y = a*sin(theta);
z = b*theta;
r_ = [x y z];
F_ = -k*r_;
R_ = F_+G_
dr_ = diff(r_,theta);
V = -int(R_*dr_.',theta);
dV = simplify(diff(V,theta));
thetae = solve(dV,theta);
d2V = diff(dV,theta);
```

For the numerical data the stable equilibrium position is calculated with:

```
list = {a, b, k, m, g};
listn= {3, 0.5, 10, 1, 9.8};
thetan=subs(thetae,list,listn);
liste = {a, b, k, m, g, theta};
listen = {3, 0.5, 10, 1, 9.8, thetan};
xe = subs(x,liste,listen);
ye = subs(y,liste,listen);
ze = subs(z,liste,listen);
```

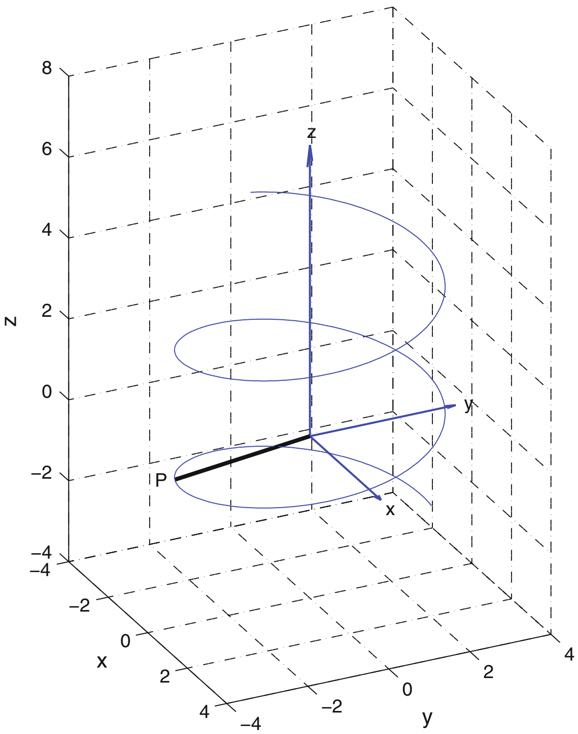


Fig. 6.11 Example 6.3 MATLAB figure for the equilibrium of particle P

and the results is:

```
equilibrium position
theta = -1.960 (rad) = -112.300 (deg)
x = -1.138 (m)
y = -2.776 (m)
z = -0.980 (m)
```

The MATLAB figure of the system is shown in Fig. 6.11.

Example 6.4 A particle P of mass m is constrained to slide without friction on the surface of a sphere of radius R as shown in the Fig. 6.12. The center of the sphere is located at the origin of a Cartesian reference frame xyz. The particle is attracted by a point located at $A(0, 0, R)$ with a force proportional to the distance between the particle and the point A: $F = kr_{PA}$. Find the equilibrium positions of the particle.

Solution The force between the particle and the attractive point A is denoted by $\mathbf{F}$ and the gravity is denoted by $\mathbf{G}$

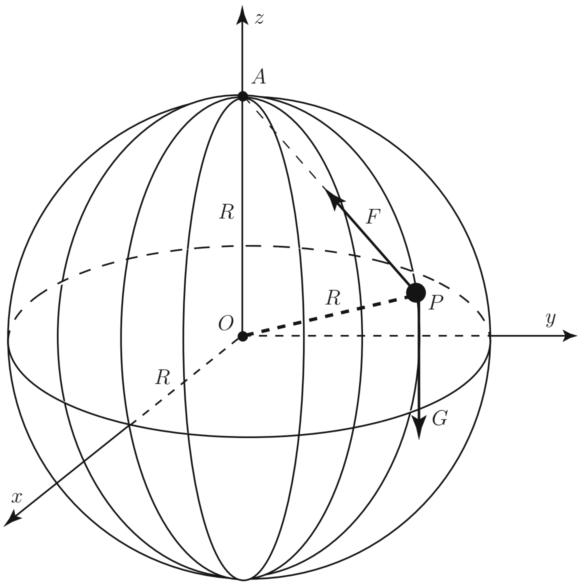


Fig. 6.12 Example 6.4

$$\mathbf{F} = k\mathbf{r}_{PA} = k[-x\mathbf{\imath} - y\mathbf{\jmath} + (R - z)\mathbf{k}],$$
$$\mathbf{G} = -mg\mathbf{k}.$$

The sum of the external forces can be written as

$$\mathbf{P} = \mathbf{F} + \mathbf{G} = -kx\mathbf{\imath} - ky\mathbf{\jmath} + (kR - kz - mg)\mathbf{k}.$$

The MATLAB expression for the forces on the particle is:

```
syms x y z R k m g C
rP_ = [x y z];
rA_ = [0 0 R];
rPA_ = rA_ - rP_;
F_ = k*rPA_;
G_ = [0 0 -m*g];
P_ = F_ + G_;
```

If curl $\mathbf{P} = \nabla \times \mathbf{P} = \mathbf{0}$ the force $\mathbf{P}$ is conservative. The expression $\nabla \times \mathbf{P}$ is calculated with MATLAB:

```
rotP_ = curl(P_,[x y z]);
```

and the results is:

```
curl(P_)=[0, 0, 0]
```

The potential energy is calculated with

$$V(x,y,z) = -\int \mathbf{P}\cdot \mathrm{d}\mathbf{r} + \mathrm{C} = -\int [-kx\mathrm{d}x - ky\mathrm{d}y + (kR - kz - mg)\mathrm{d}z] + \mathrm{C}$$

$$V(x,y,z) = k\frac{x^2}{2} + k\frac{y^2}{2} + k\frac{(R-z)^2}{2} + mgz + \mathrm{C}, \tag{6.8}$$

where C is an arbitrary constant known as the constant of integration.

The equation of the sphere can be written as

$$x^2 + y^2 + z^2 = R^2 \quad \text{or} \quad z = \sqrt{R^2 - x^2 - y^2}, \tag{6.9}$$

and with Eq. (6.8) the potential energy function of x and y is calculated in MATLAB with:

```
V1=-int(P_(1));
V2=-int(P_(2));
V3=-int(P_(3));
V=V1+V2+V3+C;
fVxy=subs(V,z,sqrt(R^2-x^2-y^2));
Vxy=simple(simplify(Vxy));
```

and $V(x, y)$ is obtained as

$$V(x,y) = kR^2 + (mg - kR)\sqrt{R^2 - x^2 - y^2} + C.$$

The partial derivative of the function $V(x, y)$ with respect to x and y are

$$\frac{\partial V}{\partial x} = -(mg - kR)\frac{x}{\sqrt{R^2 - x^2 - y^2}},$$
$$\frac{\partial V}{\partial y} = -(mg - kR)\frac{y}{\sqrt{R^2 - x^2 - y^2}}.$$

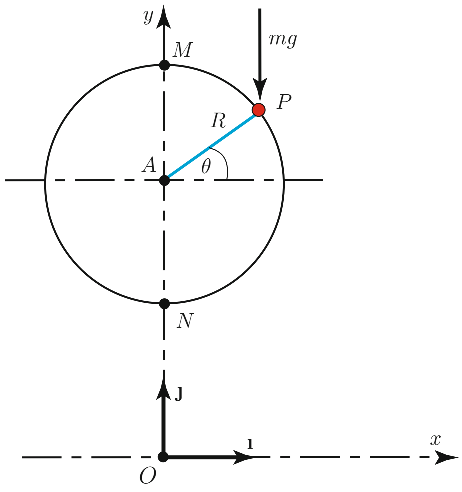
The equilibrium positions of the particle are obtained from

$$\frac{\partial V}{\partial x} = 0,$$
$$\frac{\partial V}{\partial y} = 0.$$

In MATLAB the equilibrium positions are obtained with:

```
dVxydx = simple(diff(Vxy,x));
dVxydy = simple(diff(Vxy,y));
xe=solve(dVxydx,x);
```

Fig. 6.13 Example 6.5




```
ye=solve(dVxydy,y);
ze=solve(xe^2+ye^2+z^2-R^2,z);
```

The results for the equilibrium positions are $M_1(0, 0, R)$ and $M_2(0, 0, -R)$.

Example 6.5 A particle P of mass m is on a circle of radius R as shown in the Fig. 6.13. The circle is on a vertical plane xy. Find the equilibrium positions of the particle.

Solution The independent variable is the angle θ. The position of the particle P is

```
x = R*cos(theta);
y = yN+R+R*sin(theta);
r_ = [x y];
```

where `yN` is the y coordinate of the lower end N of the circle. The gravity is the only force acting on the particle and the potential energy is calculated with:

```
dr_=diff(r_,theta);
G_ = [0 -m*g];
V = -int(G_*dr_.');
fprintf('V=%s + C\n', char(V))
```

The MATLAB expression for the potential energy is:

```
V=R*g*m*sin(theta) + C
```

where `C` is a constant of integration. The equilibrium positions are calculated from the equation:

```
dV = diff(V,theta);
thetae=solve(dV,theta);
theta1=thetae;
```

```
theta2=theta1+pi;
```

The equilibrium position are the points M and N as shown in Fig. 6.13: theta1 = pi/2 and theta2 = (3 * pi)/2. The equilibrium stability is verified with the second derivative of the potential energy:

```
d2V = diff(dV,theta);
d2V1=subs(d2V,theta,theta1);
d2V2=subs(d2V,theta,theta2);
```

and the MATLAB results are:

```
d2V/d(theta)^2=-R*g*m*sin(theta)
for theta1 => d2V/d(theta)^2=-R*g*m
for theta2 => d2V/d(theta)^2=R*g*m
```

The equilibrium position $\theta = 3\pi/2$, position N, is a stable equilibrium because $\mathrm{d}^2\mathrm{V}/\mathrm{d}\theta^2 = Rgm$ is positive.

6.6 Problems

6.1 Two bars, 1 and 2, each of mass m and length l are connected and constrained as shown in Fig. 6.14. The angle θ is between the link 1 and the vertical axes. The spring of stiffness k is not stretched in the position where $\theta = 0$. Find the force F which will produce equilibrium at the angle θ.

6.2 Figure 6.15 shows a mechanism with two links, 1 and 2. Link 1 has the mass $m_1 = m$ and the length $l_1 = l$. Link 2 has the mass $m_2 = 2m$ and the length $l_2 = 2l$. The spring is unstretched in the position $\theta = 0$. A known vertical force F is applied on link 2 at D. Determine the spring stiffness k which will establish an equilibrium at a given angle θ.

6.3 For the mechanism shown in Fig. 6.16, link 1 has the mass $m_1 = 2m$ and the length $l_1 = 2l$. The link 2 has the mass $m_2 = m$ and the length $l_2 = l$. The

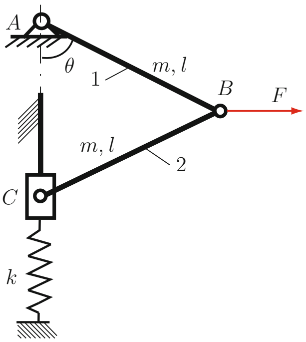


Fig. 6.14 Problem 6.1

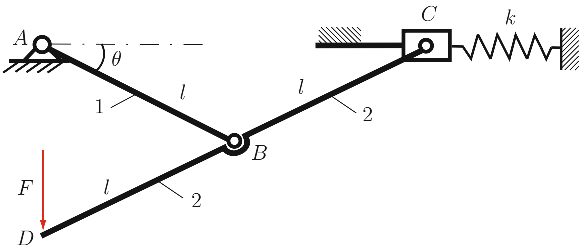
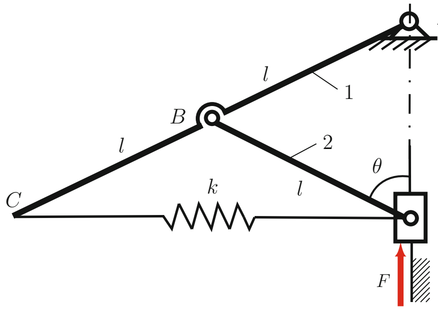


Fig. 6.15 Problem 6.2

Fig. 6.16 Problem 6.3



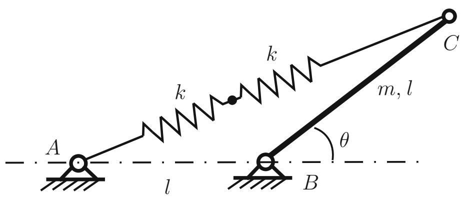
spring has an unstretched length of L_0. Determine the spring stiffness k for an equilibrium at a given angle θ and a given force F.

6.4 The link *BC* shown in Fig. 6.17 has a mass m and is connected to two springs ($AB = BC = l$). Each spring has the stiffness k and the unstretched length of the two springs is L_0. Determine the spring stiffness k which will establish an equilibrium at a given angle θ. Use the following numerical application: $l = L_0 = 300$ mm, $m = 10$ kg, and $\theta = 60°$.

6.5 The mechanism shown in Fig. 6.18, has the link *BC* with the mass m and the length l ($AB = AC = l/2$). The spring has the stiffness k and is unstretched when $\theta = 0$. Find the equilibrium value for the coordinate θ. Use the following numerical application: $l = 400$ mm, $m = 10$ kg, $F = 70$ N, and $k = 1.8$ kN/m.

6.6 The link of mass m and length l is connected to two identical horizontal springs, each of stiffness k, as shown in Fig. 6.19. The initial spring compression at $\theta = 0$ is d. For a stable equilibrium position at $\theta = 0$ find the minimum value of k.

Fig. 6.17 Problem 6.4



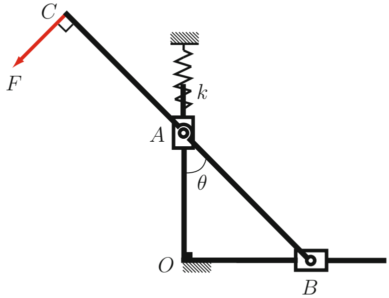


Fig. 6.18 Problem 6.5

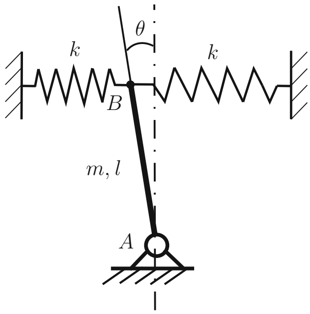


Fig. 6.19 Problem 6.6

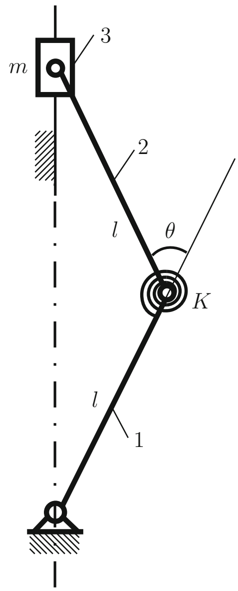
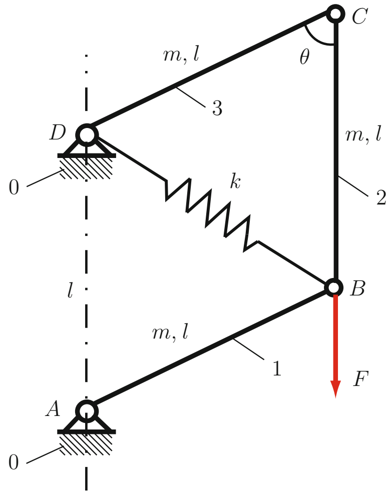


Fig. 6.20 Problem 6.7

6.7 The mechanism shown in Fig. 6.20 has two identical links, 1 and 2, each of length l and negligible mass compared with the mass m of the slider 3. The two light links have a torsion spring at their common joint. The moment developed

Fig. 6.21 Problem 6.8



by the torsion spring is $M = K\,\theta$, where θ is the relative angle between the links at
the joint. Determine the minimum value of K which will ensure the stability of the mechanism for $\theta = 0$.

6.8 Figure 6.21 shows a four-bar mechanism with $AD = l$. Each of the links has the mass m ($m_1 = m_2 = m_3 = m$) and the lenght l ($l_1 = l_2 = l_3 = l$). At B a vertical force F acts on the mechanism and the spring stiffness is k. The motion is in the vertical plane. Find the equilibrium angle θ. Use the following numerical application: $l = 15$ in, $m = 10$ lb, $F = 90$ lb, and $k = 15$ lb/in. Select an unextended (initial) length L_0 for the spring.

6.9 A particle of mass m can move freely in space. The potential energy V of the particle at $x = l$, when the particle is subject to a vertical force $F = ax^2 + bx + c$, is $V = s$. Find the equilibrium positions of the particle. For the numerical application use $a = 1$, $b = -3$, $c = 0$, $l = 0$ m, and $s = 5$ J.

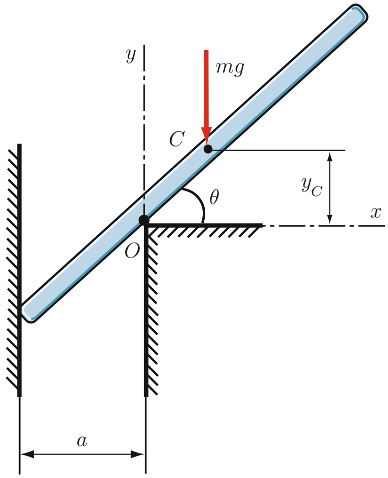
6.10 A bar of mass m and length l is supported by a vertical wall and a point at O, as shown in Fig. 6.22. Find the equilibrium positions of the bar. For the numerical application use $l = 0.5$ m, $a = 0.1$ m, $m = 1$ kg, and $g = 9.81$ m/s^2.

6.7 Programs

6.7.1 Program 6.1

```
% example 6.1

clear all; clc; close all
```

Fig. 6.22 Problem 6.10




```
syms l theta m F g k

rA_=[0,0,0];
xB=l*cos(theta);
yB=-l*sin(theta);
rB_=[xB,yB,0];
xC=2*l*cos(theta);
yC=0;
rC_=[xC,yC,0];
xD=2*l;
yD=0;
rD_=[xD,yD,0];

xS=xD-xC;
fprintf('spring deflection:\n')
fprintf('xS = x - xS0 = %s \n\n',char(xS))

Ve=1/2*k*xS^2;
fprintf('elastic potential energy:\n')
fprintf('Ve = %s \n\n',char(simple(Ve)))

dVe=diff(Ve,theta);
fprintf('dVe/d(theta)\n')
fprintf('= %s \n\n',char(simple(dVe)))

h1=l*sin(theta)/2;
h2=h1;
```

```
m1=m;
m2=m;
Vg=-m1*g*h1-m2*g*h2;

fprintf('gravitational potential energy:\n')
fprintf('Vg = %s \n\n',char(simple(Vg)))

dVg=diff(Vg,theta);
fprintf('dVg/d(theta)\n')
fprintf('= %s \n\n',char(simple(dVg)))

dUa=F*diff(xS,theta);
fprintf('F*dxS/d(theta)\n')
fprintf('= %s \n\n',char(simple(dUa)))

dW = dUa - (dVe + dVg);
fprintf('virtual work equation: 0 = \n')
pretty(dW)
fprintf('\n')

ks = solve(dW, k);
fprintf('spring stiffness: k = \n')
pretty(ks)
fprintf('\n')

lists = {l, theta, m, F, g};
listt = {5, pi/6, 1, 500, 9.81};

ks=subs(ks,lists,listt);
fprintf('k = %6.3f (N/m)\n',ks);

% plot of the mechanical system
xA=0; yA=0;
xB=subs(xB,lists,listt);
yB=subs(yB,lists,listt);
xC=subs(xC,lists,listt);
yC=subs(yC,lists,listt);
xD=subs(xD,lists,listt);
yD=subs(yD,lists,listt);
F=subs(F,lists,listt);

axis([-2 14 -2 6])
grid on, hold on
axis equal
xlabel('x'), ylabel('y')
```

```
line([0,xB],[0,yB],...
'LineStyle','-','Color','b','LineWidth',4)
line([xB,xC],[yB,yC],...
'LineStyle','-','Color','b','LineWidth',4)
quiver(xC,yC,-F/100,0,...
'Color','k','LineWidth',2.0);
text(xC-F/100+1,yC+0.5,'F',...
'fontsize',14,'fontweight','b');

ax=14; ay=6;
quiver(0,0,ax,0,...
'Color','b','LineWidth',1.0);
text(ax-1,0,'x',...
'fontsize',12,'fontweight','b');
quiver(0,0,0,ay,...
'Color','b','LineWidth',1.0);
text(0,ay-.3,'y',...
'fontsize',12,'fontweight','b');

text(xA,yA-0.7,' A',...
'fontsize',12,'fontweight','b');
text(xB,yB+0.5,' B',...
'fontsize',12,'fontweight','b');
text(xC,yC+0.5,' C',...
'fontsize',12,'fontweight','b');
text(xD,yD+0.5,' C0',...
'fontsize',12,'fontweight','b');

scatter(xA,yA,80,2)
scatter(xA,yA,30,50,'filled')
scatter(xB,yB,80,2)
scatter(xB,yB,30,50,'filled')


scatter(xC,yC,80,2)
scatter(xC,yC,30,50,'filled')
scatter(xD,yD,80,2)
scatter(xD,yD,30,50,'filled')

t = xC:0.1:xD; % spring plot
n=10;
plot3(t,cos(n*t)/n,sin(n*t)/n,...
    'LineStyle','-','Color','r','LineWidth',1.5);

% end of program
```

6.7.2 Program 6.2

```
% example 6.2

clear all; clc; close all

syms l theta k m g

xO=0;
yO=0;
rO_=[xO,yO,0];
xA=0;
yA=l*cos(theta);
rA_=[xA,yA,0];
xB=l*sin(theta);
yB=0;
rB_=[xB,yB,0];
xC=(xA+xB)/2;
yC=(yA+yB)/2;
rC_=[xC,yC,0];

yS=l-yA;
fprintf('spring deflection:\n')
fprintf('yS= %s \n\n',char(yS))

Ve=1/2*k*yS^2;
fprintf('elastic potential energy:\n')
fprintf('Ve= %s \n\n',char(simple(Ve)))

Vg= m*g*yC;
fprintf('gravitational potential energy:\n')
fprintf('Vg= %s \n\n',char(simple(Vg)))

V=Ve+Vg;
fprintf('total potential energy V=Ve+Vg\n')
pretty(V)
fprintf('\n')

dVdtheta = diff(V,theta);
fprintf('d(V)/d(theta):\n\n')
pretty(simple(dVdtheta))
fprintf('\n')

fprintf('d(V)/d(dtheta)=0 =>\n')
```

```
fprintf('\n')

thetas=solve(dVdtheta, theta);
theta1=thetas(1);
theta2=thetas(2);
theta3=thetas(3);


fprintf('theta1 = \n')
pretty(theta1)
fprintf('\n')

fprintf('theta2 = \n')
pretty(theta2)
fprintf('\n')

fprintf('theta3 = \n')
pretty(theta3)
fprintf('\n\n')

list = {l,k,m,g};
listn = {5,100,10,9.8};

theta1n = subs(theta1,list,listn);
theta2n = subs(theta2,list,listn);
theta3n = subs(theta3,list,listn);

fprintf('theta1 = %6.3g (rad) = ',theta1n);
fprintf('%6.3g (deg)\n',theta1n*180/pi);

fprintf('theta2 = %6.3g (rad) = ',theta2n);
fprintf('%6.3g (deg)\n',theta2n*180/pi);

fprintf('theta3 = %6.3g (rad) = ',theta3n);
fprintf('%6.3g (deg)\n',theta3n*180/pi);
fprintf('\n\n')

d2V = diff(dVdtheta,theta);
d2V = simplify(d2V);
fprintf('d^2(V)/d(theta)^2 = \n')
pretty(d2V)
fprintf('\n\n')

d2V1 = subs(d2V,theta,theta1);
d2V1 = simplify(d2V1);
```

```
fprintf('for theta1 => d^2(V)/d(theta)^2 = \n')
pretty(d2V1)
fprintf('\n\n')

d2V2 = subs(d2V,theta,theta2);
d2V2 = simplify(d2V2);
fprintf('for theta2 => d^2(V)/d(theta)^2 = \n')
pretty(d2V2)
fprintf('\n\n')

d2V3 = subs(d2V,theta,theta3);
d2V3 = simplify(d2V3);
fprintf('for theta3 => d^2(V)/d(theta)^2 = \n')
pretty(d2V3)
fprintf('\n\n')

d2V1n = subs(d2V1,list,listn);
d2V2n = subs(d2V2,list,listn);
d2V3n = subs(d2V3,list,listn);

fprintf('for theta1 => d^2(V)/d(theta)^2 = ')
fprintf('%6.3g \n',d2V1n);
if d2V1n > 0
fprintf('theta1 is stable equilibrium position\n')
else
fprintf('theta1 is unstable equilibrium position\n')
end

fprintf('for theta2 => d^2(V)/d(theta)^2 = ')
fprintf('%6.3g \n',d2V2n);
if d2V2n > 0
fprintf('theta2 is stable equilibrium position\n')
else
fprintf('theta2 is unstable equilibrium position\n')
end

fprintf('for theta3 => d^2(V)/d(theta)^2 = ')
fprintf('%6.3g \n',d2V3n);
if d2V3n > 0
fprintf('theta3 is stable equilibrium position\n')
else
fprintf('theta3 is unstable equilibrium position\n')
end

lists = {l,'theta'};
```

```
list1 = {5,theta1n};
xA1=subs(xA,lists,list1);
yA1=subs(yA,lists,list1);
xB1=subs(xB,lists,list1);
yB1=subs(yB,lists,list1);
xC1=subs(xC,lists,list1);
yC1=subs(yC,lists,list1);

list2 = {5,theta2n};
xA2=subs(xA,lists,list2);
yA2=subs(yA,lists,list2);
xB2=subs(xB,lists,list2);
yB2=subs(yB,lists,list2);
xC2=subs(xC,lists,list2);
yC2=subs(yC,lists,list2);

list3 = {5,theta3n};
xA3=subs(xA,lists,list3);
yA3=subs(yA,lists,list3);
xB3=subs(xB,lists,list3);
yB3=subs(yB,lists,list3);
xC3=subs(xC,lists,list3);
yC3=subs(yC,lists,list3);

as=6;
axis([-as as -as as])
grid on, hold on

line([xA2 xB2],[yA2 yB2],...
'LineStyle','--','Color','b','LineWidth',3)
line([xA3 xB3],[yA3 yB3],...
'LineStyle','-','Color','b','LineWidth',3)

xlabel('x'), ylabel('y')

quiver(0,0,as,0,...
'Color','b','LineWidth',1.0);
text(as-0.5,0,'x',...
'fontsize',12,'fontweight','b');

text(xA1+0.1,yA1+0.3,' A1',...
'fontsize',12,'fontweight','b');

text(xA2+0.1,yA2+0.1,' A2=A3',...
```

```
'fontsize',12,'fontweight','b');

text(xB2-0.6,yB2+0.3,' B2',...
'fontsize',12,'fontweight','b');
text(xC2-0.6,yC2+0.3,' C',...
'fontsize',12,'fontweight','b');

text(xB3,yB3+0.3,' B3',...
'fontsize',12,'fontweight','b');
text(xC3,yC3+0.3,' C',...
'fontsize',12,'fontweight','b');

G=subs(-m*g,list,listn);
quiver(xC3,yC3,0,G/50,...
'Color','k','LineWidth',2.0);
text(xC3,yC3+G/50,'G',...
'fontsize',14,'fontweight','b');

quiver(xC2,yC2,0,G/50,...
'Color','k','LineWidth',2.0);
text(xC2,yC2+G/50,'G',...
'fontsize',14,'fontweight','b');

text(0,0.2,' B1',...
'fontsize',12,'fontweight','b');

scatter(xA1,yA1,80,2)
scatter(xA1,yA1,30,50,'filled')
scatter(xB1,yB1,80,2)
scatter(xB1,yB1,30,50,'filled')


scatter(xA2,yA2,80,2)
scatter(xA2,yA2,30,50,'filled')

scatter(xB2,yB2,80,2)
scatter(xB2,yB2,30,50,'filled')
scatter(xC2,yC2,80,2)
scatter(xC2,yC2,30,50,'filled')
scatter(xB3,yB3,80,2)
scatter(xB3,yB3,30,50,'filled')
scatter(xC3,yC3,80,2)
scatter(xC3,yC3,30,50,'filled')

t = 2:0.1:yA2; % spring plot
```

```
n=10;
plot3(cos(n*t)/n,t,sin(n*t)/n,...
'LineStyle','-','Color','r','LineWidth',1.5);

% end of program
```

6.7.3 Program 6.3

```
% example 6.3

clear all; clc; close all

syms x y z a b theta k m g

r_ = [x y z];
F_ = -k*r_;
% force of gravity
G_ = [0 0 -m*g];
% total force
R_ = F_ + G_;
rotR = curl(R_,[x y z]);
fprintf('curl(R_) = \n')
pretty(rotR)
% curl(R_)=0 => R_ conservative

x = a*cos(theta);
y = a*sin(theta);
z = b*theta;
r_ = [x y z];
F_ = -k*r_;
R_ = F_+G_

dr_ = diff(r_,theta);

V = -int(R_*dr_.',theta);
fprintf('potential energy V = \n')
pretty(V)
fprintf('\n')

dV = simplify(diff(V,theta));
fprintf('d(V)/d(theta)=%s\n',char(simple(dV)))
fprintf('\n')
```

```
fprintf('d(V)/d(dtheta)=0 =>\n')
fprintf('\n')
thetae = solve(dV,theta);
fprintf('theta=%s\n',char(simple(thetae)))
fprintf('\n')

d2V = diff(dV,theta);
fprintf('d^2(V)/d(dtheta)^2=%s\n',char(d2V))
fprintf('\n')

% d^2(V)/d(dtheta)^2>0 => stable equilibrium

list = {a, b, k, m, g};
listn= {3, 0.5, 10, 1, 9.8};

fprintf('equilibrium position \n')
thetan=subs(thetae,list,listn);
fprintf ...
('theta = %6.3f (rad) = %6.3f (deg)\n',...
thetan,thetan*180/pi);


liste = {a, b, k, m, g, theta};
listen = {3, 0.5, 10, 1, 9.8, thetan};

xe = subs(x,liste,listen);
ye = subs(y,liste,listen);
ze = subs(z,liste,listen);

fprintf('x = %6.3f (m)\n',xe);
fprintf('y = %6.3f (m)\n',ye);
fprintf('z = %6.3f (m)\n',ze);

aa=4;
axis manual
axis equal
axis([-aa aa -aa aa -aa 2*aa])
grid on
az = 64;el = 26;
view(az, el);
hold on

quiver3(0,0,0,aa,0,0,...
'Color','b','LineWidth',1.0);
text(aa,0,' x','fontsize',12);
```

```
quiver3(0,0,0,0,aa,0,...
'Color','b','LineWidth',1.0);
text(0,aa,' y','fontsize',12);

quiver3(0,0,0,0,0,2*aa,...
'Color','b','LineWidth',1.0);
text(0,0,2*aa,' z','fontsize',12);

start_value=-3*pi/2;
end_value=3*pi;
step=pi/100;
a=3;b=0.5;
theta = start_value:step:end_value;
plot3(a*cos(theta),a*sin(theta),b*theta)
xlabel('x')
ylabel('y')
zlabel('z')

line([0 xe],[0 ye],[0 ze],...
'Color','k','LineWidth',2);
text(xe-.3,ye-.3,ze-.3,'P','fontsize',12);

% end of program
```

6.7.4 Program 6.4

```
% example 6.4

clear all; clc; close all

syms x y z R k m g C

rP_ = [x y z];
rA_ = [0 0 R];
rPA_ = rA_ - rP_;

F_ = k*rPA_;
fprintf('atractive force acting on particle\n');
fprintf('F_=[%s, %s, %s] \n\n',...
    char(F_(1)),char(F_(2)),char(F_(3)));

G_ = [0 0 -m*g];
```

```
fprintf('force of gravity on particle\n');
fprintf('G_=[0 0 %s] \n\n',char(G_(3)));

P_ = F_ + G_;
fprintf('sum of forces on particle\n');
fprintf('P_=[%s, %s, %s] \n\n',...
    char(P_(1)),char(P_(2)),char(P_(3)));

rotP_ = curl(P_,[x y z]);
fprintf('curl(P_)=[%s, %s, %s] \n\n',...
char(rotP_(1)),char(rotP_(2)),char(rotP_(3)));
% curl(P_)=0 => P_ conservative

V1=-int(P_(1));
V2=-int(P_(2));
V3=-int(P_(3));

V=V1+V2+V3+C;
fprintf('potential energy V(x,y,z)=')
pretty(V)
fprintf('\n')

Vxy=subs(V,z,sqrt(R^2-x^2-y^2));
Vxy=simplify(Vxy);
fprintf('with z=sqrt(R^2-x^2-y^2)=>V(x,y)=')
pretty(Vxy)
fprintf('\n\n')

dVxydx = simple(diff(Vxy,x));
fprintf('dV(x,y)/dx=')
pretty(dVxydx)
fprintf('\n\n')

dVxydy = simple(diff(Vxy,y));
fprintf('dV(x,y)/dy=')
pretty(dVxydy)
fprintf('\n\n')

xe=solve(dVxydx,x);
ye=solve(dVxydy,y);
ze=solve(xe^2+ye^2+z^2-R^2,z);

fprintf('equilibrium positions =>\n\n');
fprintf('M1(%s ,%s ,%s)  \n\n',...
    char(xe),char(ye),char(ze(1)));
```

```
fprintf('M2(%s ,%s ,%s)  \n\n',...
    char(xe),char(ye),char(ze(2)));

% end of program
```

6.7.5 Program 6.5

```
% example 6.5
clear all; clc; close all

syms R theta yN m g C

% position of the particle
x = R*cos(theta);
y = yN+R+R*sin(theta);
r_ = [x y];

fprintf('r_=[%s,%s]\n',char(x),char(y))

dr_=diff(r_,theta);
fprintf('dr_=[%s,%s] d(theta)\n',...
    char(dr_(1)),char(dr_(2)))

% gravity force on the particle
G_ = [0 -m*g];

% potential energy
V = -int(G_*dr_.');
fprintf('V=%s + C\n', char(V))
% C constant of integration

dV = diff(V,theta);
fprintf('dV/d(theta)=%s\n', char(dV))

thetae=solve(dV,theta);
theta1=thetae;
theta2=theta1+pi;

% equilibrium positions
fprintf('theta1=%s\n', char(theta1))
fprintf('theta2=%s\n', char(theta2))

d2V = diff(dV,theta);
```

```
fprintf('d2V/d(theta)^2=%s\n', char(d2V))

d2V1=subs(d2V,theta,theta1);
d2V2=subs(d2V,theta,theta2);

fprintf...
('for theta1 => d2V/d(theta)^2=%s\n',char(d2V1))
fprintf...
('for theta2 => d2V/d(theta)^2=%s\n',char(d2V2))

% end o program
```

References

1. P. Appell, *Traité de Mécanique Rationnelle* (Gauthier-Villars, Paris, 1955)
2. M. Atanasiu, *Mechanics* (EDP, Bucharest, 1973)
3. S. Attaway, *MATLAB: A Practical Introduction to Programming and Problem Solving* (Butterworth-Heinemann, Elsevier, Amsterdam, 2012)
4. H. Baruh, *Analytical Dynamics* (WCB/McGraw-Hill, Boston, 1999)
5. A.M. Bedford, W. Fowler, *Engineering Mechanics: Statics*, 5th edn. (Prentice Hall, Inc., Upper Saddle River, 2007)
6. A.M. Bedford, W. Fowler, K.M. Liechti, *Statics and Mechanics of Materials* (Prentice Hall, Inc., Upper Saddle River, 2002)
7. F.P. Beer, E.R. Johnston, *Vector Mechanics for Engineers: Statics and Dynamics*, 5th edn. (McGraw-Hill Publishing Company, New York, 1988)
8. F.P. Beer, E.R. Johnston, D.F. Mazurek, *Vector Mechanics for Engineers: Statics*, 10th edn. (McGraw-Hill Publishing Company, New York, 2012)
9. A.P. Boresi, R.J. Schmidt, *Engineering Mechanics: Statics* (PWS Publishing Company, Boston, 2000)
10. M.I. Buculei, *Mechanics* (University of Craiova Press, Craiova, 1974)
11. V. Ceausu, N. Enescu, F. Ceausu, *Mechanics Problems* (Printech, Bucuresti, 1999)
12. S.J. Chapman, *MATLAB Programming for Engineers* (Thomson Learning, Pacific Grove, CA, 2002)
13. D.M. Etter, D.C. Kuncicky, *Introduction to MATLAB for Engineers and Scientists* (Prentice Hall, Inc., Upper Saddle River, 1996)
14. J.H. Ginsberg, *Advanced Engineering Dynamics* (Cambridge University Press, Cambridge, 1995)
15. L.E. Goodman, W.H. Warner, *Statics* (Dover Publications, Inc., Mineola, 2001)
16. D.T. Greenwood, *Principles of Dynamics* (Prentice-Hall, Inc., Englewood Cliffs, 1998)
17. R.C. Hibbeler, *Engineering Mechanics: Statics and Dynamics*, 13th edn. (Prentice-Hall, Inc., Upper Saddle River, 2013)
18. C. Iacob, *Theoretical Mechanics* (EDP, Bucharest, 1980)
19. J.H. Jackson, H.G. Wirtz, *Schaum's Outline of Theory and Problems of Statics and Strength of Materials* (McGraw-Hill Publishing Company, New York, 1983)
20. T.R. Kane, *Analytical Elements of Mechanics*, vol. 1 (Academic Press, New York, 1959)
21. T.R. Kane, *Analytical Elements of Mechanics*, vol. 2 (Academic Press, New York, 1961)
22. T.R. Kane, D.A. Levinson, *Dynamics* (McGraw-Hill, New York, 1985)

23. R. Maeder, *Programming in Mathematica* (Addison-Wesley Publishing Company, Redwood City, 1990)
24. D.B. Marghitu, Statics and dynamics, class notes (2004), http://www.eng.auburn.edu/users/marghitu/
25. D.B. Marghitu, *Mechanical Engineer's Handbook* (Academic Press, San Diego, 2001)
26. D.B. Marghitu, *Kinematic Chains and Machine Component Design* (Elsevier, Amsterdam, 2005)
27. D.B. Marghitu, *Mechanisms and Robots Analysis with MATLAB* (Springer, New York, 2009)
28. D.B. Marghitu, M. Dupac, *Advanced Dynamics: Analytical and Numerical Calculations with MATLAB* (Springer, New York, 2012)
29. D.J. McGill, W.W. King, *Engineering Mechanics: Statics and an Introduction to Dynamics* (PWS Publishing Company, Boston, 1995)
30. J.L. Meriam, L.G. Kraige, *Engineering Mechanics: Statics*, 7th edn. (John Wiley & Sons, New York, 2011)
31. L.A. Pars, *A Treatise on Analytical Dynamics* (John Wiley & Sons, New York, 1965)
32. M. Plesha, G. Gray, F. Costanzo, *Engineering Mechanics: Statics*, 2nd edn. (McGraw-Hill Publishing Company, New York, 2012)
33. M. Radoi, E. Deciu, *Mechanics* (EDP, Bucharest, 1981)
34. W.F. Riley, L.D. Sturges, *Engineering Mechanics: Statics*, 2nd edn. (John Wiley & Sons, Inc., New York, 1995)
35. A. Ripianu, P. Popescu, B. Balan, *Technical Mechanics* (EDP, Bucharest, 1979)
36. A. Ruina, R. Pratap, *Introduction to Statics and Dynamics* (Oxford University Press, New York, 2002)
37. I.H. Shames, *Engineering Mechanics Statics*, 4th edn. (Prentice Hall, Inc., Englewood Cliffs, 1996)
38. S.D. Sheppard, B.H. Tongue, *Statics: Analysis and Design of Systems in Equilibrium* (John Wiley & Sons, New York, 2005)
39. D. Smith, *Engineering Computation with MATLAB* (Pearson Education, Upper Saddle River, 2008)
40. R.W. Soutas-Little, D.J. Inman, *Engineering Mechanics: Statics and Dynamics* (Prentice-Hall, Upper Saddle River, 1999)
41. R.W. Soutas-Little, D.J. Inman, D. Balint, *Engineering Mechanics: Statics* (Cengage Learning, Boston, 2007)
42. S. Staicu, *Theoretical Mechanics* (EDP, Bucharest, 1998)
43. A. Stan, M. Grumazescu, *Mechanics Problems* (EDP, Bucharest, 1973)
44. Statics eBook, https://ecourses.ou.edu/
45. J. Sticklen, M.T. Eskil, *An Introduction to Technical Problem Solving with MATLAB* (Great Lakes Press, Wildwood, 2006)
46. A. Stoenescu, G. Silas, *Theoretical Mechanics* (ET, Bucharest, 1957)
47. I. Stroe et al., *Analytical Mechanics Problems* (University Politehnica of Bucharest, Bucharest, 1997)
48. The MathWorks, https://www.mathworks.com/
49. V. Valcovici, S. Balan, R. Voinea, *Theoretical Mechanics* (ET, Bucharest, 1959)
50. R. Voinea, D. Voiculescu, V. Ceausu, *Mechanics* (EDP, Bucharest, 1983)
51. K.J. Waldron, G.L. Kinzel, *Kinematics, Dynamics, and Design of Machinery* (John Wiley & Sons, New York, 1999)
52. J.H. Williams Jr, *Fundamentals of Applied Dynamics* (John Wiley & Sons, New York, 1996)
53. H.B. Wilson, L.H. Turcotte, D. Halpern, *Advanced Mathematics and Mechanics Applications Using MATLAB* (Chapman & Hall/CRC, Boca Raton, 2003)
54. S. Wolfram, *Mathematica* (Wolfram Media/Cambridge University Press, Cambridge, 1999)

Index

D. B. Marghitu et al., *Statics with MATLAB®*,
DOI: 10.1007/978-1-4471-5110-4, © Springer-Verlag London 2013