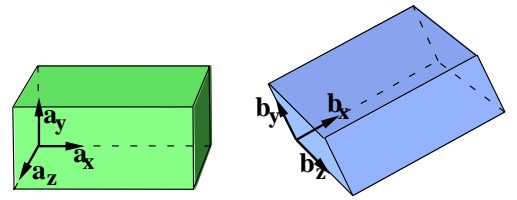


Chapter 4

Rotation matrices

4.1 Basis orientation

The figure to the right shows two *right-handed, unitary orthogonal bases*, a and b whose basis vectors are $\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z$, and $\mathbf{b}_x, \mathbf{b}_y, \mathbf{b}_z$, respectively.



A convenient way to store the *orientation* between right-handed unitary orthogonal bases a and b is with the 3×3 *rotation matrix* ${}^aR^b$, whose elements ${}^aR_{ij}^b$ ($i, j = x, y, z$) are defined in equation (1). The definition of the dot-product in equation (2.2) shows ${}^aR_{ij}^b$ is also equal to the cosine of the angle between \mathbf{a}_i and \mathbf{b}_j .^a

$${}^aR_{ij}^b \triangleq \mathbf{a}_i \cdot \mathbf{b}_j \stackrel{(2.2)}{=} \cos[\mathcal{L}(\mathbf{a}_i, \mathbf{b}_j)] \quad (1)$$

$$(i, j = x, y, z)$$

^aA rotation matrix is sometimes called a *direction cosine matrix* and its elements are called *direction cosines*.

A rotation matrix R is an orthogonal matrix which means that the transpose of R is equal to the inverse of R , i.e., $R^T = R^{-1}$. It is convenient to encapsulate the orientation information in a *rotation table* that can be read either horizontally or vertically.^a

${}^aR^b$	\mathbf{b}_x	\mathbf{b}_y	\mathbf{b}_z	(2)
\mathbf{a}_x	$\mathbf{a}_x \cdot \mathbf{b}_x$	$\mathbf{a}_x \cdot \mathbf{b}_y$	$\mathbf{a}_x \cdot \mathbf{b}_z$	
\mathbf{a}_y	$\mathbf{a}_y \cdot \mathbf{b}_x$	$\mathbf{a}_y \cdot \mathbf{b}_y$	$\mathbf{a}_y \cdot \mathbf{b}_z$	
\mathbf{a}_z	$\mathbf{a}_z \cdot \mathbf{b}_x$	$\mathbf{a}_z \cdot \mathbf{b}_y$	$\mathbf{a}_z \cdot \mathbf{b}_z$	

^aIf one or both of the bases are non-orthogonal, a rotation matrix (**not** a rotation table) stores information because the inverse of a non-orthogonal matrix is **not** its transpose.

$${}^bR^a = ({}^aR^b)^{-1} = ({}^aR^b)^T \quad (3)$$

Equation (4) shows the rotation matrix ${}^aR^d$ can be formed by successive matrix multiplication of the ${}^aR^b$, ${}^bR^c$, and ${}^cR^d$ rotation matrices.

$${}^aR^d = {}^aR^b * {}^bR^c * {}^cR^d \quad (4)$$

Uses for rotation matrices

Several uses for the rotation matrix ${}^aR^b$ in geometry, statics, motion analysis, etc., include

- Determining the **dot-product** between the unit vectors \mathbf{a}_i and \mathbf{b}_j , ($i, j = x, y, z$)
- Calculating the **angle** between the unit vectors \mathbf{a}_i and \mathbf{b}_j , ($i, j = x, y, z$)
- Calculating the **dot-product** or **cross-product** between a vector \mathbf{v} and a vector \mathbf{w} , each of which may be expressed in terms of $\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z$ and/or $\mathbf{b}_x, \mathbf{b}_y, \mathbf{b}_z$
- **Expressing** a vector written in terms of $\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z$ in terms of $\mathbf{b}_x, \mathbf{b}_y, \mathbf{b}_z$ (or vice-versa)
- Using **matrix multiplication** to calculate other rotation matrices, e.g., ${}^aR^d = {}^aR^b * {}^bR^c * {}^cR^d$
- Relating the column matrix representation of a vector \mathbf{v} expressed in basis a to the column matrix representation of \mathbf{v} expressed in basis b , i.e., $[\mathbf{v}]_a = {}^aR^b [\mathbf{v}]_b$

Example: Calculation of rotation matrix inverse

The following rotation matrix R relates two right-handed, orthogonal, unitary bases. Calculate its inverse by-hand (no calculator) in less than 30 seconds.

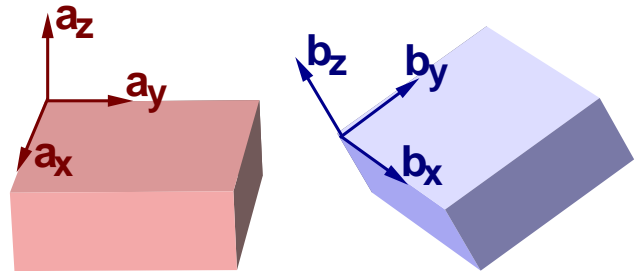
$$R = \begin{bmatrix} 0.3830 & -0.6634 & 0.6428 \\ 0.9237 & 0.2795 & -0.2620 \\ -0.0058 & 0.6941 & 0.7198 \end{bmatrix} \quad R^{-1} = \begin{bmatrix} 0.3830 & 0.9237 & -0.0058 \\ -0.6634 & 0.2795 & 0.6941 \\ 0.6428 & -0.2620 & 0.7198 \end{bmatrix}$$

Example: Calculating angles between unit vectors

The following rotation table ${}^aR^b$ relates right-handed, orthogonal, unit vectors $\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z$ and $\mathbf{b}_x, \mathbf{b}_y, \mathbf{b}_z$. Calculate the angle between \mathbf{a}_x and \mathbf{b}_z to four (or more) significant digits.

${}^aR^b$	\mathbf{b}_x	\mathbf{b}_y	\mathbf{b}_z
\mathbf{a}_x	0.9622502	-0.08418598	0.258819
\mathbf{a}_y	0.1700841	0.9284017	-0.3303661
\mathbf{a}_z	-0.2124758	0.3619158	0.9076734

$$\angle(\mathbf{a}_x, \mathbf{b}_z) = 75^\circ$$



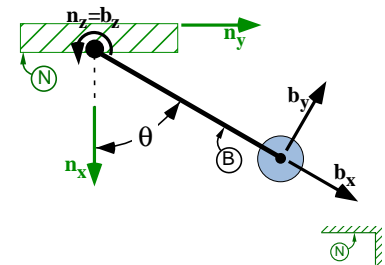
4.2 Rotation matrices and matrix multiplication

Two bases can be related with **column matrices** of unit vectors as

$$\begin{bmatrix} \mathbf{a}_x \\ \mathbf{a}_y \\ \mathbf{a}_z \end{bmatrix} = {}^aR^b \begin{bmatrix} \mathbf{b}_x \\ \mathbf{b}_y \\ \mathbf{b}_z \end{bmatrix} \quad \text{or} \quad \begin{bmatrix} \mathbf{b}_x \\ \mathbf{b}_y \\ \mathbf{b}_z \end{bmatrix} = {}^bR^a \begin{bmatrix} \mathbf{a}_x \\ \mathbf{a}_y \\ \mathbf{a}_z \end{bmatrix} \quad \text{where} \quad {}^bR^a = ({}^aR^b)^{-1}$$

4.3 Rotation matrices - who cares?

In 2D (*two dimensional*) analysis, it is possible to characterize the orientation of a rigid body B in a reference frame N with a **single angle**, e.g., the “pendulum” angle θ shown to the right.

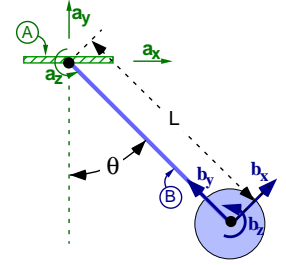


In 3D (*three dimensional*) analysis, the orientation of a rigid body B (e.g., a spiraling, wobbling, football) in a reference frame N (e.g., a stadium) **cannot** be characterized by a single angle. One convenient way to characterize the orientation of the football in the stadium is with a rotation matrix.



4.4 Example: Using a rotation matrix

The figure to the right shows a rod B connected to a fixed support A by a revolute joint. Right-handed sets of orthogonal unit vectors $\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z$ and $\mathbf{b}_x, \mathbf{b}_y, \mathbf{b}_z$, are fixed in A and B , respectively. The ${}^bR^a$ rotation matrix is given to the right.



This example shows how to **use** a rotation matrix to **express a vector** in another basis and to perform dot-products and cross-products. For example, the \mathbf{b}_x row of the ${}^bR^a$ rotation table allows \mathbf{b}_x to be **expressed** in terms of $\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z$ as

$$\mathbf{b}_x = \cos(\theta) \mathbf{a}_x + \sin(\theta) \mathbf{a}_y$$

The \mathbf{a}_x column of the ${}^bR^a$ rotation table **expresses** \mathbf{a}_x in terms of $\mathbf{b}_x, \mathbf{b}_y, \mathbf{b}_z$ as

$$\mathbf{a}_x = \cos(\theta) \mathbf{b}_x - \sin(\theta) \mathbf{b}_y$$

The **dot-product** $\mathbf{b}_y \cdot \mathbf{a}_x$ is simply the element in the \mathbf{b}_y row and \mathbf{a}_x column of the ${}^bR^a$ rotation table, i.e.,

$$\mathbf{b}_y \cdot \mathbf{a}_x = -\sin(\theta)$$

A more complicated **dot-product** example computes

$$\begin{aligned} (\mathbf{a}_x + 2\mathbf{a}_y) \cdot (x\mathbf{b}_x + y\mathbf{b}_y) &= x(\mathbf{a}_x \cdot \mathbf{b}_x) + y(\mathbf{a}_x \cdot \mathbf{b}_y) + 2x(\mathbf{a}_y \cdot \mathbf{b}_x) + 2y(\mathbf{a}_y \cdot \mathbf{b}_y) \\ &= x[\cos(\theta)] + y[-\sin(\theta)] + 2x[\sin(\theta)] + 2y[\cos(\theta)] \end{aligned}$$

An example of doing a mixed-basis **cross-product** is

$$\mathbf{b}_x \times (x\mathbf{a}_x + y\mathbf{a}_y) = [\cos(\theta)\mathbf{a}_x + \sin(\theta)\mathbf{a}_y] \times (x\mathbf{a}_x + y\mathbf{a}_y) = [y\cos(\theta) - x\sin(\theta)]\mathbf{a}_z$$

A more complicated **cross-product** example computes

$$\begin{aligned} (\mathbf{a}_x + 2\mathbf{a}_y) \times (x\mathbf{b}_x + y\mathbf{b}_y) &= \{[\cos(\theta) + 2\sin(\theta)]\mathbf{b}_x + [2\cos(\theta) - \sin(\theta)]\mathbf{b}_y\} \times (x\mathbf{b}_x + y\mathbf{b}_y) \\ &= \{y[\cos(\theta) + 2\sin(\theta)] - x[2\cos(\theta) - \sin(\theta)]\}\mathbf{b}_z \end{aligned}$$

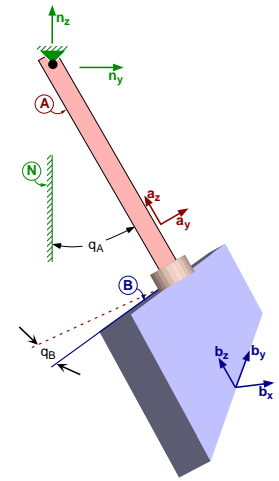
${}^bR^a$	\mathbf{a}_x	\mathbf{a}_y	\mathbf{a}_z
\mathbf{b}_x	$\cos(\theta)$	$\sin(\theta)$	0
\mathbf{b}_y	$-\sin(\theta)$	$\cos(\theta)$	0
\mathbf{b}_z	0	0	1

4.5 Rotation matrix example

The figure to the right shows a plate B connected by a revolute joint to a rod A so that B can rotate freely about A 's axis. Rod A is connected to a fixed support N by a revolute joint. (Note: The revolute joints' axes are perpendicular *not* parallel.).

There are three sets of basis vectors, namely $\mathbf{n}_x, \mathbf{n}_y, \mathbf{n}_z$; $\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z$; and $\mathbf{b}_x, \mathbf{b}_y, \mathbf{b}_z$, fixed in N, A , and B , respectively. The point of this problem is to relate these sets of unit vectors with rotation matrices.^a

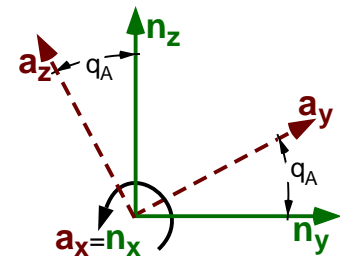
^aSimple rotation matrices can be determined with little more than the definitions of sine and cosine (SohCahToa).



4.5.1 Simple rotation matrix ${}^aR^n$

A rotation matrix ${}^aR^n$ is called a *simple rotation matrix* when one of the a basis vectors is always equal to one of the n basis vectors. In this example, ${}^aR^n$ is a simple rotation matrix because $\mathbf{a}_x = \mathbf{n}_x$.

To calculate ${}^aR^n$, the rotation matrix relating the $\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z$ and $\mathbf{n}_x, \mathbf{n}_y, \mathbf{n}_z$ unit vectors, it is helpful to **redraw these vectors** in a geometrically suggestive way as shown to the right. After using the definitions of sine and cosine to express each of $\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z$ in terms of $\mathbf{n}_x, \mathbf{n}_y, \mathbf{n}_z$ one can form the ${}^aR^n$ rotation table as shown below.

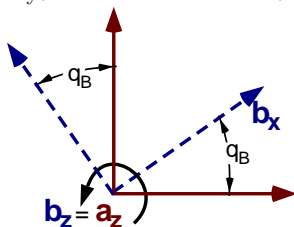


$$\begin{aligned}\mathbf{a}_x &= \mathbf{n}_x \\ \mathbf{a}_y &= \cos(q_A) \mathbf{n}_y + \sin(q_A) \mathbf{n}_z \\ \mathbf{a}_z &= -\sin(q_A) \mathbf{n}_y + \cos(q_A) \mathbf{n}_z\end{aligned}$$

${}^aR^n$	\mathbf{n}_x	\mathbf{n}_y	\mathbf{n}_z
\mathbf{a}_x	1	0	0
\mathbf{a}_y	0	$\cos(q_A)$	$\sin(q_A)$
\mathbf{a}_z	0	$-\sin(q_A)$	$\cos(q_A)$

4.5.2 Simple rotation matrix ${}^bR^a$

In order to form ${}^bR^a$, the rotation matrix relating the $\mathbf{b}_x, \mathbf{b}_y, \mathbf{b}_z$ and $\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z$ unit vectors, it is worthwhile to see that ${}^bR^a$ is a simple rotation matrix because \mathbf{b}_z is always equal to \mathbf{a}_z . An easy way to form ${}^bR^a$ is to first **redraw the vectors** in a geometrically suggestive way by drawing the plane perpendicular to $\mathbf{b}_z = \mathbf{a}_z$ (complete the figure below by adding $\mathbf{a}_x, \mathbf{a}_y$, and \mathbf{b}_x). Then, use the definitions of sine and cosine to express each of $\mathbf{b}_x, \mathbf{b}_y, \mathbf{b}_z$ in terms of $\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z$ and complete the rotation table.¹



${}^bR^a$	\mathbf{a}_x	\mathbf{a}_y	\mathbf{a}_z
\mathbf{b}_x	$\cos(q_B)$	$\sin(q_B)$	0
\mathbf{b}_y	$-\sin(q_B)$	$\cos(q_B)$	0
\mathbf{b}_z	0	0	1

¹When two sets of basis vectors $\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z$ and $\mathbf{b}_x, \mathbf{b}_y, \mathbf{b}_z$ are initially aligned and then the bases undergoes a simple rotation about $\mathbf{b}_i = \mathbf{a}_i$ by an angle θ , the ${}^bR^a$ rotation matrix has a specific pattern. The \mathbf{b}_i row and \mathbf{a}_i column contains only 1 or 0 and the remaining elements have the pattern $\begin{matrix} \cos(\theta) & \pm \sin(\theta) \\ \pm \sin(\theta) & \cos(\theta) \end{matrix}$ where the \pm sign is plus (+) if the unit vector is "hugged" (e.g, \mathbf{b}_x is **between** \mathbf{a}_x and \mathbf{a}_y) or minus (-) when the unit vector is "left out in the cold" (e.g, \mathbf{b}_y is **not between** \mathbf{a}_x and \mathbf{a}_y). [Analogy courtesy of Dr. Mandy Koop].

4.5.3 Matrix multiplication and ${}^bR^n$

The rotation matrix ${}^bR^n$ which relates $\mathbf{b}_x, \mathbf{b}_y, \mathbf{b}_z$ with $\mathbf{n}_x, \mathbf{n}_y, \mathbf{n}_z$ is formed by matrix multiplication of ${}^bR^a$ by ${}^aR^n$ as shown to the right.

$${}^bR^n = {}^bR^a * {}^aR^n \quad (4)$$

$${}^bR^n = \begin{bmatrix} \cos(q_B) & \sin(q_B) & 0 \\ -\sin(q_B) & \cos(q_B) & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(q_A) & \sin(q_A) \\ 0 & -\sin(q_A) & \cos(q_A) \end{bmatrix} = \begin{bmatrix} \cos(q_B) & \sin(q_B) \cos(q_A) & \sin(q_B) \sin(q_A) \\ -\sin(q_B) & \cos(q_B) \cos(q_A) & \cos(q_B) \sin(q_A) \\ 0 & -\sin(q_A) & \cos(q_A) \end{bmatrix}$$

The ${}^bR^n$ rotation **table** shown to the right is copied from its associated rotation **matrix**. The ${}^bR^n$ rotation table is the starting point for most relationships between $\mathbf{b}_x, \mathbf{b}_y, \mathbf{b}_z$ and $\mathbf{n}_x, \mathbf{n}_y, \mathbf{n}_z$.

${}^bR^n$	\mathbf{n}_x	\mathbf{n}_y	\mathbf{n}_z
\mathbf{b}_x	$\cos(q_B)$	$\sin(q_B) \cos(q_A)$	$\sin(q_B) \sin(q_A)$
\mathbf{b}_y	$-\sin(q_B)$	$\cos(q_B) \cos(q_A)$	$\cos(q_B) \sin(q_A)$
\mathbf{b}_z	0	$-\sin(q_A)$	$\cos(q_A)$

For example, reading the first row and first column of the rotation table gives

$$\begin{aligned} \mathbf{b}_x &= \cos(q_B) \mathbf{n}_x + \sin(q_B) \cos(q_A) \mathbf{n}_y + \sin(q_B) \sin(q_A) \mathbf{n}_z \\ \mathbf{n}_x &= \cos(q_B) \mathbf{b}_x + -\sin(q_B) \mathbf{b}_y + 0 \mathbf{b}_z \end{aligned}$$

As shown below, the dot product $\mathbf{b}_x \cdot \mathbf{n}_z$ is simply the element in the \mathbf{b}_x row and \mathbf{n}_z column of the ${}^bR^n$ rotation table (the angle between \mathbf{b}_x and \mathbf{n}_z can be calculated via the definition of the dot-product).

$$\mathbf{b}_x \cdot \mathbf{n}_z = \sin(q_B) \sin(q_A) \quad \angle(\mathbf{b}_x, \mathbf{n}_z) = \text{acos}[\sin(q_B) \sin(q_A)]$$

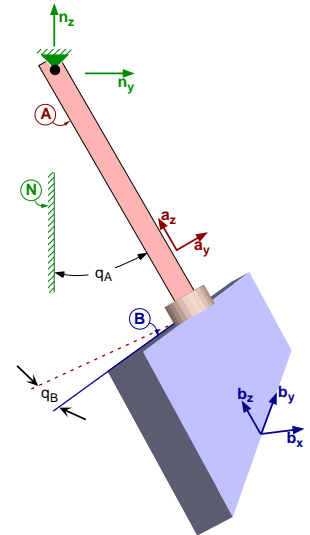
4.5.4 Calculating babyboot rotation matrices with Autolev

The Autolev input and output responses for the rotation matrices in Section 4.5 are shown below.

```
(1) % File: BabybootRotationMatrices.al
(2) %-----
(3) RigidBody N          % Reference frame
(4) RigidBody A          % Upper rod
(5) RigidBody B          % Lower plate
(6) Variable qA         % Pendulum angle
(7) Variable qB         % Plate angle
(8) %-----
(9) A.RotateX( N, qA )  % A rotates "about +x" in N by qA
-> (10) A_N = [1, 0, 0; 0, cos(qA), sin(qA); 0, -sin(qA), cos(qA)]

(11) B.RotateZ( A, qB ) % B rotates "about +z" in A by qB
-> (12) B_A = [cos(qB), sin(qB), 0; -sin(qB), cos(qB), 0; 0, 0, 1]

(13) BRotationMatrixN = B.GetRotationMatrix( N )
-> (14) BRotationMatrixN[1,1] = cos(qB)
-> (15) BRotationMatrixN[1,2] = sin(qB)*cos(qA)
-> (16) BRotationMatrixN[1,3] = sin(qA)*sin(qB)
-> (17) BRotationMatrixN[2,1] = -sin(qB)
-> (18) BRotationMatrixN[2,2] = cos(qA)*cos(qB)
-> (19) BRotationMatrixN[2,3] = sin(qA)*cos(qB)
-> (20) BRotationMatrixN[3,1] = 0
-> (21) BRotationMatrixN[3,2] = -sin(qA)
-> (22) BRotationMatrixN[3,3] = cos(qA)
```

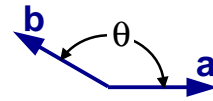


4.6 What is an angle?

As described in Section ??, **distance** is the measure of space between **two points**. The rotational analog to distance is **angle**, defined as the measure of the sweep between **two vectors**. For example:

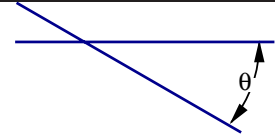
Angle between two vectors

The angle θ between **two vectors** \mathbf{a} and \mathbf{b} can be calculated from equation (2.2) as $\theta = \arccos\left(\frac{\mathbf{a} \cdot \mathbf{b}}{|\mathbf{a}| |\mathbf{b}|}\right)$ where the \arccos function returns $0 \leq \theta \leq 180^\circ$.



Angle between two lines

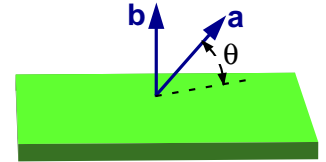
The angle θ between **two lines** is defined as the smallest angle between all vectors aligned with the lines, hence $0 \leq \theta \leq 90^\circ$.



Angle between a vector and a plane

The angle θ between a **vector** \mathbf{a} and a **plane** perpendicular to a vector \mathbf{b} is $\theta = 90^\circ - \angle(\mathbf{a}, \mathbf{b})$, where $\angle(\mathbf{a}, \mathbf{b})$ is the angle between \mathbf{a} and \mathbf{b} .

Alternately, if \mathbf{b} points oppositely so $\mathbf{a} \cdot \mathbf{b} < 0$, $\theta = \angle(\mathbf{a}, \mathbf{b}) - 90^\circ$.



Using three vectors (or two vectors and a sense) an angle θ can be calculated with the `atan2` function described in Section 1.6.4 so that $-180 \leq \theta \leq 180^\circ$. By using *wrap*, θ may have values $-\infty < \theta < +\infty$.

It is worth reiterating that, like distances, angles are inherently positive quantities. Angles may be *regarded* as negative when one associates a *sense* with their value. For example, one may *measure* a distance from sea level as 10 m. Alternately, one may say the distance from sea-level is -10 m by implying that a upward sense is positive. Similarly, angles *measurements* are inherently positive, but an angle may be regarded as negative when a *sense* is associated with the angle (e.g., counter-clockwise is positive). Historically, angles (e.g., used by the ancient Greeks) predate negative numbers (in widespread use by Europeans in 1700 A.D.) by thousands of years.