1 Transformation of Vectors

1.1 A Basic Example

A vector $\mathbf{a}$ is defined by its components $[\mathbf{a}] = [a_1, a_2]^T$ in the Cartesian base $\{\mathbf{e}_1, \mathbf{e}_2\}$:

$$\mathbf{a} = a_1\mathbf{e}_1 + a_2\mathbf{e}_2 = a_i\mathbf{e}_i \tag{1}$$

The base vectors are rotated to $\mathbf{e}'_1$ and $\mathbf{e}'_2$ with an angle α around the out-of-plane axis. It follows from simple trigonometry, that their components in the original frame are:

$$\left[\mathbf{e}'_1\right] = [cos(\alpha) \quad sin(\alpha)]^T \quad \text{and} \quad \left[\mathbf{e}'_2\right] = [-sin(\alpha) \quad cos(\alpha)]^T . \tag{2}$$

From here it is clear, that if we are looking for a matrix $[\mathbf{Q}]$ for which $\left[\mathbf{e}'_1\right] = [\mathbf{Q}]\ [\mathbf{e}_1]$ and $\left[\mathbf{e}'_2\right] = [\mathbf{Q}]\ [\mathbf{e}_2]$, it must be

$$[\mathbf{Q}] = \begin{bmatrix} cos(\alpha) & -sin(\alpha) \\ sin(\alpha) & cos(\alpha) \end{bmatrix} = \begin{bmatrix} \mathbf{e}'_{11} & \mathbf{e}'_{21} \\ \mathbf{e}'_{12} & \mathbf{e}'_{22} \end{bmatrix}, \tag{3}$$

where $\mathbf{e}'_{1i}$ are the components of $\mathbf{e}'_1$, and $\mathbf{e}'_{2i}$ are the components of $\mathbf{e}'_2$. It shows, that the first column of $[\mathbf{Q}]$ are the components of the first base vector $\mathbf{e}'_1$ in the original frame, etc. If as the base vectors $\mathbf{e}_1$ and $\mathbf{e}_2$ are transformed to $\mathbf{e}'_1$ and $\mathbf{e}'_2$ the vector $\mathbf{a}$ transforms with them, it transforms into a new vector $\mathbf{b}$, the components of which in the new frame are b'_1 and b'_2. These new components can be found by applying the same transformation on $\mathbf{a}$:

$$\mathbf{b} = \mathbf{Q}\mathbf{a} = a_1\mathbf{Q}\mathbf{e}_1 + a_2\mathbf{Q}\mathbf{e}_2 = a_1\mathbf{e}'_1 + a_2\mathbf{e}'_2 = b'_1\mathbf{e}'_1 + b'_2\mathbf{e}'_2, \tag{4}$$

which confirms the somewhat trivial fact that the components of the new vector $\mathbf{b}$ in the frame $(\mathbf{e}'_1, \mathbf{e}'_2)$ are the same as the components of the old vector $\mathbf{a}$ in the frame $(\mathbf{e}_1, \mathbf{e}_2)$. To find the components of vector $\mathbf{b}$ in the original frame, we can express it in both frames as

$$\mathbf{b} = b'_1\mathbf{e}'_1 + b'_2\mathbf{e}'_2 = b_1\mathbf{e}_1 + b_2\mathbf{e}_2 = b_i\mathbf{e}_i \tag{5}$$

and then multiply with the base vectors of the frame we know the coordinates of our vector at:

$$\mathbf{b}\cdot\mathbf{e}_j = b'_1(\mathbf{e}'_1\cdot\mathbf{e}_j) + b'_2(\mathbf{e}'_2\cdot\mathbf{e}_j) = b_i(\mathbf{e}_i\cdot\mathbf{e}_j) = b_i\delta_{ij} = b_j. \tag{6}$$

This gives us the components b_i. The results in matrix form:

$$\begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} \mathbf{e}'_1\cdot\mathbf{e}_1 & \mathbf{e}'_2\cdot\mathbf{e}_1 \\ \mathbf{e}'_1\cdot\mathbf{e}_2 & \mathbf{e}'_2\cdot\mathbf{e}_2 \end{bmatrix} \begin{bmatrix} b'_1 \\ b'_2 \end{bmatrix}. \tag{7}$$

If we break down the matrix on the middle, we can see that

$$\begin{bmatrix} \mathbf{e}'_1\cdot\mathbf{e}_1 & \mathbf{e}'_2\cdot\mathbf{e}_1 \\ \mathbf{e}'_1\cdot\mathbf{e}_2 & \mathbf{e}'_2\cdot\mathbf{e}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \end{bmatrix} \begin{bmatrix} \mathbf{e}'_1 & \mathbf{e}'_2 \end{bmatrix} = \begin{bmatrix} \mathbf{e}_{11} & \mathbf{e}_{12} \\ \mathbf{e}_{21} & \mathbf{e}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{e}'_{11} & \mathbf{e}'_{21} \\ \mathbf{e}'_{12} & \mathbf{e}'_{22} \end{bmatrix} = \mathbf{e}_i \otimes \mathbf{e}'_j. \tag{8}$$

Recognizing $[\mathbf{Q}]$ in the above expression, we can write

$$b_i = b'_j\ \mathbf{e}_i \otimes \mathbf{e}'_j = Q_{ij}b'_j \quad \Longleftrightarrow \quad [\mathbf{b}] = [\mathbf{Q}]\,[\mathbf{b}]' . \tag{9}$$

With similar thinking, we can calculate the components of $\mathbf{a}$ in the new frame as:

$$\mathbf{a} = a_1\mathbf{e}_1 + a_2\mathbf{e}_2 = a'_1\mathbf{e}'_1 + a'_2\mathbf{e}'_2 = a'_i\mathbf{e}'_i \tag{10}$$

$$\mathbf{a}\cdot\mathbf{e}'_j = a_1(\mathbf{e}_1\cdot\mathbf{e}'_j) + a_2(\mathbf{e}_2\cdot\mathbf{e}'_j) = a'_i(\mathbf{e}_i\cdot\mathbf{e}'_j) = a'_i\delta_{ij} = a'_j. \tag{11}$$

and rearrange the results as

$$\begin{bmatrix} a_1^{'} \\ a_2^{'} \end{bmatrix} = \begin{bmatrix} \mathbf{e}_1 \cdot \mathbf{e}_1^{'} & \mathbf{e}_2 \cdot \mathbf{e}_1^{'} \\ \mathbf{e}_1 \cdot \mathbf{e}_2^{'} & \mathbf{e}_2 \cdot \mathbf{e}_2^{'} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}, \tag{12}$$

or simply

$$a_i^{'} = a_j \, \mathbf{e}_i^{'} \otimes \mathbf{e}_j = Q_{ji} a_j \quad \iff \quad [\mathbf{a}]^{'} = [\mathbf{Q}]^T \, [\mathbf{a}] \,. \tag{13}$$

Finally, the components of the original basis vectors in the new frame can be expressed as

$$[\mathbf{e}_1]^{'} = [\mathbf{Q}]^T \, [\mathbf{e}_1] \quad \text{and} \quad [\mathbf{e}_2]^{'} = [\mathbf{Q}]^T \, [\mathbf{e}_2] \,. \tag{14}$$

Note that $[\mathbf{e}_1]^{'}$ are the components of $\mathbf{e}_1$ in the transformed frame, while $\left[\mathbf{e}_1^{'}\right]$ are the components of $\mathbf{e}_1^{'}$ in the original frame.

Another way of introducing these concepts is to first recognize, that the components of $\mathbf{a}$ in the basis $\{\mathbf{e}_1^{'}, \mathbf{e}_2^{'}, \mathbf{e}_3^{'}\}$ are, by definition

$$a_1^{'} = \mathbf{a} \cdot \mathbf{e}_1^{'}, \tag{15}$$
$$a_2^{'} = \mathbf{a} \cdot \mathbf{e}_2^{'}. \tag{16}$$

then to substitute $\mathbf{a} = a_1 \mathbf{e}_1 + a_2 \mathbf{e}_2$

$$a_1^{'} = (a_1 \mathbf{e}_1 + a_2 \mathbf{e}_2) \cdot \mathbf{e}_1^{'} = a_1(\mathbf{e}_1^{'} \cdot \mathbf{e}_1) + a_2(\mathbf{e}_1^{'} \cdot \mathbf{e}_2), \tag{17}$$
$$a_2^{'} = (a_1 \mathbf{e}_1 + a_2 \mathbf{e}_2) \cdot \mathbf{e}_2^{'} = a_1(\mathbf{e}_2^{'} \cdot \mathbf{e}_1) + a_2(\mathbf{e}_2^{'} \cdot \mathbf{e}_2), \tag{18}$$

to come to the matrix expression we've already seen:

$$\begin{bmatrix} a_1^{'} \\ a_2^{'} \end{bmatrix} = \begin{bmatrix} \mathbf{e}_1^{'} \cdot \mathbf{e}_1 & \mathbf{e}_1^{'} \cdot \mathbf{e}_2 \\ \mathbf{e}_2^{'} \cdot \mathbf{e}_1 & \mathbf{e}_2^{'} \cdot \mathbf{e}_2 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = [\mathbf{Q}]^T \, [\mathbf{a}] \,. \tag{19}$$

The literature is not very consistent here, the matrix $[\mathbf{Q}]^T$ is commonly known as the "**direction cosine matrix** from e to $e^{'}$", or the "**transformation matrix** from e to $e^{'}$" or as the "**rotation matrix** from e to $e^{'}$". Or its transpose (for example in SymPy). Some sources make a difference between a rotation matrix and a direction cosine matrix, some not. The good news is that using orthonormal base vectors takes a lot away from the ambiguity, and the answer is always either the matrix or the transpose of it.

It is beneficial from a computational point of view to define the DCM matrix as $[\mathbf{Q}]^T$. This way, if a frame is embedded in a parent frame, then the DCM matrix of the transformation of components from the parent frame to the child frame is a matrix, where each row is the component array of the i-th base vector of the child frame, relative to the parent frame. Simply speaking, to access a base vector of a frame in Python syntax, one writes $A[0,:]$, or simply $A[0]$. It is obvioust the short version that takes our attention, which is simply not possible if we have to type $A[:,0]$. Then $A[0]$ would have a totally different (and wrong) meaning .

Definition **Direction Cosine Matrix**
A direction cosine matrix from frame $A = \{\mathbf{a}_1, \mathbf{a}_2\}$ to frame $B = \{\mathbf{b}_1, \mathbf{b}_2\}$ is denoted as ${}^{A}\mathbf{R}^{B}$, is defined as the matrix

$$\left[{}^{A}\mathbf{R}^{B}\right] = \begin{bmatrix} \mathbf{a}_1 \cdot \mathbf{b}_1 & \mathbf{a}_2 \cdot \mathbf{b}_1 \\ \mathbf{a}_1 \cdot \mathbf{b}_2 & \mathbf{a}_2 \cdot \mathbf{b}_2 \end{bmatrix}, \tag{20}$$

and should be referred to as *"the direction cosine matrix from A to B"* or simply *"the DCM from A to B"*.

If a vector $\mathbf{v}$ is given in frames A and B as

$$\mathbf{v} = \alpha_1 \mathbf{a}_1 + \alpha_2 \mathbf{a}_2 = \beta_1 \mathbf{b}_1 + \beta_2 \mathbf{b}_2, \tag{21}$$

then the matrix ${}^{A}\mathbf{R}^{B}$ is the matrix that

- transforms the components as

$$[\beta] = \left[{}^{A}\mathbf{R}^{B}\right][\alpha], \quad [\alpha] = \left[{}^{A}\mathbf{R}^{B}\right]^{-1}[\beta] = \left[{}^{B}\mathbf{R}^{A}\right][\beta] \tag{22}$$

- transforms the base vctors as

$$[\mathbf{b}_i] = \left[{}^{B}\mathbf{R}^{A}\right][\mathbf{a}_i], \quad (i = 1, 2, 3, ...). \tag{23}$$

> Here we used the fact that the DCM matrices are orthogonal matrices, hence $\left[{}^{A}\mathbf{R}^{B}\right]^{-1} = \left[{}^{A}\mathbf{R}^{B}\right]^{T} = \left[{}^{B}\mathbf{R}^{A}\right]$.

1.2 Chaining Transformations

Let now have four co-ordinate frames A, B, C and D and one vector $\mathbf{v}$. Frame B and D are defined relative to frame A, frame C is defined relative to B. We have the components of $\mathbf{v}$ available at frame C and we want to get the components of the same vector in frame D. Now, as the complexity of the problem grows, we need to be a bit more careful about the notations we use. For the present discussion, let the components of $\mathbf{v}$ be denoted with the greek versions of the the actual frame. Now, we can simply write

$$\mathbf{v} = \gamma_1 \mathbf{c}_1 + \gamma_2 \mathbf{c}_2 = \delta_1 \mathbf{d}_1 + \delta_2 \mathbf{d}_2, \tag{24}$$

and get the required components as

$$\begin{bmatrix} \delta_1 \\ \delta_2 \end{bmatrix} = \left[{}^{C}\mathbf{R}^{D}\right] \begin{bmatrix} \gamma_1 \\ \gamma_2 \end{bmatrix} = \begin{bmatrix} \mathbf{c}_1 \cdot \mathbf{d}_1 & \mathbf{c}_2 \cdot \mathbf{d}_1 \\ \mathbf{c}_1 \cdot \mathbf{d}_2 & \mathbf{c}_2 \cdot \mathbf{d}_2 \end{bmatrix} \begin{bmatrix} \gamma_1 \\ \gamma_2 \end{bmatrix}, \tag{25}$$

but to carry out the actual calculation, we need the sets of base vectors $\{\mathbf{c}_1, \mathbf{c}_2\}$ and $\{\mathbf{d}_1, \mathbf{d}_2\}$ expressed in the same coordinate system. This common system can be the frame of C, D, or a third frame, some sort of neutral ground.

It is clear, that since we only have the components in frame C, it must be on one side of the equality sign. Since C itself is only known with respect to B, we actually do not have much choice about where to start:

$$\mathbf{v} = \gamma_1 \mathbf{c}_1 + \gamma_2 \mathbf{c}_2 = \beta_1 \mathbf{b}_1 + \beta_2 \mathbf{b}_2. \tag{26}$$

From here we can readily write

$$\begin{bmatrix}\beta_1\\\beta_2\end{bmatrix} = \begin{bmatrix}{}^C\mathbf{R}^B\end{bmatrix}\begin{bmatrix}\gamma_1\\\gamma_2\end{bmatrix} = \begin{bmatrix}\mathbf{c}_1\cdot\mathbf{b}_1 & \mathbf{c}_2\cdot\mathbf{b}_1\\ \mathbf{c}_1\cdot\mathbf{b}_2 & \mathbf{c}_2\cdot\mathbf{b}_2\end{bmatrix}\begin{bmatrix}\gamma_1\\\gamma_2\end{bmatrix} = \begin{bmatrix}c_{11} & c_{21}\\ c_{12} & c_{22}\end{bmatrix}\begin{bmatrix}\gamma_1\\\gamma_2\end{bmatrix}.$$

Combine this with,

$$\begin{bmatrix}\alpha_1\\\alpha_2\end{bmatrix} = \begin{bmatrix}{}^B\mathbf{R}^A\end{bmatrix}\begin{bmatrix}\beta_1\\\beta_2\end{bmatrix} \quad \text{and} \quad \begin{bmatrix}\delta_1\\\delta_2\end{bmatrix} = \begin{bmatrix}{}^A\mathbf{R}^D\end{bmatrix}\begin{bmatrix}\alpha_1\\\alpha_2\end{bmatrix}$$

to finally get

$$\begin{bmatrix}\delta_1\\\delta_2\end{bmatrix} = \begin{bmatrix}{}^A\mathbf{R}^D\end{bmatrix}\underbrace{\begin{bmatrix}{}^B\mathbf{R}^A\end{bmatrix}\begin{bmatrix}{}^C\mathbf{R}^B\end{bmatrix}}_{{}^C\mathbf{R}^A}\begin{bmatrix}\gamma_1\\\gamma_2\end{bmatrix} = \begin{bmatrix}{}^C\mathbf{R}^D\end{bmatrix}\begin{bmatrix}\gamma_1\\\gamma_2\end{bmatrix}.$$

Similarly,

$$\begin{bmatrix}\gamma_1\\\gamma_2\end{bmatrix} = \underbrace{\begin{bmatrix}{}^B\mathbf{R}^C\end{bmatrix}\begin{bmatrix}{}^A\mathbf{R}^B\end{bmatrix}}_{{}^A\mathbf{R}^C}\begin{bmatrix}{}^D\mathbf{R}^A\end{bmatrix}\begin{bmatrix}\delta_1\\\delta_2\end{bmatrix} = \begin{bmatrix}{}^D\mathbf{R}^C\end{bmatrix}\begin{bmatrix}\delta_1\\\delta_2\end{bmatrix}.$$

Using this technique, a DCM matrix can be established between any two frames in a two-step procedure, and it only requires to store the components of the frames in their parent frames and to keep track of parent-child relationships.

> If a vector $\mathbf{v}$ is known in frame $A = \{\mathbf{a}_1, \mathbf{a}_2\}$ with components $[\mathbf{a}] = [\alpha_1, \alpha_2]^T$, then the components of the same vector in frame B can be calculated in two steps, which are
>
> - **Step 1** Calculate the DCM from A to B as $\left[{}^A\mathbf{R}^B\right] = \left[{}^X\mathbf{R}^B\right]\left[{}^A\mathbf{R}^X\right]$, where X is a frame in which frames A and B are readily expressed.
> - **Step 2** Calculate the components of $\mathbf{v}$ in B as $[\boldsymbol{\beta}] = \left[{}^A\mathbf{R}^B\right][\boldsymbol{\alpha}]$.

Usually, frame X is selected as the ambient space, which at the technical level is probably the root entity in a tree-like data structure formalizing the parent-child relationships of the frames. In fact, if we plan to handle deep levels of frames being embedded in one another, then the concept of the ambient frame is inevitable

Definition **Ambient Frame**
A frame is ambient, if it is not embedded in another frame.

It is important to note, that the ambient frame could be just another frame embedded in an even broader context, but since every other frame in the system is oriented relative to this frame, it wouldn't matter. The point is, that with the concept of the ambient frame, we can work out a simple recursive algorithm, that returns the DCM of the ambient frame to any other frame, independent on the depth at which it is embedded, and we can reuse this machinery to automate the two-step procedure between arbitrary frames, as long as there is a path between the two.

1.3 Applications

Application 1

A load vector $\mathbf{f}$ at a point of a beam element is known in its local coordinate frame $L = \{\mathbf{r}, \mathbf{s}, \mathbf{t}\}$ as $[\mathbf{f}]_L$ and we want to calculate its components $[\mathbf{f}]_G$ in the global frame $G = \{\mathbf{i}, \mathbf{j}, \mathbf{k}\}$. The DCM matrix from L to G is

$$\left[{}^{L}\mathbf{R}^{G}\right] = \begin{bmatrix} \mathbf{r}\cdot\mathbf{i} & \mathbf{s}\cdot\mathbf{i} & \mathbf{t}\cdot\mathbf{i} \\ \mathbf{r}\cdot\mathbf{j} & \mathbf{s}\cdot\mathbf{j} & \mathbf{t}\cdot\mathbf{j} \\ \mathbf{r}\cdot\mathbf{k} & \mathbf{s}\cdot\mathbf{k} & \mathbf{t}\cdot\mathbf{k} \end{bmatrix} = \begin{bmatrix} r_1 & s_1 & t_1 \\ r_2 & s_2 & t_2 \\ r_3 & s_3 & t_3 \end{bmatrix}, \tag{27}$$

and the components in the global frame can be computed as

$$[\mathbf{f}]_G = \left[{}^{L}\mathbf{R}^{G}\right][\mathbf{f}]_L\,. \tag{28}$$

If we applied this to the generalized load vector $\mathbf{f}_e$ of a 2-noded Bernoulli beam element, the transformation matrix $\left[{}^{L}\mathbf{R}_e^{G}\right]$ to apply would be a block-diagonal one, with blocks of $\left[{}^{L}\mathbf{R}^{G}\right]$, the number of blocks being 4, producing a $12x12$ matrix.

Application 2

The stiffness matrix $[\mathbf{K}_e]_L$ of a Bernoulli beam element is known in the element's local coordinate frame $L = \{\mathbf{r}, \mathbf{s}, \mathbf{t}\}$ as $[\mathbf{f}]_L$ and we want to calculate its components $[\mathbf{f}]_G$ in the global frame $G = \{\mathbf{i}, \mathbf{j}, \mathbf{k}\}$. We can formulate the equilibrium of the beam element in the two systems as

$$[\mathbf{K}_e]_L\,[\mathbf{u}_e]_L = [\mathbf{f}_e]_L \quad \text{and} \quad [\mathbf{K}_e]_G\,[\mathbf{u}_e]_G = [\mathbf{f}_e]_G\,. \tag{29}$$

If we reuse the transformation rule from the previous application and apply it on the local equilibrium equation, we get

$$\left[{}^{G}\mathbf{R}_e^{L}\right]^T[\mathbf{K}_e]_L\left[{}^{G}\mathbf{R}_e^{L}\right][\mathbf{u}_e]_G = [\mathbf{f}_e]_G\,, \tag{30}$$

from which the global components be recognized as

$$[\mathbf{K}_e]_G = \left[{}^{G}\mathbf{R}_e^{L}\right]^T[\mathbf{K}_e]_L\left[{}^{G}\mathbf{R}_e^{L}\right] = \left[{}^{L}\mathbf{R}_e^{G}\right][\mathbf{K}_e]_L\left[{}^{G}\mathbf{R}_e^{L}\right]. \tag{31}$$