Polar Decomposition I.

In last example the object has clearly been stretched and rotated. But by how much? The rotation doesn't contribute to stress, but the deformation does. So it is necessary to partition the two mechanisms out of **F** in order to determine the stress and strain state.

$$\mathbf{F} = \begin{bmatrix} 1.300 & -0.375 \\ 0.750 & 0.650 \end{bmatrix} = \begin{bmatrix} 0.866 & -0.500 \\ 0.500 & 0.866 \end{bmatrix} \begin{bmatrix} 1.50 & 0.00 \\ 0.00 & 0.75 \end{bmatrix}$$

or

$$\mathbf{F} = \mathbf{R} \cdot \mathbf{U}$$

where **R** is the **rotation matrix**, and **U** is the **right stretch tensor** that is responsible for all the problems in life: stress, strain, fatigue, cracks, fracture, etc. Note that the process is read from right to left, not left to right. **U** is applied first, then **R**.

This partitioning of the **deformation gradient** into the product of a **rotation matrix** and **stretch tensor** is known as a **Polar Decomposition**.

Polar Decomposition II.

The deformed and rotated state could equally-well be arrived at by rotating it first, and then deforming it second. In this case, the reference configuration, is first rotated by the same 30° angle to arrive at an intermediate configuration and then the intermediate configuration is deformed to arrive at the final, deformed state.

The deformation gradient can be written as:

$$\mathbf{F} = \begin{bmatrix} 1.300 & -0.375 \\ 0.750 & 0.650 \end{bmatrix} = \begin{bmatrix} 1.313 & 0.325 \\ 0.325 & 0.938 \end{bmatrix} \begin{bmatrix} 0.866 & -0.500 \\ 0.500 & 0.866 \end{bmatrix}$$

or

$$\mathbf{F} = \mathbf{V} \cdot \mathbf{R}$$

where $\mathbf{R}$ is the same **rotation matrix**, and $\mathbf{V}$ is the **left stretch tensor**.

Polar Decomposition III.

It is relatively easy to develop a relationship between $\mathbf{V}$ and $\mathbf{U}$. Since $\mathbf{F} = \mathbf{V} \cdot \mathbf{R}$ and $\mathbf{F} = \mathbf{R} \cdot \mathbf{U}$, then

$$\mathbf{V} \cdot \mathbf{R} = \mathbf{R} \cdot \mathbf{U}$$

and post-multiplying through by $\mathbf{R}^T$ gives

$$\mathbf{V} \cdot \mathbf{R} \cdot \mathbf{R}^T = \mathbf{R} \cdot \mathbf{U} \cdot \mathbf{R}^T$$

But since $\mathbf{R} \cdot \mathbf{R}^T = \mathbf{I}$, this leaves

$$\mathbf{V} = \mathbf{R} \cdot \mathbf{U} \cdot \mathbf{R}^T$$

as the relationship between $\mathbf{V}$ and $\mathbf{U}$. Alternatively, solving for $\mathbf{U}$ gives

$$\mathbf{U} = \mathbf{R}^T \cdot \mathbf{V} \cdot \mathbf{R}$$

Strain ellipse or ellipsoid

According to definition, the **strain ellipse** results from transformation of unit circle, which in matrix form is given by equation:

$$\begin{bmatrix} X & Y \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} = X^T X = 1$$

The deformation gradient equation could be written in terms of deformed coordinates as:

$$X = \mathbf{F}^{-1} x$$

Substituting into equation of unit circle we obtain:

$$(\mathbf{F}^{-1} x)^T \cdot \mathbf{F}^{-1} x = x^T (\mathbf{F}^{-1})^T \cdot \mathbf{F}^{-1} x = x^T (\mathbf{F} \cdot \mathbf{F}^T)^{-1} x = x^T \mathbf{B}^{-1} x = 1$$

where matrix $\mathbf{B} = \mathbf{F} \cdot \mathbf{F}^T$ is called **Finger** or **Left Cauchy-Green** deformation tensor. It's inverse represents ellipse or ellipsoid and is commonly called **ellipsoid tensor**.

Reciprocal ellipse or ellipsoid

According to definition, the **reciprocal ellipse** is transformed to unit circle, which in matrix form is given by equation:

$$\begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = x^T x = 1$$

Substituting equation for deformation gradient $x = \mathbf{F}X$ into equation of unit circle in deformed coordinates we obtain:

$$(\mathbf{F}X)^T \cdot \mathbf{F}X = X^T \mathbf{F}^T \cdot \mathbf{F}X = X^T \mathbf{C} X = 1$$

where matrix $\mathbf{C} = \mathbf{F}^T \cdot \mathbf{F}$ is called **Green's** or **Right Cauchy-Green** deformation tensor.

Deformation tensors and polar decomposition

Plugging the polar decomposition into equations for above defined deformation tensors gives a rather surprising result. Using $\mathbf{F} = \mathbf{V} \cdot \mathbf{R}$:

$$\mathbf{B} = \mathbf{F} \cdot \mathbf{F}^T = (\mathbf{V} \cdot \mathbf{R}) \cdot (\mathbf{V} \cdot \mathbf{R})^T = \mathbf{V} \cdot \mathbf{R} \cdot \mathbf{R}^T \cdot \mathbf{V}^T = \mathbf{V} \cdot \mathbf{V}^T$$

Using $\mathbf{F} = \mathbf{R} \cdot \mathbf{U}$:

$$\mathbf{C} = \mathbf{F}^T \cdot \mathbf{F} = (\mathbf{R} \cdot \mathbf{U})^T \cdot (\mathbf{R} \cdot \mathbf{U}) = \mathbf{U}^T \cdot \mathbf{R}^T \cdot \mathbf{R} \cdot \mathbf{U} = \mathbf{U}^T \cdot \mathbf{U}$$

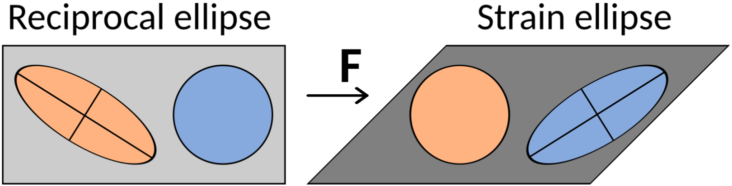
As $\mathbf{U}$ and $\mathbf{V}$ are both symmetric, so $\mathbf{U} = \mathbf{U}^T$ and $\mathbf{U}^T \cdot \mathbf{U} = \mathbf{U} \cdot \mathbf{U}$. Likewise $\mathbf{V} = \mathbf{V}^T$ and $\mathbf{V} \cdot \mathbf{V}^T = \mathbf{V} \cdot \mathbf{V}$. Finally, $\mathbf{U} \cdot \mathbf{U}$ is sometimes written as $\mathbf{U}^2$ and $\mathbf{V} \cdot \mathbf{V}$ is sometimes written as $\mathbf{V}^2$. Therefore:

$$\mathbf{B} = \mathbf{V}^2 \text{ and } \mathbf{C} = \mathbf{U}^2$$

The surprising result here is that the rotation matrix, $\mathbf{R}$, has been eliminated from the problem in both cases.

Properties of strain and reciprocal deformation

The eigenvectors of **B** define orientation of principal axes of the **strain ellipse/ellipsoid** in deformed state. The eigenvalues are quadratic elongations along principal directions, i.e. the lengths of semi-axes of the strain ellipse or ellipsoid are the square roots of the corresponding eigenvalues.



The eigenvectors of **C** define orientation of principal axes of the **reciprocal ellipse/ellipsoid** in undeformed state. The eigenvalues are quadratic elongations along principal directions, i.e. the lengths of semi-axes of the reciprocal ellipse or ellipsoid are the square roots of the corresponding eigenvalues.

Python exercise

Calculate orientation and axial ratio of strain ellipse for deformation gradient **F**.

```
>>> from pylab import *
>>> # deformation gradient
>>> F = array([[1, 1], [0, 1]])
>>> # calculate Left Cauchy-Green deformation tensor
>>> B = dot(F, F.T)
>>> s, U = eig(B)
>>> # calculate axial ratio and orientation
>>> ar = sqrt(s[0]/s[1])
>>> ori = degrees(arccos(U[0, 0]))
>>> print('Ori:{:g} Ar:{:g}'.format(ori, ar))
Ori:31.7175 Ar:2.61803
```