# LAB 8 - LINEAR TRANSFORMATIONS

## 1. Introduction to Linear Transformations

Vectors provide a very useful way to represent images in computer graphics and animation. These images are aptly referred to as *vector graphics*, and use 2-dimensional arrangements of points which are connected by lines and curves to form polygons and other shapes. Vector graphics are found in PDF and ESP file formats (as well as others), and have many advantages over bitmap formats such as JPG and PNG which store images directly as a series of pixel values. For example, unlike JPG images, vector graphics can resized arbitrarily large without the images becoming pixelated. Furthermore, because images are represented as a series of vectors, they can be resized, rotated, reflected, and modified in other ways using techniques from linear algebra, such as linear transformations. In this lab we will explore linear transformations from a geometric viewpoint, and see simple applications to graphics.

A *transformation* is a function, or rule, which assigns a vector in $\mathbb{R}^m$ to each vector in $\mathbb{R}^n$. For instance, a transformation $L$ which maps vectors in $\mathbb{R}^2$ to vectors in $\mathbb{R}^3$ could be defined by

$$L\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = \begin{bmatrix} x + y^2 \\ x^2 - 5y \\ y + 1 \end{bmatrix}.$$

Thus, for instance,

$$L\left(\begin{bmatrix} 1 \\ 2 \end{bmatrix}\right) = \begin{bmatrix} 5 \\ -9 \\ 3 \end{bmatrix}.$$

We use the notation $L : \mathbb{R}^n \to \mathbb{R}^m$ to denote a transformations that accepts vectors in $\mathbb{R}^n$ as input, and returns vectors in $\mathbb{R}^m$ as output. We call the set $\mathbb{R}^n$ of all possible input vectors the *domain of* $L$, and the set $\mathbb{R}^m$ where all of the output vectors lie the *codomain of* $L$.

A *linear transformation* is a special type of transformation which respects vector addition and scalar multiplication. More precisely, we say that a transformation $T : \mathbb{R}^n \to \mathbb{R}^m$ is a

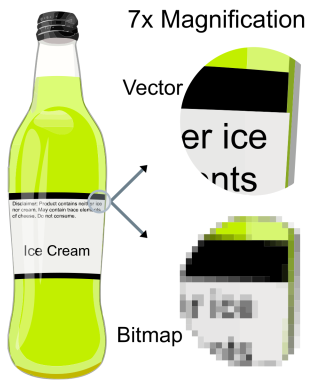
7x Magnification
Vector
er ice
nts
Disclaimer: Product contains neither ice nor cream. May contain trace elements of cheese. Do not consume.
Ice Cream
Bitmap

FIGURE 1. The effect of magnifying vector graphics compared with magnifying a bitmap image. Image courtesy of Wikipedia.

linear transformation from $\mathbb{R}^n$ into $\mathbb{R}^m$ if

$$T(a\mathbf{v} + b\mathbf{w}) = aT(\mathbf{v}) + bT(\mathbf{w})$$

for all vectors $\mathbf{v}, \mathbf{w} \in \mathbb{R}^n$ and scalars $a, b \in \mathbb{R}$.

Every linear transformation $T$ from $\mathbb{R}^n$ into $\mathbb{R}^m$ can be represented by an $m \times n$ matrix $A$, called the *standard matrix of* $T$, or equivalently the *matrix representation of* $T$. For example, the linear transformation $T : \mathbb{R}^2 \to \mathbb{R}^3$ given by

$$T(\mathbf{v}) = \begin{bmatrix} 2x + y \\ x - 3y \\ y \end{bmatrix} \tag{1}$$

has the matrix representation

$$A = \begin{bmatrix} 2 & 1 \\ 1 & -3 \\ 0 & 1 \end{bmatrix}$$

because for all vectors $\mathbf{x}$ in $\mathbb{R}^2$ we have $T(\mathbf{x}) = A\mathbf{x}$ (check this yourself for several different vectors in $\mathbb{R}^2$).

Applying the linear transformation $T$ to a vector $\mathbf{x}$ is therefore the same as multiplying $\mathbf{x}$ on the left by the matrix representation of $T$. This results in a new vector $\mathbf{x}'$, where each component of $\mathbf{x}'$ is some linear combination of the components of $\mathbf{x}$. For linear transformations from $\mathbb{R}^2$ to $\mathbb{R}^2$, this process takes the form

$$A\mathbf{x} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax + by \\ cx + dy \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{x}'.$$

*Problem* 1. Define a function `transform(x)` which takes as input a NumPy vector `x` in $\mathbb{R}^2$, performs the linear transformation $T$ from (1) above to `x`, and returns the resulting vector as a NumPy vector.

Check that your function works by using it to compute `transform([1,2])` which should return `array([ 4, -5, 2])`.

## 2. Linear transformations in $\mathbb{R}^2$

Linear transformations can be interpreted geometrically. It is easy to observe the effect of a linear transformation $T$ on vectors in the plane by plugging an image into $T$ and observing how the resulting image differs from the original. To demonstrate this we will use a data set contained in the file `cougar.csv`. The file `cougar.csv` contains a list of points of the form $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$. When each of these points are plotted we see the image of a cougar and the BYU 'Y' (see the top left image in Figure 2). This is a very crude example of a vector graphics format like the ones mentioned above.

Instead of arranging the data in `cougar.csv` as a list of points, we could instead organize it as a single $2 \times n$ matrix $H$, where each column of $H$ contains the $x$- and $y$-coordinates of a single data point. More precisely, the $j$th column of $H$ will contain the $x$-coordinate $x_j$ of the $j$th point in the first row, with the corresponding $y$-coordinate $y_j$ in the second row. This gives us a matrix of the form

$$H = \begin{bmatrix} x_1 & x_2 & \dots & x_n \\ y_1 & y_2 & \dots & y_n \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_n \end{bmatrix}.$$

Matrix multiplication on the left of $H$ transforms each point in our image (i.e. each column of $H$), resulting in another matrix $H'$ whose columns are the transformed coordinate pairs:

$$\begin{aligned} AH = A\begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \end{bmatrix} &= A\begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \end{bmatrix} = \begin{bmatrix} A\mathbf{x}_1 & A\mathbf{x}_2 & \cdots & A\mathbf{x}_n \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{x}'_1 & \mathbf{x}'_2 & \cdots & \mathbf{x}'_n \end{bmatrix} = \begin{bmatrix} x'_1 & x'_2 & \cdots & x'_n \\ y'_1 & y'_2 & \cdots & y'_n \end{bmatrix} = H'. \end{aligned}$$

Thus multiplying the image matrix $H$ by a $2 \times 2$ matrix $A$ on the left will result in a new image matrix $H'$, which will correspond to a transformed image.

- Import data from `cougar.csv` into a python array called `cougar`. (See the instructions on how to do this in the notebook for this lab.) If you get an error message try importing again (the file doesn't seem to completely upload on the first try sometimes).
- Plot the image using the function `showplot(H)` which has already been defined in your notebook. Compare the result of calling `showplot(cougar)` with the image in Figure 2.

**Types of Linear Transformations.** Linear transformations from $\mathbb{R}^2$ into $\mathbb{R}^2$ fall into several different families:

- **Stretch**: Transformations which stretch or compress the vector along the standard coordinate axes. The matrix representation of such linear transformations are diagonal:
$$\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}.$$
If $a = b$, the transformation is called a *dilation*. The stretch in Figure 2 uses $a = \frac{1}{2}$ to compress the plane along the $x$-axis and $b = \frac{6}{5}$ to stretch it along the $y$-axis.
- **Shear**: Transformations which slant vectors in the plane by a scalar factor horizontally or vertically (or both simultaneously). The matrix representations of such linear transformations are of the form:
$$\begin{bmatrix} 1 & a \\ b & 1 \end{bmatrix}.$$
Pure horizontal shears (i.e. $b = 0$) only skew the $x$-coordinate of the vector, while pure vertical shears (i.e. $a = 0$) only skew the $y$-coordinate. Figure 2 has a horizontal shear with $a = \frac{1}{2}, b = 0$.
- **Reflection**: Transformations which reflect vectors about a line that passes through the origin. The reflection around the line spanned by the vector $[a, b]^T$ has the matrix

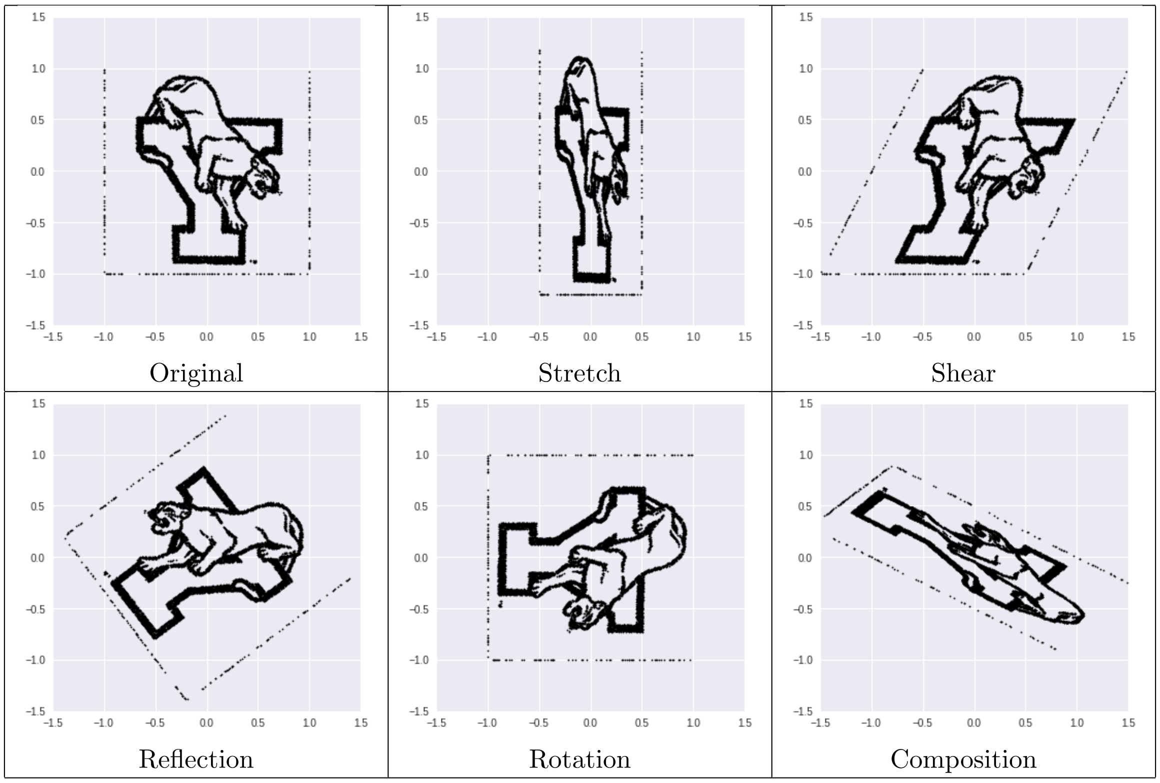

FIGURE 2. Images of the data in the array `cougar` under various linear transformations.

representation

$$\frac{1}{a^2+b^2}\begin{bmatrix} a^2-b^2 & 2ab \\ 2ab & b^2-a^2 \end{bmatrix}.$$

The reflection in Figure 2 reflects the image about the line spanned by $\left[\frac{1}{2}, 1\right]^T$ (i.e. $a = \frac{1}{2}$, $b = 1$).

- **Rotation**: Transformations which rotate vectors around the origin. A counterclockwise rotation of $\theta$ radians has the following matrix representation:

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

A negative value of $\theta$ performs a clockwise rotation. Choosing $\theta = \frac{3\pi}{2}$ produces the rotation in Figure 2.

*Problem* 2. Define the following four functions: `stretch(image,a,b)`, `shear(image,a,b)`, `reflect(image,a,b)`, and `rotate(image,theta)` which correspond to the families of linear transformations defined above.

Each function should accept as input a $2 \times n$ array `image` as above, as well as the parameters needed to define the transformation (`a` and `b` for stretch, shear, and reflection, and `theta` for rotation). The functions should then return the array which is obtained from `image` by applying the corresponding linear transformation.

Test your functions by applying the transformations defined above, plotting them using the function `showplot`, and comparing your results to Figure 2.

**Compositions of Linear Transformations.** If $L : \mathbb{R}^p \to \mathbb{R}^n$ and $K : \mathbb{R}^n \to \mathbb{R}^m$ are linear transformations with standard matrices $A$ and $B$ respectively, then the *composition* function $K \circ L : \mathbb{R}^p \to \mathbb{R}^m$ is also a linear transformation, defined by

$$K \circ L(\mathbf{x}) = K(L(\mathbf{x})) \quad \text{for all} \quad \mathbf{x} \in \mathbb{R}^p.$$

The standard matrix of $K \circ L$ is the matrix product $BA$.

For example, if $S$ is a matrix representing a shear and $R$ is a matrix representing a rotation, then $RS$ represents a shear followed by a rotation. While not every linear transformation from $\mathbb{R}^2$ to $\mathbb{R}^2$ is one of the four types described above, any linear transformation $L : \mathbb{R}^2 \to \mathbb{R}^2$ can be written as a composition of these four types of transformations. In fact, once we learn about the *singular value decomposition* we will see that every linear transformation from $\mathbb{R}^2$ to $\mathbb{R}^2$ can be written as a rotation, followed by a stretch, followed by another rotation. The bottom right image in Figure 2 displays the composition of all four previous transformations, applied in order (stretch, shear, reflection, then rotation).

*Problem* 3. Use the functions you defined in the Problem 2 to construct a single matrix which performs the following sequence of transformations (in order) when acting on an image.

(1) Stretch the image vertically by a factor of 2.
(2) Rotate the image $\frac{\pi}{4}$ radians clockwise.
(3) Reflect the image over the line $2y = -3x$.
(4) Stretch the image horizontally by a factor of $1/2$.

save the resulting $2 \times 2$ matrix as `comp_matrix`.

*Hint:* Instead of rewriting out the matrix representations of each type of function again, notice (for example) that the standard matrix of a shear with $a = 3$ and $b = 0.2$ can be obtained by calling `shear(iden,3,0.2)`, where `iden` is the $2 \times 2$ identity matrix. The same is true for the functions `stretch`, `reflect`, and `rotate` for any values of the parameters $a$, $b$, and $\theta$.

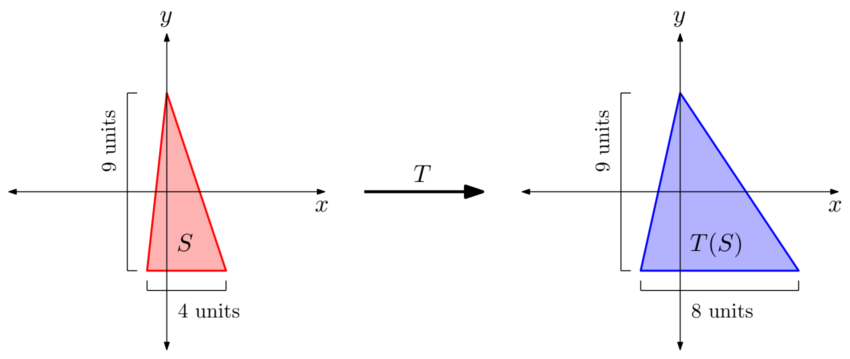

FIGURE 3. The effect of applying the linear transformation $T$ defined by the matrix (2) to a triangle in the plane. The triangle is stretched by a factor of 2 in the $x$-direction, and kept constant in the $y$-direction.

## 3. Linear transformations and area

We'd like to understand how linear transformations deform shapes in the plane $\mathbb{R}^2$. Consider for example the transformation $T(\mathbf{x}) = A\mathbf{x}$ defined by the matrix

$$A = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}. \tag{2}$$

How does this transformation affect the area of a shape $S$ (such as a circle or polygon) in $\mathbb{R}^2$? Notice that since

$$T\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2x \\ y \end{bmatrix},$$

the transformation $T$ stretches objects by a factor of 2 in the $x$-direction, and leaves the position of points fixed in the $y$-direction. Thus if we plug a shape $S$ into the linear transformation $T$ (or rather, if we plug all of the vectors contained in the shape $S$ into $T$) then we get a new shape $T(S)$ whose area is twice as large as the area of $S$ (see Figure 3). Similarly, if we plug $S$ into a linear transformation defined by the matrix

$$B = \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix},$$

we would obtain an object whose area is 6 times the original area of $S$ (scaled by a factor of 2 in the $x$-direction, and a factor of 3 in the $y$-direction).

Both of the examples mentioned above were defined by diagonal matrices (and were thus examples of stretches), which is why we were able to determine how they changed the area of figures in the plane. If our transformation $T$ is defined by a matrix that is not diagonal it is more difficult to see how $T$ will scale the area. We will investigate this problem for the remainder of this lab.

We begin by observing that linear transformations always send straight lines to either straight lines or points. Indeed, a line through the point $\mathbf{p} \in \mathbb{R}^2$ parallel to the vector $\mathbf{d} \in \mathbb{R}^2$ can be expressed using an equation in vector form

$$\mathbf{x} = \mathbf{p} + s\mathbf{d},$$

where $s$ is a scalar. Since $T$ is linear, by plugging both sides of the equation into $T$, we obtain

$$\mathbf{x}' = T(\mathbf{x}) = T(\mathbf{p} + s\mathbf{d}) = T(\mathbf{p}) + sT(\mathbf{d}).$$

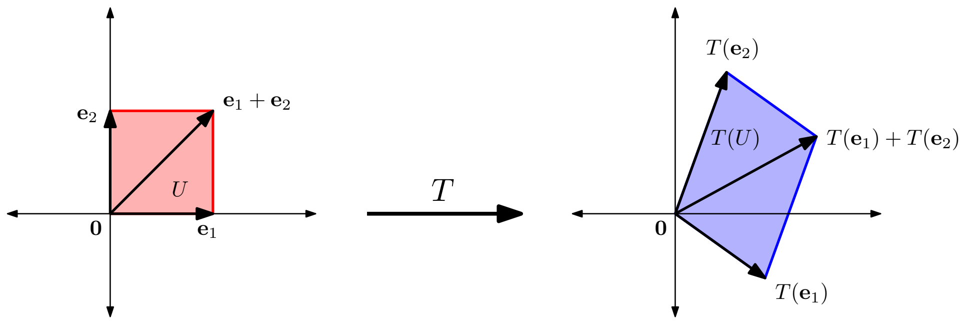

FIGURE 4. The effect of applying the linear transformation $T$ to the unit square $U$.

If $T(\mathbf{d}) = \mathbf{0}$ then this just gives the point $T(\mathbf{p})$; otherwise this is the equation of a line through $T(\mathbf{p})$ in the direction of $T(\mathbf{d})$.

Consider now a unit square $U$ in $\mathbb{R}^2$ with corners at the points

$$\mathbf{0} = \begin{bmatrix}0\\0\end{bmatrix}, \qquad \mathbf{e}_1 = \begin{bmatrix}1\\0\end{bmatrix}, \qquad \mathbf{e}_2 = \begin{bmatrix}0\\1\end{bmatrix}, \qquad \text{and} \qquad \mathbf{e}_1+\mathbf{e}_2 = \begin{bmatrix}1\\1\end{bmatrix}.$$

Since $T$ maps straight lines to straight lines, $T(U)$ will be a parallelogram with corners at

$$T(\mathbf{0}) = \mathbf{0}, \qquad T(\mathbf{e}_1), \qquad T(\mathbf{e}_2), \qquad \text{and} \qquad T(\mathbf{e}_1+\mathbf{e}_2) = T(\mathbf{e}_1)+T(\mathbf{e}_2)$$

(see Figure 4).

To help visualize these transformations, we've defined a function `plot_transformation(mat)` in the lab notebook. It takes as input a $2\times 2$ NumPy array `mat`, and plots both the unit square (in red) and its image $T(U)$ (in blue), where $T$ is the linear transformation defined by `mat`.

- Copy the code for the function `plot_transformation(mat)` to your practice notebook, and evaluate it on the matrices
$$M = \begin{bmatrix}2 & 1\\-2 & 4\end{bmatrix}, \quad N = \begin{bmatrix}3 & 1\\2 & -1\end{bmatrix}, \quad P = \begin{bmatrix}-1 & 2\\-1 & -1\end{bmatrix}, \quad \text{and} \quad Q = \begin{bmatrix}0.6 & -0.5\\0.2 & -0.5\end{bmatrix}.$$
- Create a number of $2\times 2$ matrices on your own, and plot them using the function `plot_transformation`. Try examples of stretch, shear, reflection, and rotation matrices like those discussed above.

We return now to the problem of understanding how areas change under a linear transformation $T$. Note that if we can compute the area of $T(U)$ then we can compare this value to the area of $U$ to see how $T$ scales area. For example, if $T(U)$ has area 4 then we know that $T$ quadruples the area of shapes in the plane, since the original area of $U$ was 1.

An easy calculation involving trigonometry shows that the area of a parallelogram with edges of length $a$ and $b$, and angle $\theta$ as shown in Figure 5 is given by

$$\text{Area} = ab\sin\theta.$$

We will use this formula in the following problem to compute the area of $T(U)$.

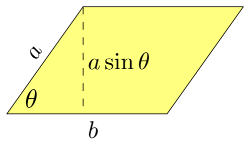

FIGURE 5. A parallelogram with edge lengths $a$ and $b$, and area equal to $ab\sin\theta$.

*Problem* 4. Define a function `area(A)` which accepts as input a $2 \times 2$ NumPy array `A`, and returns the area of the parallelogram $T(U)$. Here $U$ is the unit square and $T$ is the linear transformation defined by $T(\mathbf{x}) = A\mathbf{x}$.

*Hint: Recall that the edges of $T(U)$ are given by the vectors $T(\mathbf{e}_1)$ and $T(\mathbf{e}_2)$. Your function will need to find the length of these edges, as well as the angle between them. Recall that the angle $\theta$ between vectors $\mathbf{v}$ and $\mathbf{w}$ satisfies the equation*

$$\cos\theta = \frac{\mathbf{v}\cdot\mathbf{w}}{\|\mathbf{v}\|\|\mathbf{w}\|}.$$

*You may find the NumPy functions `np.linalg.norm` and `np.arccos` helpful when writing your function.*

To test your function, evaluate the function `area(D)`, where `D=np.array([[1,3],[2,-1]])`. It should return the value `7.0000000000000009`.

- Let `M`, `N`, `P`, and `Q` be the $2 \times 2$ matrices defined by

$$M = \begin{bmatrix} 2 & 1 \\ -2 & 4 \end{bmatrix}, \quad N = \begin{bmatrix} 3 & 1 \\ 2 & -1 \end{bmatrix}, \quad P = \begin{bmatrix} -1 & 2 \\ -1 & -1 \end{bmatrix}, \quad \text{and} \quad Q = \begin{bmatrix} 0.6 & -0.5 \\ 0.2 & -0.5 \end{bmatrix}.$$

  Compute the output of the function `area` you defined in Problem 4 on the matrices `M`, `N`, `P`, and `Q`.

You may have already noticed, but the areas you just computed are related to the determinants of the matrices `M`, `N`, `P`, and `Q`.

*Problem* 5. Compute the determinants of the matrices

$$M = \begin{bmatrix} 2 & 1 \\ -2 & 4 \end{bmatrix}, \quad N = \begin{bmatrix} 3 & 1 \\ 2 & -1 \end{bmatrix}, \quad P = \begin{bmatrix} -1 & 2 \\ -1 & -1 \end{bmatrix}, \quad \text{and} \quad Q = \begin{bmatrix} 0.6 & -0.5 \\ 0.2 & -0.5 \end{bmatrix}.$$

and save these values as variables `detM`, `detN`, `detP`, and `detQ` respectively. You may compute them in your head or using a pencil and paper, but save their values in your lab notebook.

The relations between the areas of the transformed parallelograms and the determinants of the standard matrices $M, N, P$, and $Q$ in the above example are no coincidence. In fact, if $T(\mathbf{x}) = A\mathbf{x}$ is a linear transformation, then it scales the area of any shape in the plane by a

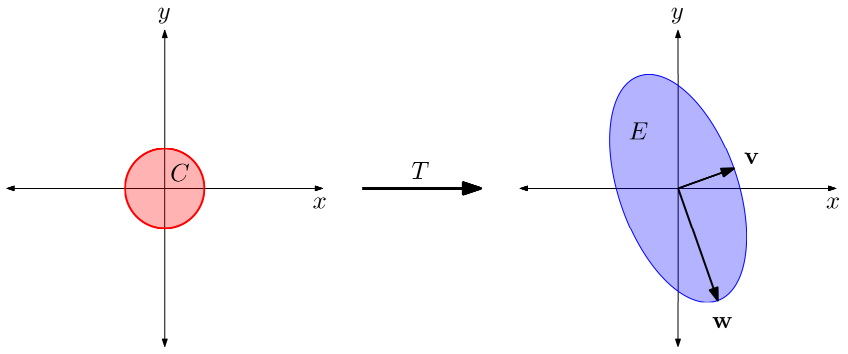

FIGURE 6. An ellipse $E$, thought of as the image of the unit circle $C$ under a linear transformation.

factor of $|\det A|$ (i.e. the absolute value of the determinant of $A$). We record this fact in the following theorem:

**Theorem.** *Let $S$ be a region in the plane $\mathbb{R}^2$, and let $T : \mathbb{R}^2 \to \mathbb{R}^2$ be a linear transformation defined by $T(\mathbf{x}) = A\mathbf{x}$ for some $2 \times 2$ matrix $A$. If $S$ has area $m$, then the area of $T(S)$ is $|\det(A)| \cdot m$.*

This theorem is true in higher dimensions as well. For example, a $3 \times 3$ matrix $A$ scales volumes in $\mathbb{R}^3$ by a factor of $|\det A|$.

We can use this theorem to find the area of certain regions in the plane $\mathbb{R}^2$. For example, suppose that we would like to find the area of a region $E$ in the plane. In order to find the area of $E$, we can begin by trying to find a simple region $C$ in the plane, and a linear transformation $T(\mathbf{x}) = A\mathbf{x}$ which takes $C$ and sends it to $E$. If we can compute the area of $C$ easily, then we can then compute the area of $E$ using the formula

$$\text{Area of } E = |\det A| \cdot (\text{Area of } C).$$

For example, suppose that we want to find the area of the ellipse $E$ shown on the right-hand side of Figure 6. Here the vectors $\mathbf{v}$ and $\mathbf{w}$ are given by

$$\mathbf{v} = \begin{bmatrix} 1.5 \\ 0.5 \end{bmatrix} \qquad \text{and} \qquad \mathbf{w} = \begin{bmatrix} 1 \\ -2.75 \end{bmatrix}.$$

As the ellipse $E$ can be obtained from the unit circle by stretching and rotating, we can take our simple shape to be the unit circle $C$ centered at the origin. Because we can compute the area of $C$ very easily, all we need to do to find the area of $E$ is to find a linear transformation $T : \mathbb{R}^2 \to \mathbb{R}^2$ with $T(C) = E$.

*Problem* 6. Find a $2 \times 2$ matrix $A$, such that the linear transformation $T : \mathbb{R}^2 \to \mathbb{R}^2$ defined by $T(\mathbf{x}) = A\mathbf{x}$ sends the unit circle $C$ to the ellipse $E$. Save the matrix $A$ as a $2 \times 2$ NumPy array called `ellipse_matrix`. Then use the matrix $A$ to compute the area of $E$, and save its value as a variable `ellipse_area`.

*Hint: Recall that the standard matrix of a linear transformation from $\mathbb{R}^2$ to $\mathbb{R}^2$ is given by*

$$A = \begin{bmatrix} T(\mathbf{e}_1) & T(\mathbf{e}_2) \end{bmatrix}.$$

*Where should the transformation $T$ send the standard basis vectors $\mathbf{e}_1$ and $\mathbf{e}_2$?*

## 4. Non-invertible transformations

So far every linear transformation $T : \mathbb{R}^2 \to \mathbb{R}^2$ we have explored in this lab has been *invertible*, meaning there exists a linear transformation $T^{-1} : \mathbb{R}^2 \to \mathbb{R}^2$ such that

$$T(T^{-1}(\mathbf{x})) = \mathbf{x} \qquad \text{and} \qquad T^{-1}(T(\mathbf{x})) = \mathbf{x}$$

for all $\mathbf{x} \in \mathbb{R}^2$. This is equivalent to saying that the standard matrix $A$ for the linear transformation $T(\mathbf{x}) = A\mathbf{x}$ is an invertible $2 \times 2$ matrix. We will now explore the situation when $T$ is not invertible, i.e. when the matrix $A$ is not an invertible $2 \times 2$ matrix.

- Before proceeding, you may find it helpful to review the Fundamental Theorem of Invertible Matrices (sometimes called the Invertible Matrix Theorem) and make sure that you are familiar with the different conditions that are equivalent to an $n \times n$ being invertible.

Consider the matrices

$$B_1 = \begin{bmatrix} \mathbf{4} & 3 \\ 2 & 6 \end{bmatrix}, \quad B_2 = \begin{bmatrix} \mathbf{2} & 3 \\ 2 & 6 \end{bmatrix}, \quad B_3 = \begin{bmatrix} \mathbf{1.5} & 3 \\ 2 & 6 \end{bmatrix}, \quad B_4 = \begin{bmatrix} \mathbf{1.1} & 3 \\ 2 & 6 \end{bmatrix},$$

$$B_5 = \begin{bmatrix} \mathbf{1.001} & 3 \\ 2 & 6 \end{bmatrix}, \quad B_6 = \begin{bmatrix} \mathbf{1.00001} & 3 \\ 2 & 6 \end{bmatrix}, \quad \text{and} \quad B_0 = \begin{bmatrix} \mathbf{1} & 3 \\ 2 & 6 \end{bmatrix}.$$

We will define linear transformations $T_0, T_1, \ldots, T_6$ by

$$T_1(\mathbf{x}) = B_1\mathbf{x}, \quad T_2(\mathbf{x}) = B_2\mathbf{x}, \quad T_3(\mathbf{x}) = B_3\mathbf{x}, \quad T_4(\mathbf{x}) = B_4\mathbf{x},$$

$$T_5(\mathbf{x}) = B_5\mathbf{x}, \quad \text{and} \quad T_6(\mathbf{x}) = B_6\mathbf{x}, \quad \text{and} \quad T_0(\mathbf{x}) = B_0\mathbf{x}.$$

Each of these will be a linear transformation from $\mathbb{R}^2$ to $\mathbb{R}^2$. As the $B_j$ matrices match in all but the top left entry, we think of $T_1, T_2, \ldots, T_6$ as giving us a sequence of linear transformations that are getting closer and closer to the linear transformation $T_0$.

- In your practice notebook, plot the transformations defined by the matrices $B_1$ through $B_6$, and the matrix $B_0$ using the function `plot_transformation` (copying and pasting your code would be helpful here).
- What happens to the area of the transformed parallelogram as the top left entry of the matrix $B_n$ gets closer and closer to 1? Verify your observations by plugging the matrices into the `area` function you defined in Problem 4.
- What does this tell you about the determinants of these matrices?

Notice that because the unit square $U$ sits between the vectors $\mathbf{e}_1$ and $\mathbf{e}_2$, the transformed parallelograms $T_j(U)$ will sit between the vectors $T_j(\mathbf{e}_1)$ and $T_j(\mathbf{e}_2)$ (you may find it helpful to refer back to Figure 4). Furthermore, $T_j(\mathbf{e}_1)$ and $T_j(\mathbf{e}_2)$ are the columns of the matrix $B_j$. Thus as the columns of $B_j$ become closer and closer to positive scalar multiples of each other, $T_j(\mathbf{e}_1)$ and $T_j(\mathbf{e}_2)$ will become closer and closer to lying on the same line. This in turn implies that the parallelogram $T_j(U)$ will become more and more narrow (as it's being squished between $T_j(\mathbf{e}_1)$ and $T_j(\mathbf{e}_2)$). Thus the area of $T_j(U)$ gets smaller and smaller, as the determinants of the matrices $B_1, B_2, \ldots, B_6$ get closer and closer to zero.

*Problem* 7. Compute the determinant of the matrices $B_1$, $B_3$, $B_6$, and $B_0$, and save them in your lab notebook as variables `detB1`, `detB3`, `detB6`, and `detB0` respectively. You may compute them by hand, or using the built-in NumPy function `np.linalg.det`.

To understand what is going on here, we will consider the *ranges* of the linear transformations $T_1, \ldots, T_6$, and $T_0$. Recall that a vector $\mathbf{y} \in \mathbb{R}^m$ is in the range of a linear transformation $T : \mathbb{R}^n \to \mathbb{R}^m$ if there is a vector $\mathbf{x} \in \mathbb{R}^n$ such that $T(\mathbf{x}) = \mathbf{y}$. Roughly speaking, the range of a linear transformation is the set of all vectors $\mathbf{y}$ in the codomain $\mathbb{R}^m$ that get hit by some vector $\mathbf{x}$ from the domain $\mathbb{R}^n$. Suppose now that $T(\mathbf{x}) = A\mathbf{x}$ for some $m \times n$ matrix $A$. How is the range of $T$ related to the standard matrix $A$?

Notice that $\mathbf{y}$ is in the range of $T$ whenever there is a vector $\mathbf{x}$ in $\mathbb{R}^n$ such that

$$\mathbf{y} = T(\mathbf{x}) = A\mathbf{x} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \cdots & \mathbf{a}_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = x_1\mathbf{a}_1 + x_2\mathbf{a}_2 + \cdots + x_n\mathbf{a}_n.$$

Thus $\mathbf{y}$ is in the range of $T$ precisely when $\mathbf{y}$ can be written as a linear combination of the columns $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$ of $A$ with some coefficients $x_1, x_2, \ldots, x_n$. In other words, the range of the linear transformation $T(\mathbf{x}) = A\mathbf{x}$ is equal to the column space $\mathrm{Col}(A)$ of $A$. Thus for any subset $S$ of $\mathbb{R}^n$, the transformed subset $T(S)$ will always sit inside $\mathrm{Col}(A)$.

Returning now to the $B_j$ matrices above, note that each of the matrices $B_1, B_2, \ldots, B_6$ (not including $B_0$) are invertible, and hence when you row-reduce them you will find that each has a pivot in both rows. Hence the equation $T_j(\mathbf{x}) = \mathbf{y}$ has *at least one solution* for all $\mathbf{y}$ in the codomain $\mathbb{R}^2$. This implies that the columns of $B_j$ span all of $\mathbb{R}^2$, or in other words,

$$\mathrm{range}(T_j) = \mathrm{Col}(B_j) = \mathbb{R}^2 \quad \text{for each} \quad j = 1, 2, \ldots, 6.$$

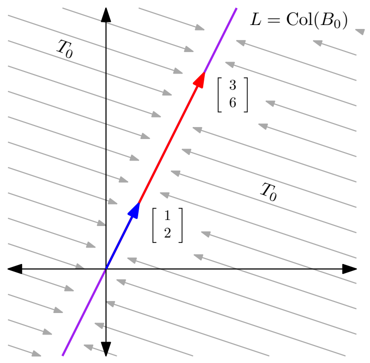
$L = \mathrm{Col}(B_0)$
$T_0$
$\begin{bmatrix}3\\6\end{bmatrix}$
$T_0$
$\begin{bmatrix}1\\2\end{bmatrix}$

FIGURE 7. The linear transformation $T_0$ takes the entire plane $\mathbb{R}^2$ and sends it to the line $L = \mathrm{Col}(B_0)$ following the directions of the grey arrows.

Geometrically, this implies that every vector in the codomain $\mathbb{R}^2$ gets hit by some vector in the domain $\mathbb{R}^2$ (such transformations are called *onto*).

Because there is a pivot in every column of $B_j$, the equation $T_j(\mathbf{x}) = \mathbf{y}$ also has *at most one solution* for all $\mathbf{y}$ in the codomain $\mathbb{R}^2$ (such transformations are called *one-to-one*). Combining these two observations, this implies that every vector in the codomain $\mathbb{R}^2$ gets hit by *exactly one vector* from the domain $\mathbb{R}^2$. Thus a 2-dimensional shape $S$ in the domain $\mathbb{R}^2$ will be transformed into another 2-dimensional shape $T_j(S)$ in the codomain $\mathbb{R}^2$, even though that 2-dimensional shape might be deformed.

Now consider what happens with the linear transformation $T_0(\mathbf{x}) = B_0\mathbf{x}$. The columns of the matrix $B_0$ are scalar multiples of each other, so it follows that

$$\mathrm{range}(T_0) = \mathrm{Col}(B_0) = \mathrm{Span}\left\{\begin{bmatrix}1\\2\end{bmatrix}, \begin{bmatrix}3\\6\end{bmatrix}\right\} = \mathrm{Span}\left\{\begin{bmatrix}1\\2\end{bmatrix}\right\},$$

which is just a line $L$ in $\mathbb{R}^2$. The transformation $T_0$ thus takes the entire domain $\mathbb{R}^2$ (including the unit square $U$) and crushes it down to the line $L$ (see Figure 7). The transformed unit square $T_0(U)$ is now no longer a parallelogram, but rather a segment of the line $L = \mathrm{Col}(B_0)$. The area of $T_0(U)$ is now zero, which corresponds to the determinant of $B_0$ being zero.

Unlike the linear transformations $T_1, \ldots, T_6$, the transformation $T_0$ is not onto (since the only vectors in $\mathbb{R}^2$ that are hit by $T_0$ are on the line $L$), and is not one-to-one (since each of the vectors on $L$ are hit by infinitely many vectors from the domain $\mathbb{R}^2$). The reason the transformation $T_0$ behaves so differently from the transformations $T_1, T_2, \ldots, T_6$ is that the matrix $B_0$ is not invertible, while the other matrices $B_1, \ldots, B_6$ are invertible.