Matrix Algebra

Geometric Algorithms
Lecture 10

CAS CS 132

Practice Problem

Write the matrix for the transformation which projects vectors in $\mathbb{R}^2$ *vertically onto the line* $y = 2x + 3$ *in* $\mathbb{R}^2$.

Answer

Objectives

1. Connect questions about matrix equations and linear transformations
2. Motivate matrix multiplication
3. Define matrix multiplication
4. Look at the algebra of matrix multiplication

Keywords

one-to-one transformation

onto transformation

matrix multiplication

row-column rule

matrix addition and scaling

non-commutativity

Recap: Geometry of Linear Transformations

Recall: Matrices as Transformations

Matrices allow us to *transform* vectors.

The transformed vector lies in the span of its columns.

$$\mathbf{x} \mapsto A\mathbf{x}$$

map a vector $\mathbf{x}$ to the vector $A\mathbf{v}$

Recall: Motivating Questions

What kind of functions can we define in this way?

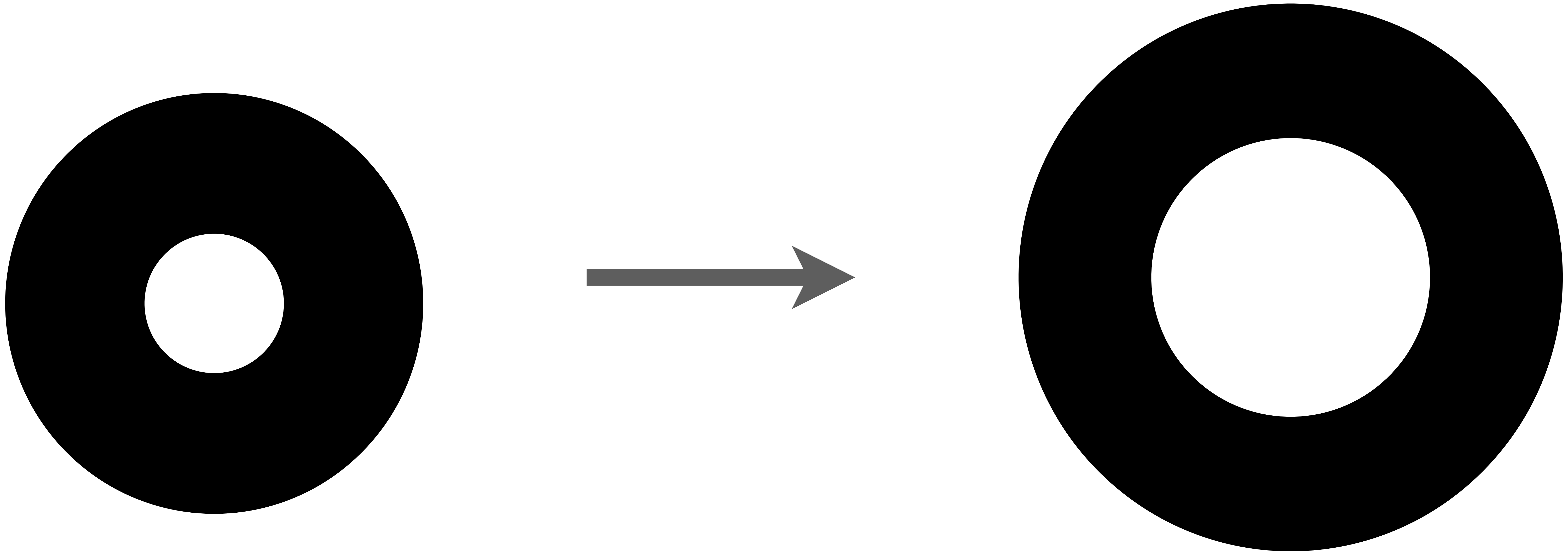
How do we interpret what the transformation does to a set of vectors?

How does this relate back to matrix equations?

Motto

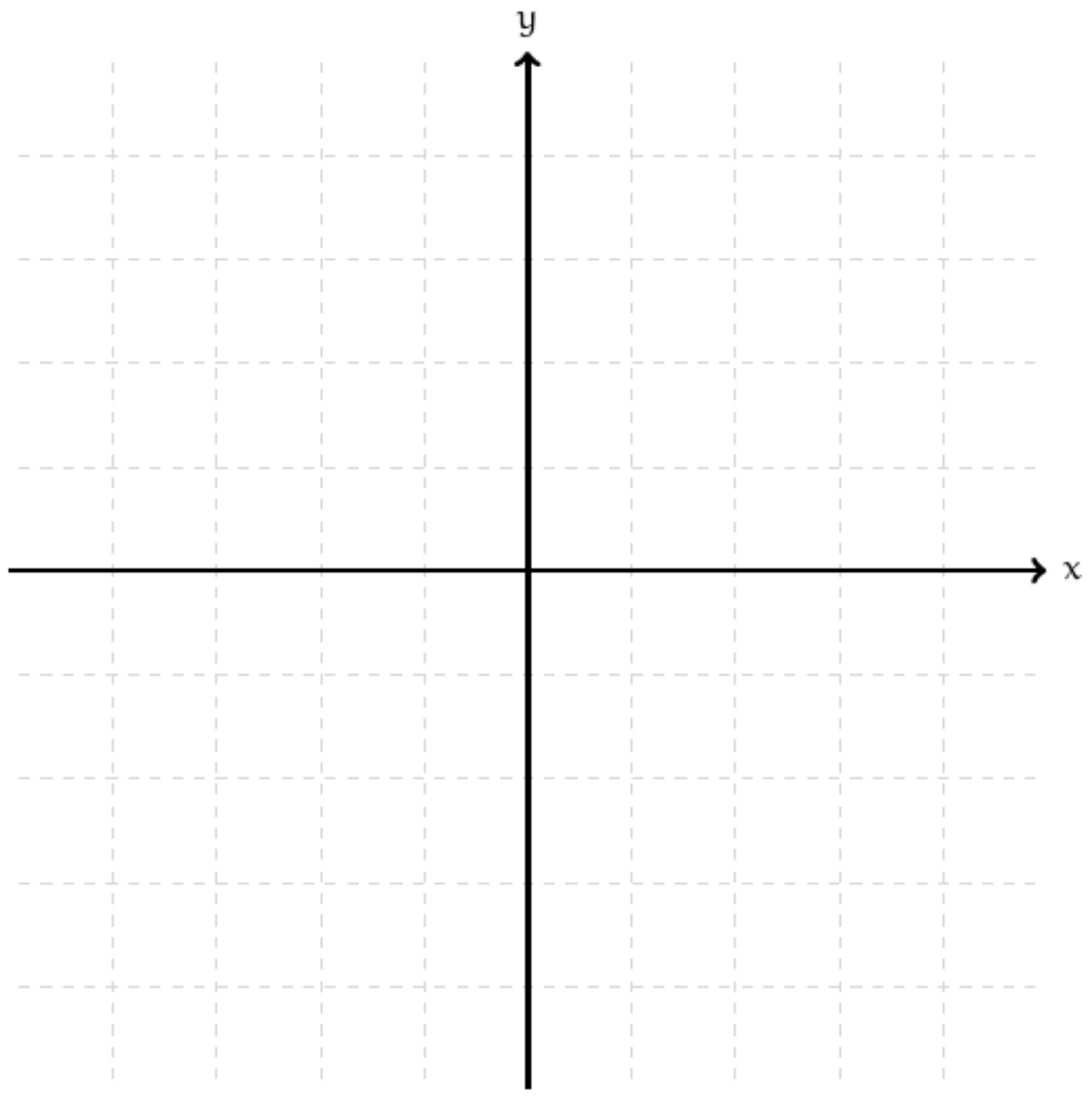
Matrix transformations change the "shape" of a set of set of vectors (points).

Example: Dilation

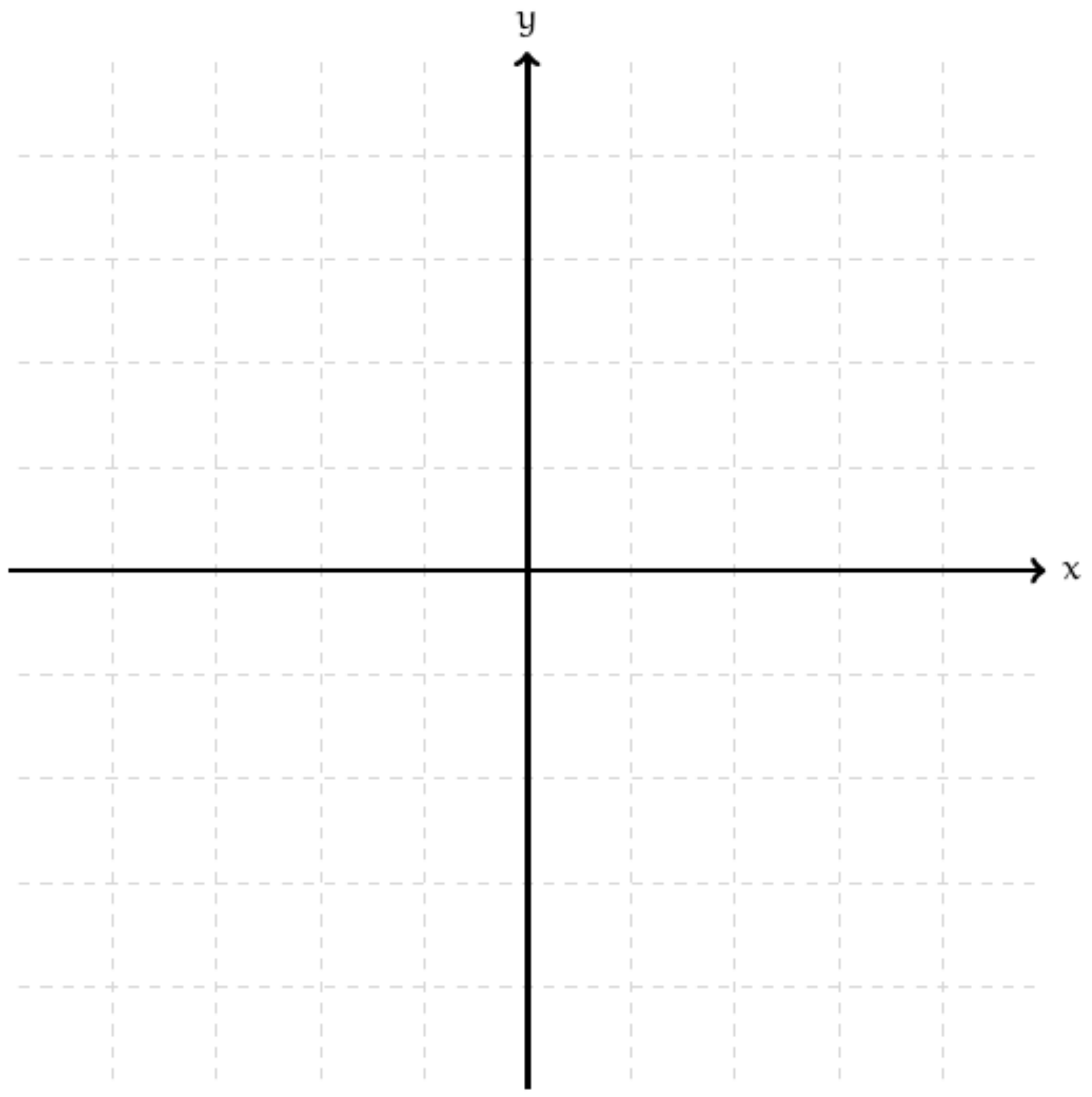


Example: Dilation

$$\begin{bmatrix} r & 0 \\ 0 & r \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} rx_1 \\ rx_2 \end{bmatrix}$$

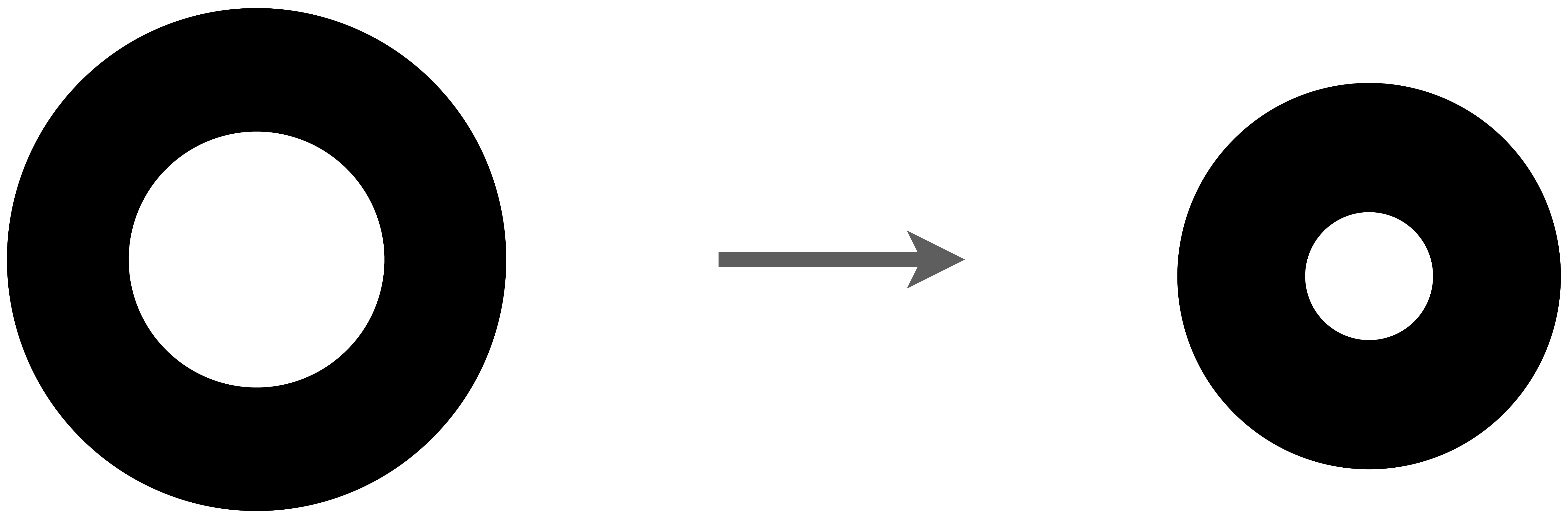


$$\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \mathbf{x}$$



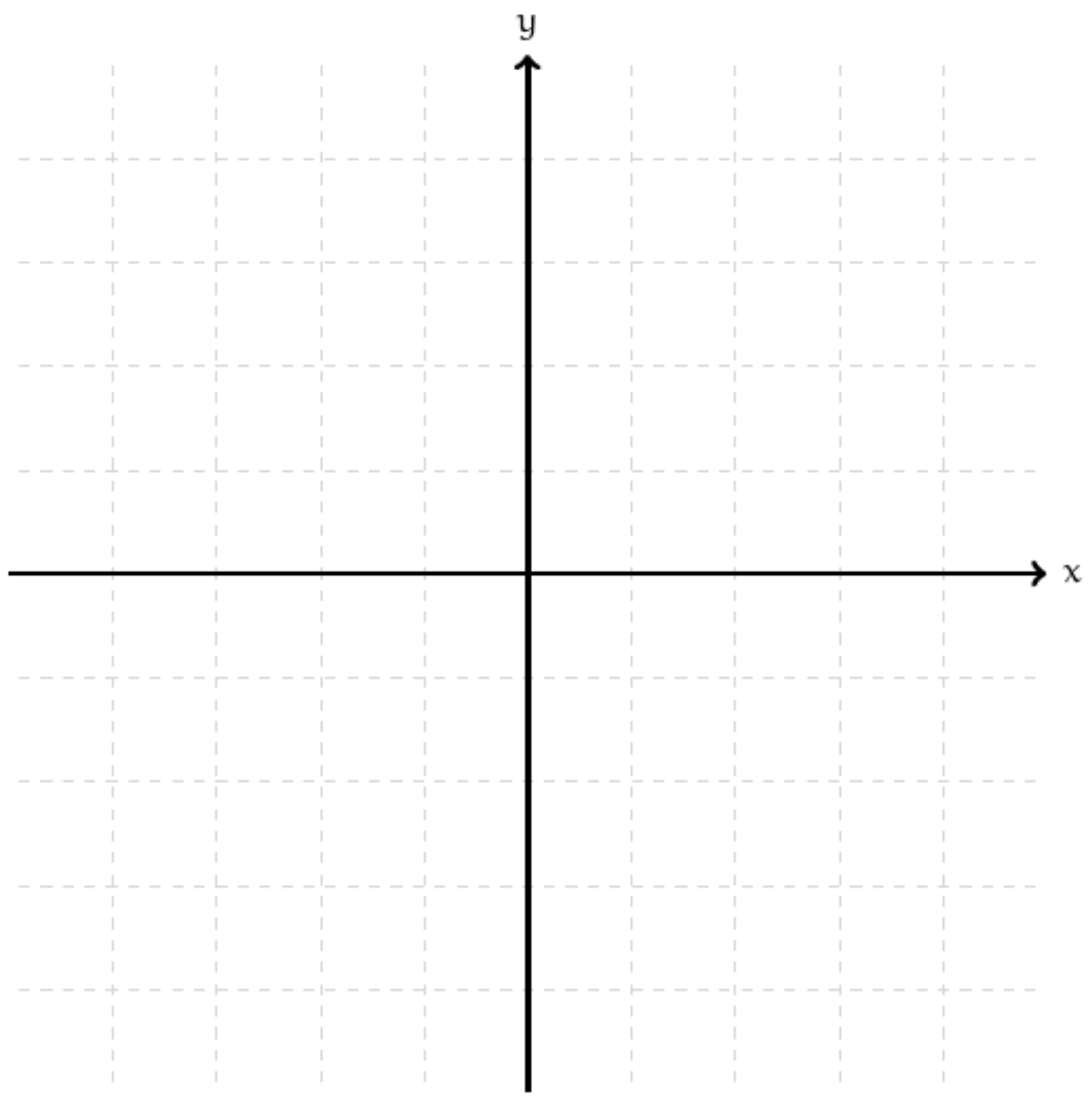
if $r > 1$, then the transformation pushes points away from the origin.

Example: Contraction

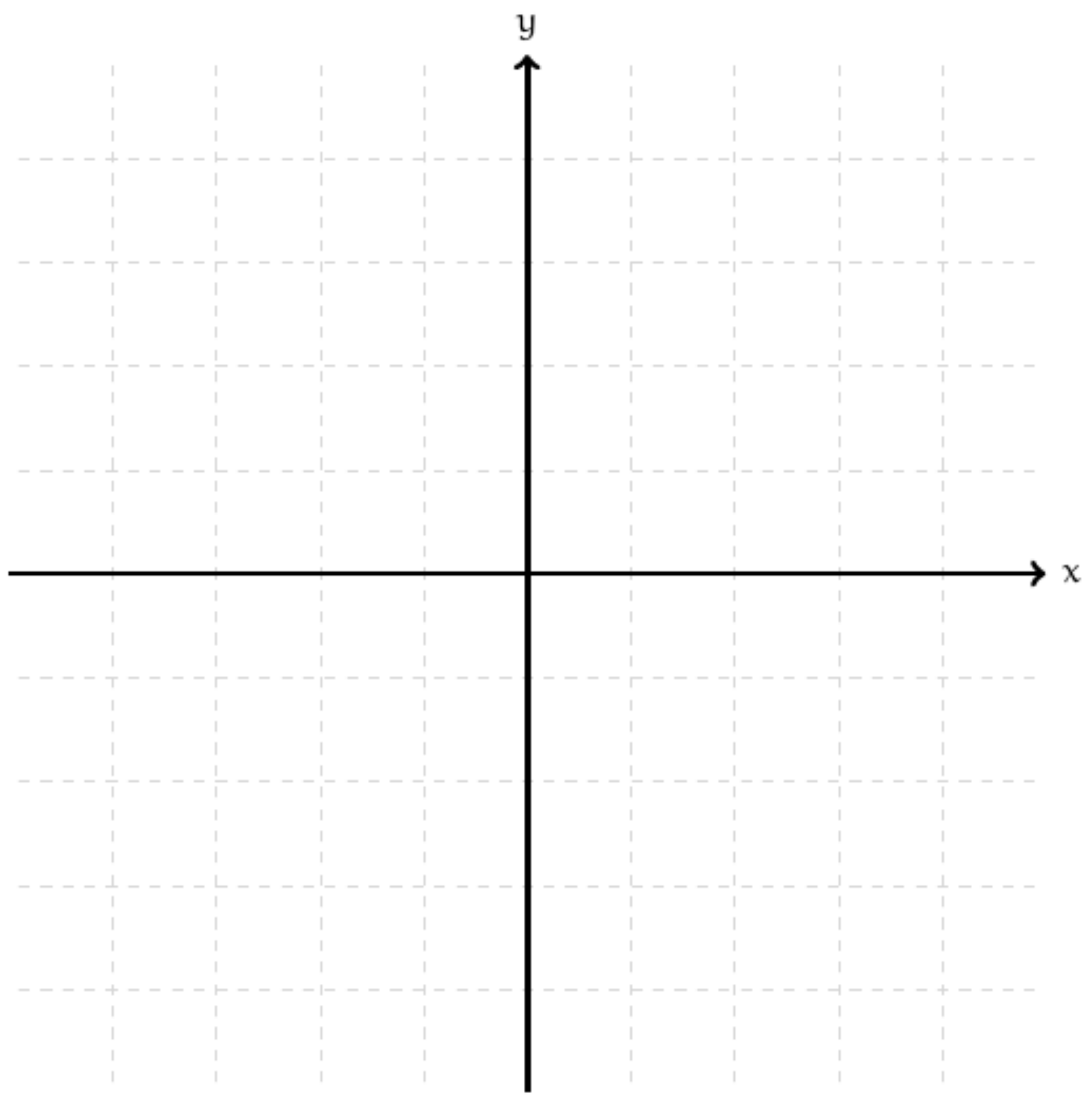


Example: Contraction

$$\begin{bmatrix} r & 0 \\ 0 & r \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} rx_1 \\ rx_2 \end{bmatrix}$$



$$\begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \mathbf{x} \longrightarrow$$

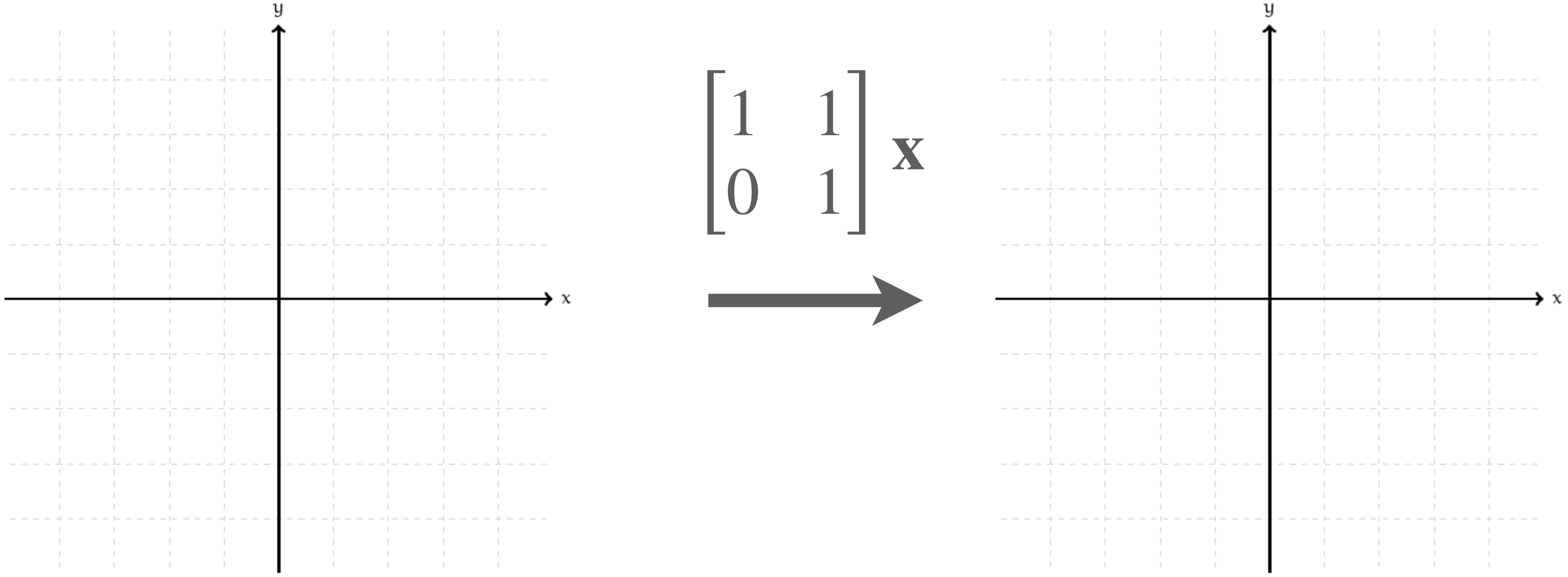


if $0 \leq r \leq 1$, then the transformation pulls points towards the origin.

Example: Shearing

Example: Shearing

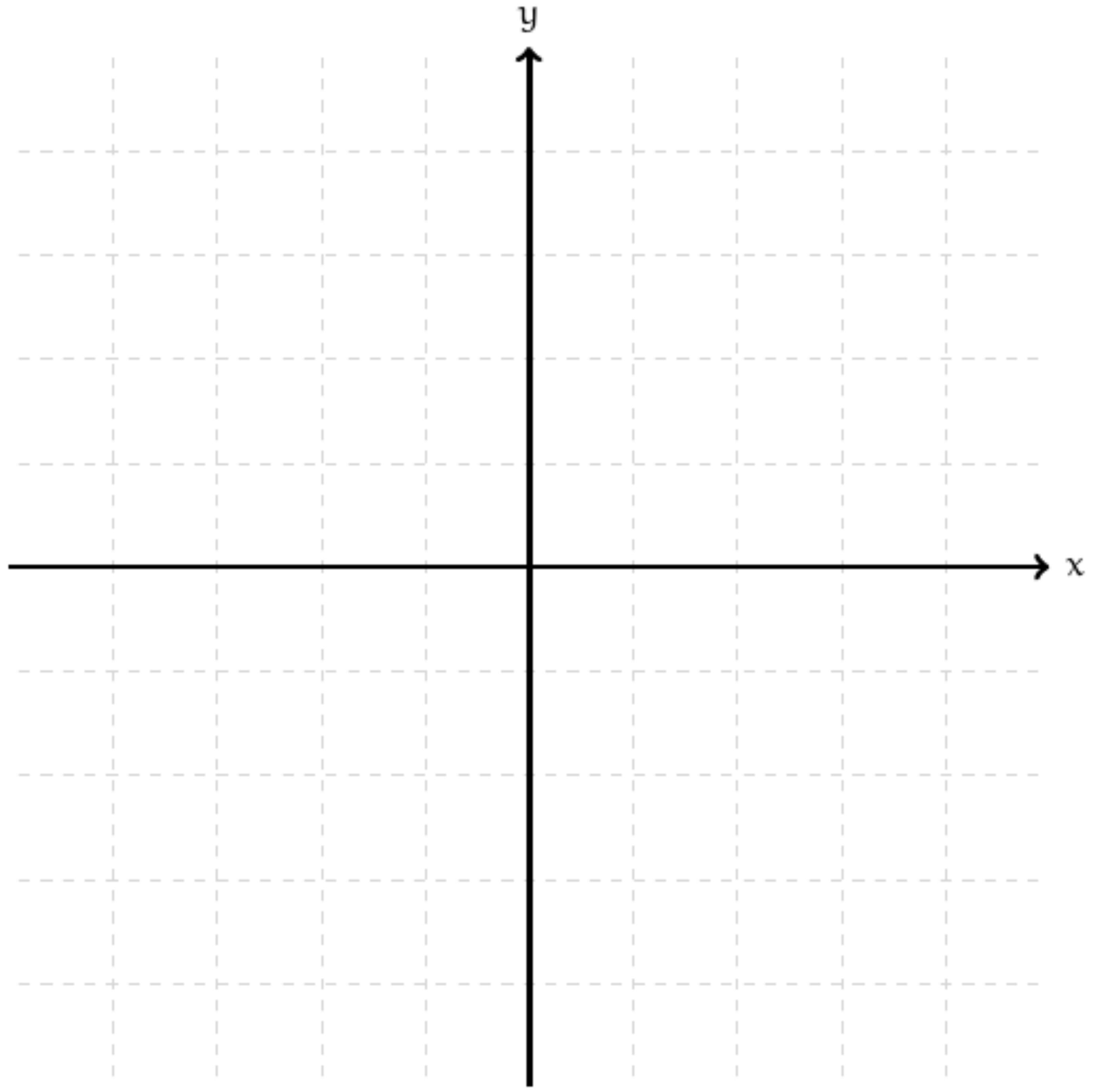
$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 + x_2 \\ x_2 \end{bmatrix}$$



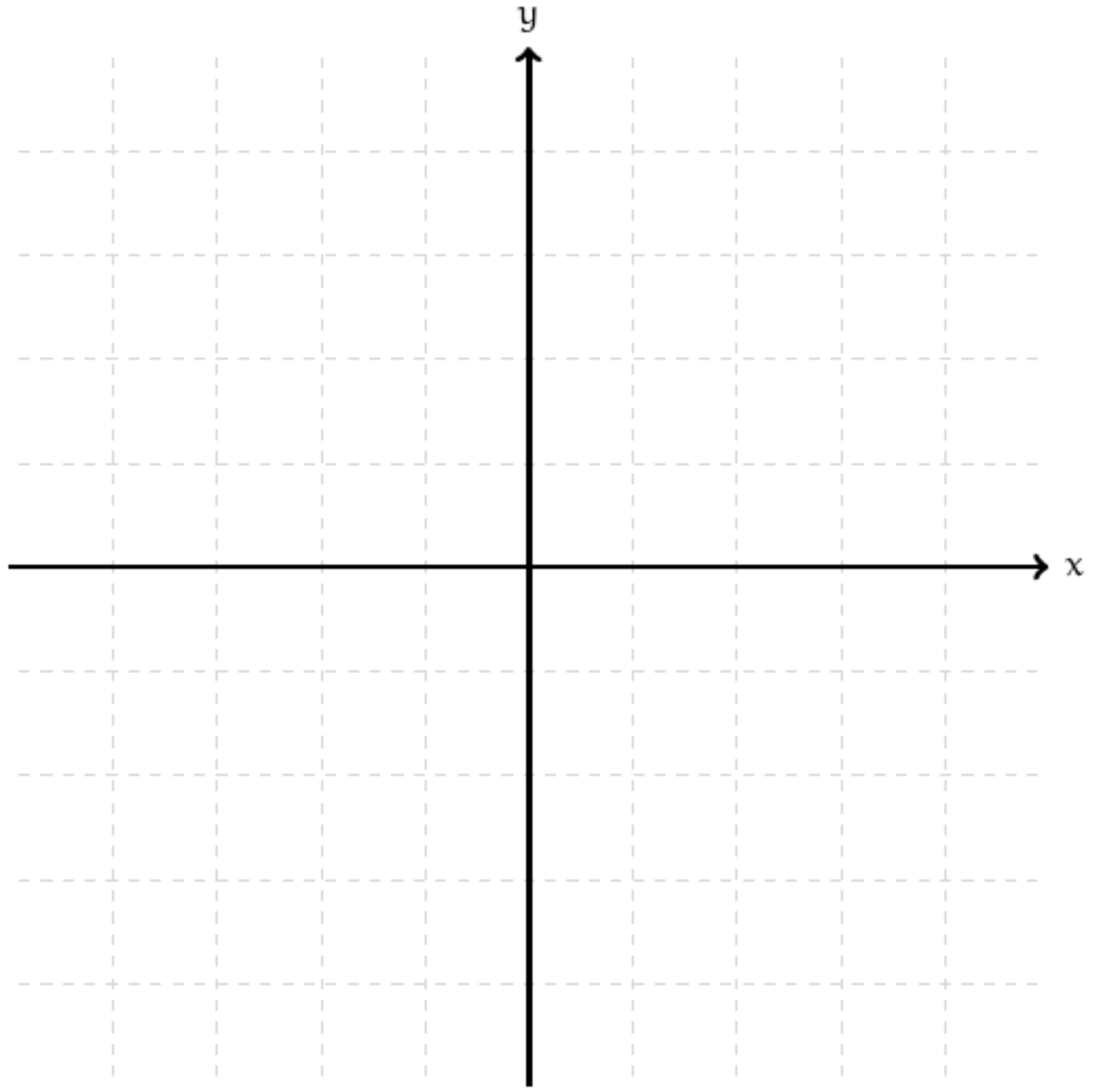
$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \mathbf{x}$$

Imagine shearing like with rocks or metal.

Example: Reflection

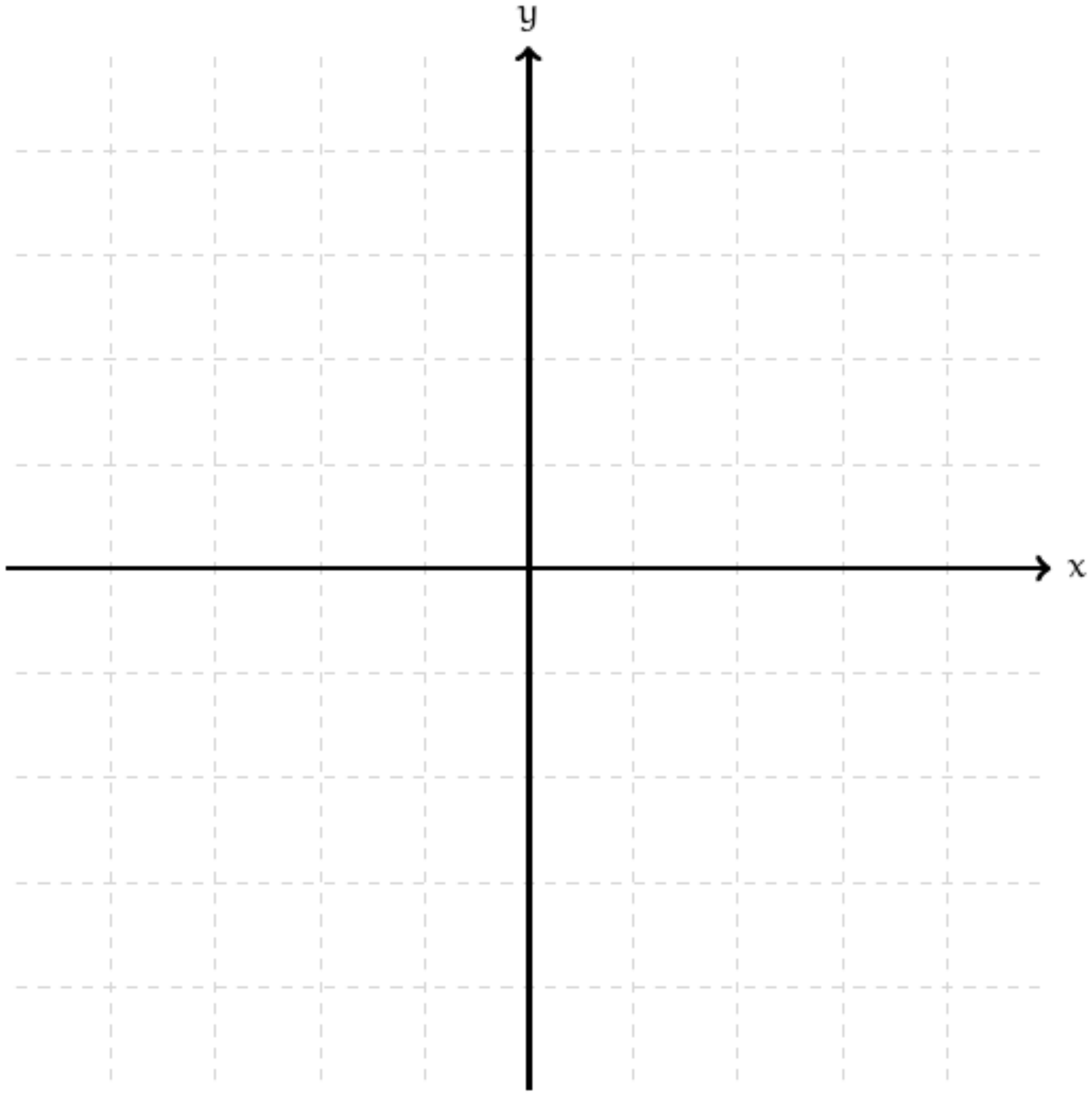


$$\begin{bmatrix}1 & 0\\0 & -1\end{bmatrix}\mathbf{x}$$



General Rotation

How does rotation affect the standard basis?



Rotation Matrix

Rotation Matrix

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

Rotation Matrix

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

Note: This is rotation about the origin.

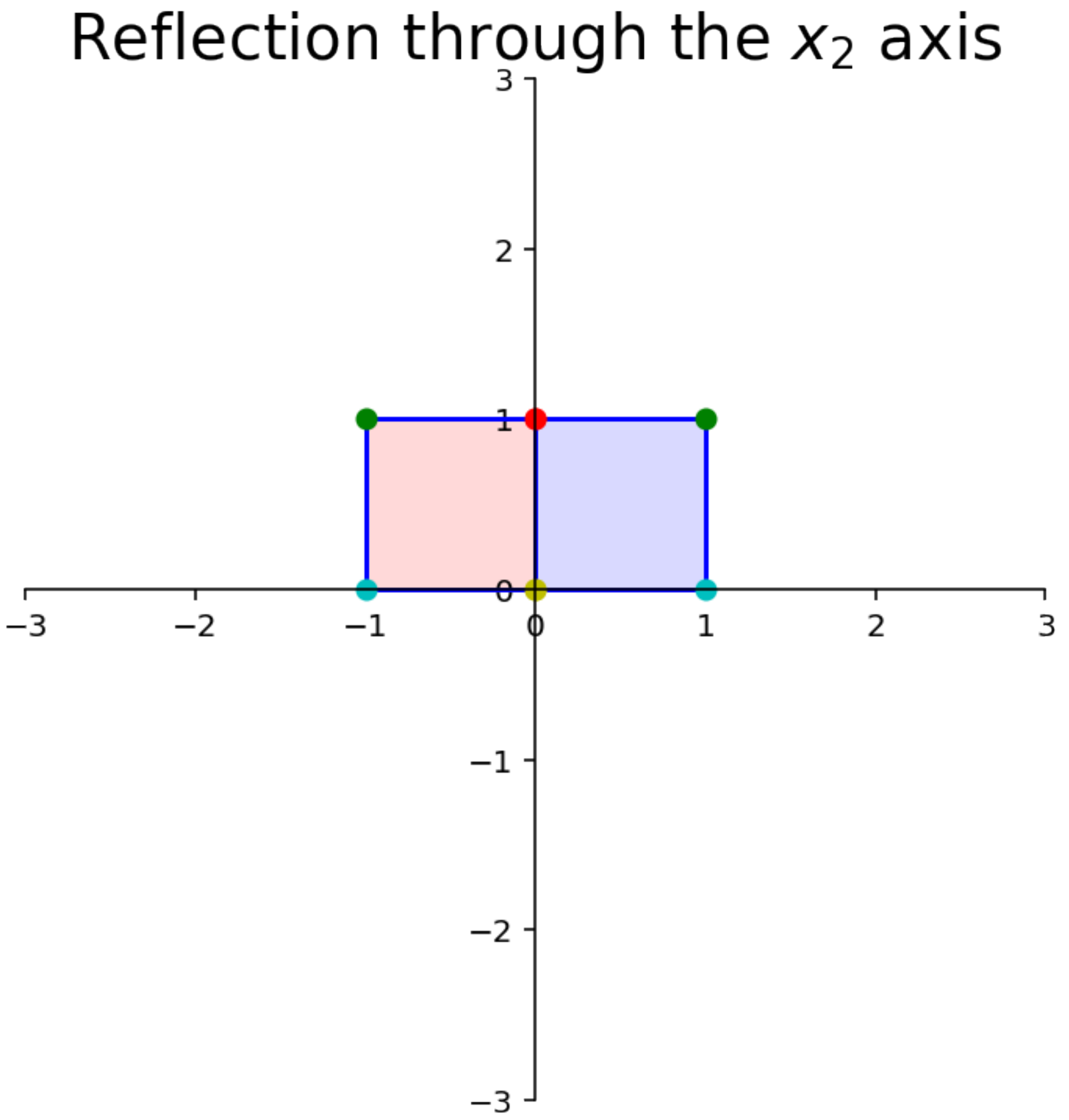
Rotation Matrix

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

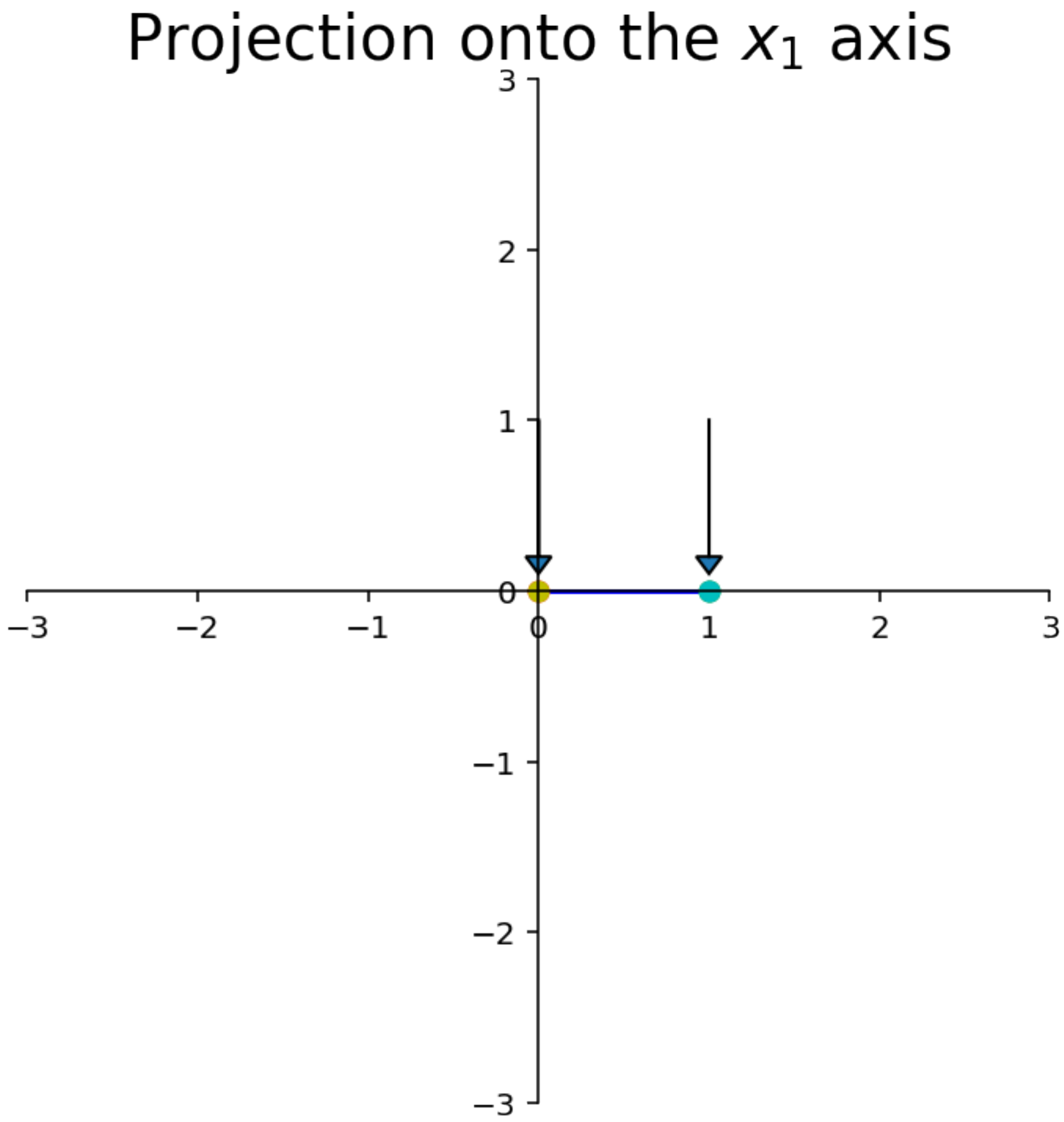
Note: This is rotation about the origin.

The Takeaway: We can figure out the matrices which implement complex linear transformations by understanding what they do to the standard basis.

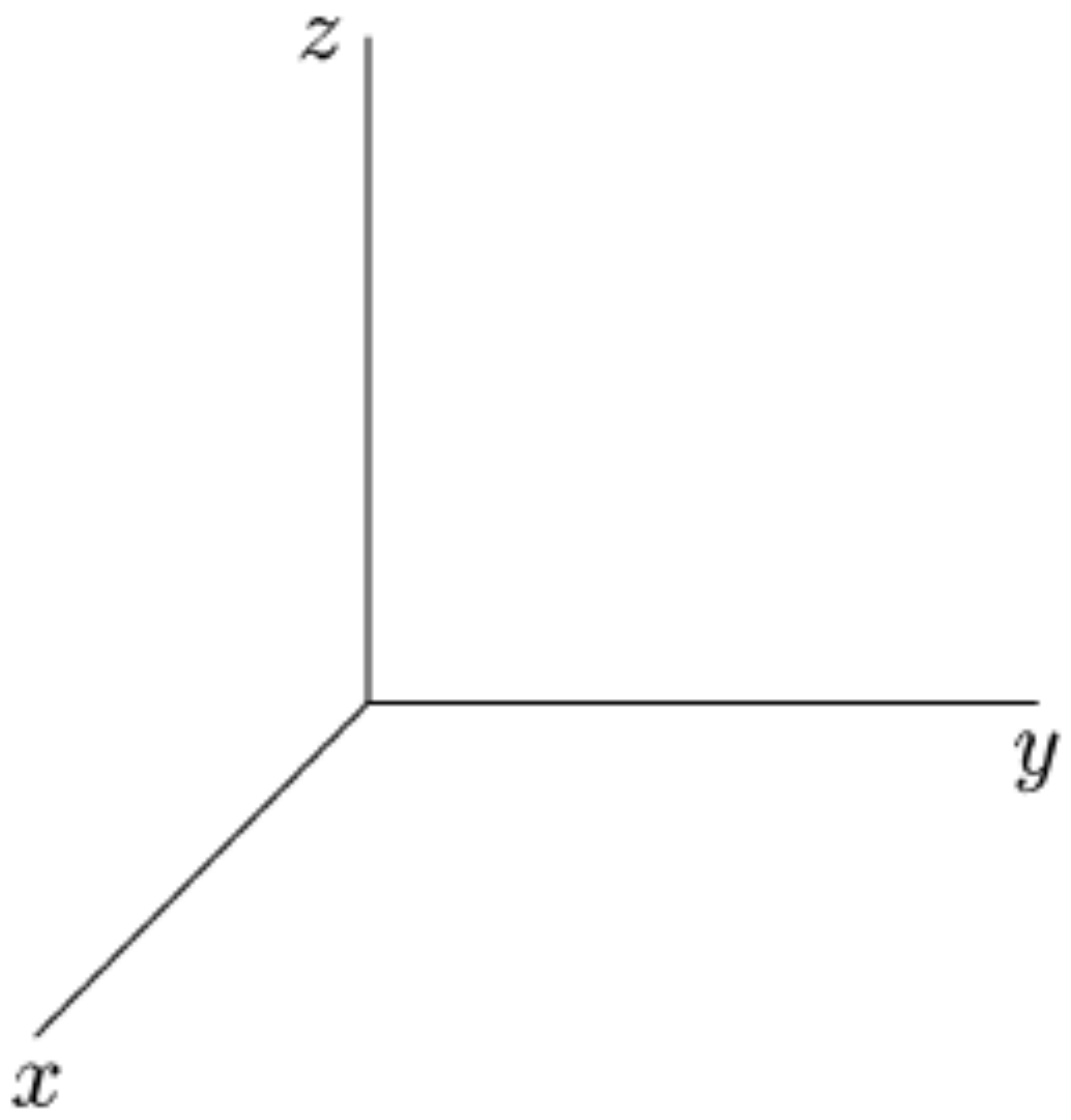
Example: Reflection through the x_2-axis



Example: Projections



3D Example: Rotation about the x_3-Axis (z-Axis)



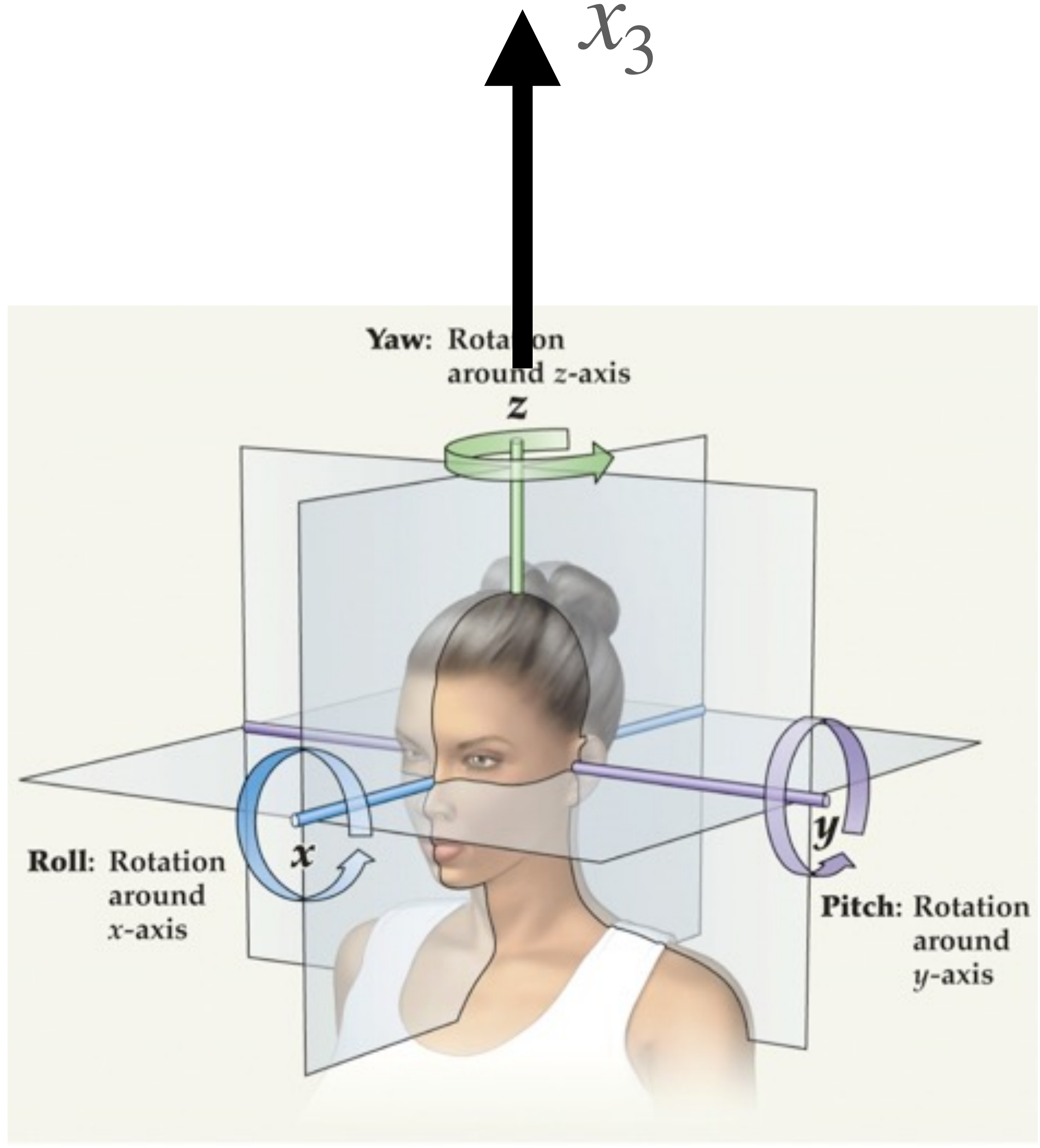
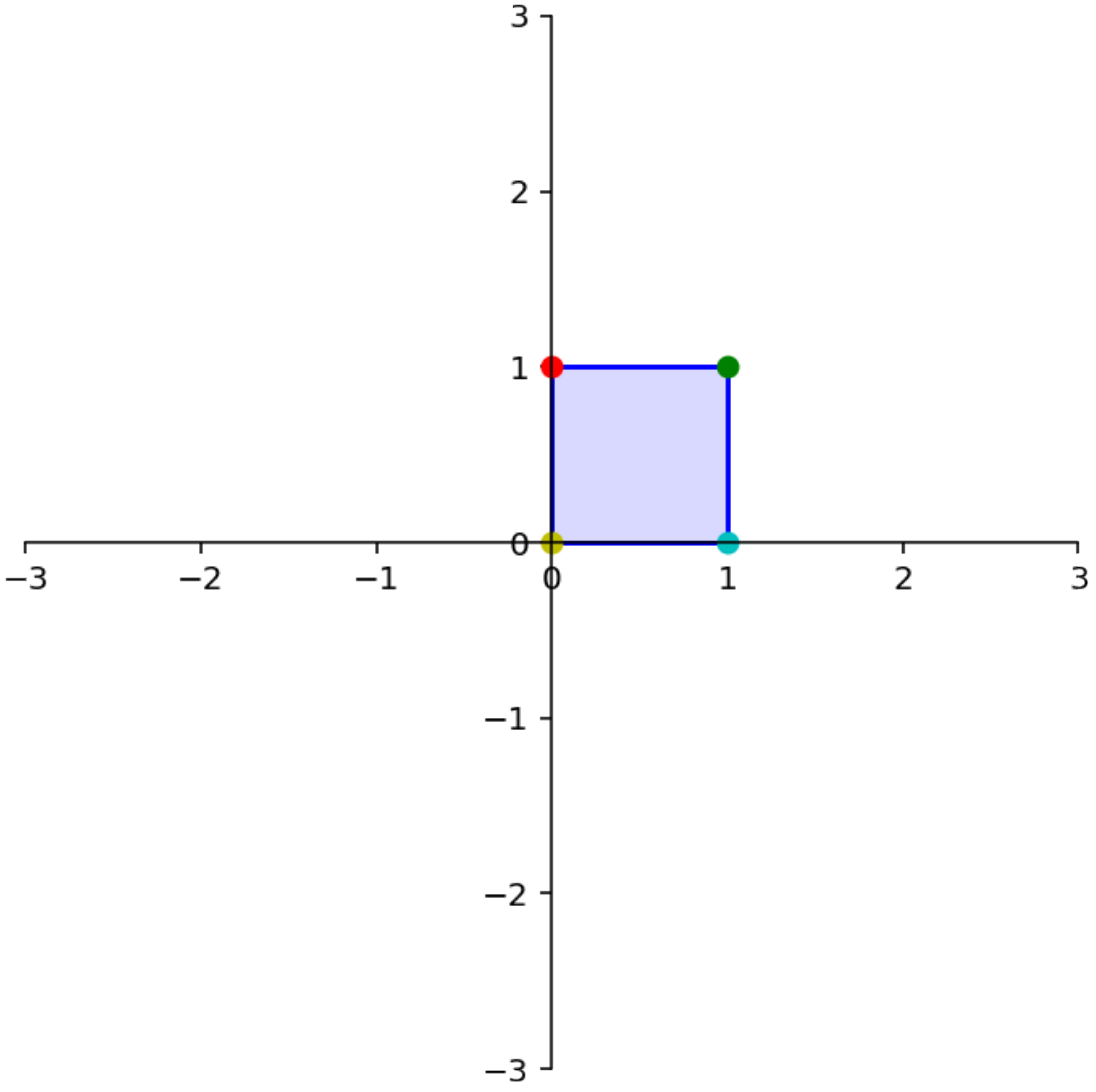


image source

The Unit Square

The ***unit square*** is the set of points in $\mathbb{R}^2$ enclosed by the points $(0,0)$, $(0,1)$, $(1,0)$, $(1,1)$.



How To: The Unit Square and Matrices

How To: The Unit Square and Matrices

Question. Find the matrix which implements the linear transformation which is represented geometrically in the following picture.

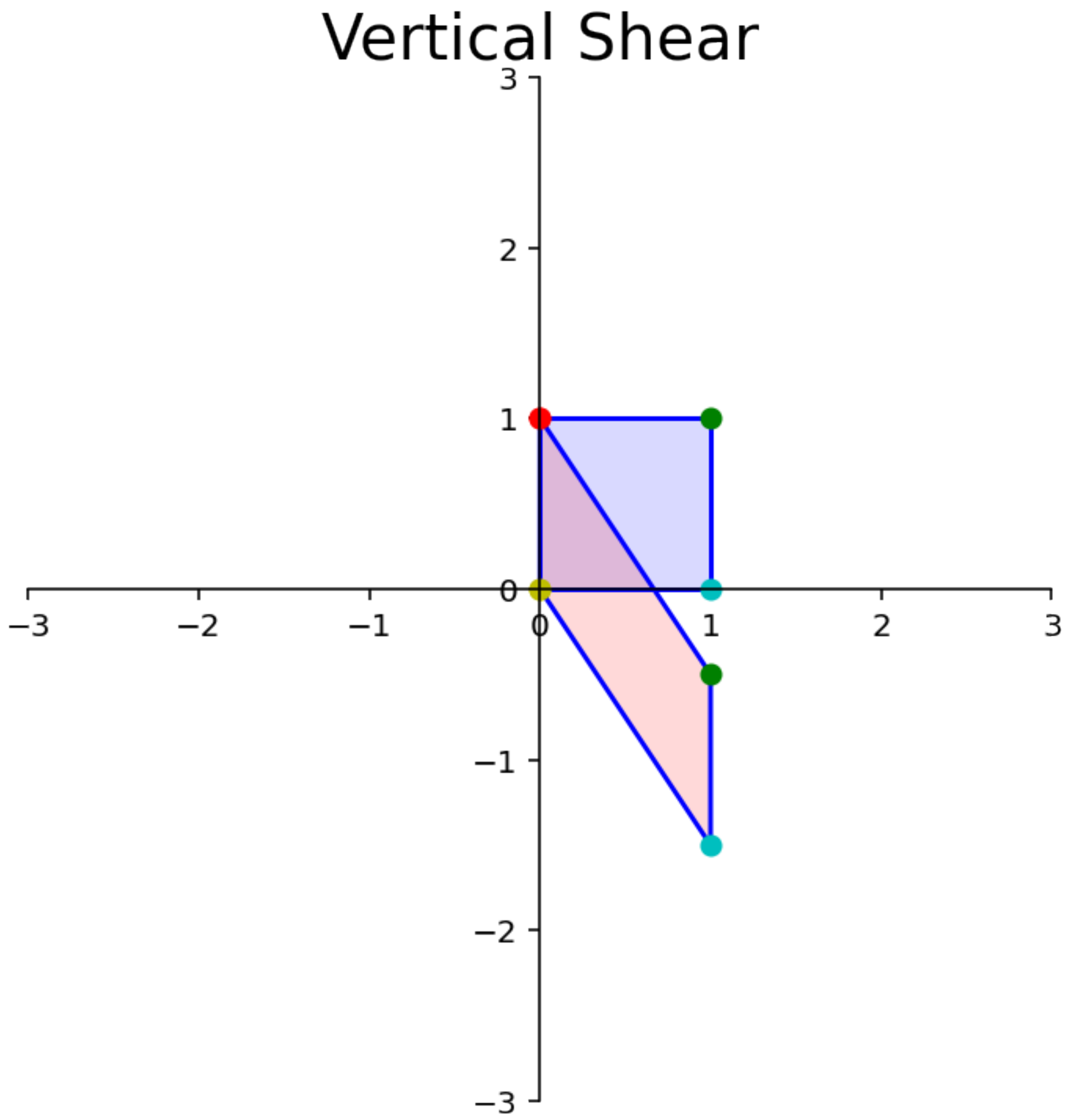
How To: The Unit Square and Matrices

Question. Find the matrix which implements the linear transformation which is represented geometrically in the following picture.

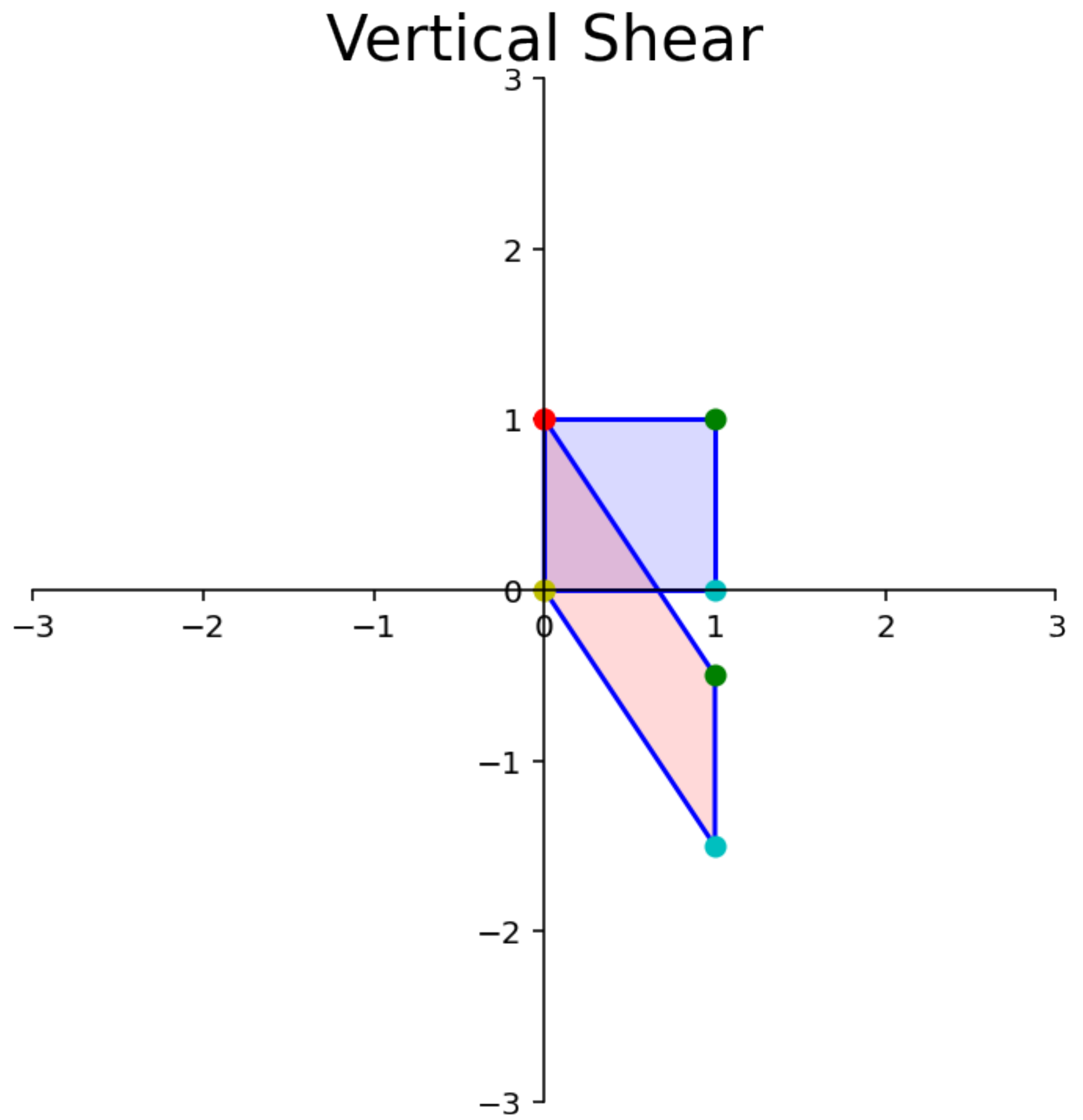
Solution. Find where the standard basis vectors go.

Question

Write down the matrix for the following shearing operation using this method.



Answer



You need to **know** these matrices, but you don't need to memorize them.

Remember: What does this matrix do to the unit square? Then build the matrix from there.

List of Important 2D Linear Transformations

- dilation, contraction
- reflections
- projections
- horizontal/vertical contractions
- horizontal/vertical shearing

Look through the notes for a comprehensive collection of pictures or...

One-to-One and Onto

Recall: Motivating Questions

What kind of functions can we define in this way?

How do we interpret what the transformation does to a set of vectors?

How does this relate back to matrix equations?

Recall: A New Interpretation of the Matrix Equation

$A\mathbf{x} = \mathbf{b}$? $\equiv$ is there a vector which A transforms into $\mathbf{b}$?

Solve $A\mathbf{x} = \mathbf{b}$ $\equiv$ find a vector which A transforms into $\mathbf{b}$

Recall: A New Interpretation of the Matrix Equation

$A\mathbf{x} = \mathbf{b}$?	$\equiv$	is there a vector which A transforms into $\mathbf{b}$?
Solve $A\mathbf{x} = \mathbf{b}$	$\equiv$	find a vector which A transforms into $\mathbf{b}$

What about other questions?

Other Questions Like...

Does $A\mathbf{x} = \mathbf{b}$ have a solution for any choice of $\mathbf{b}$?

Does $A\mathbf{x} = \mathbf{0}$ have a unique solution?

Other Questions Like...

Does $A\mathbf{x} = \mathbf{b}$ have at least one solution for any choice of $\mathbf{b}$?

Does $A\mathbf{x} = \mathbf{b}$ have at most one solution for any choice of $\mathbf{b}$?

Wait

$A\mathbf{x} = \mathbf{0}$ has a unique solution $\equiv$ $A\mathbf{x} = \mathbf{b}$ has at most one solution

why?:

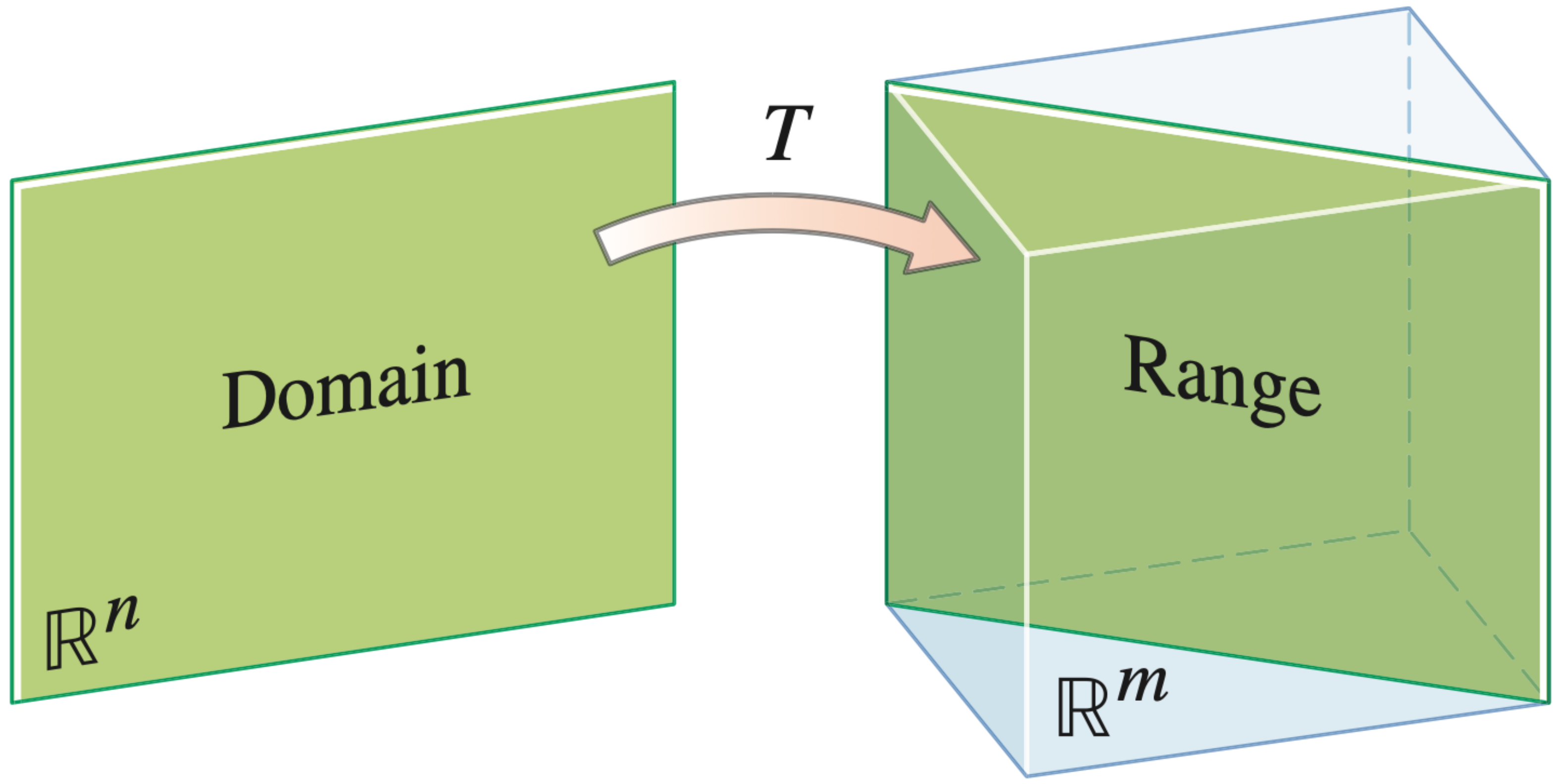
Onto Transformations

Onto Transformations

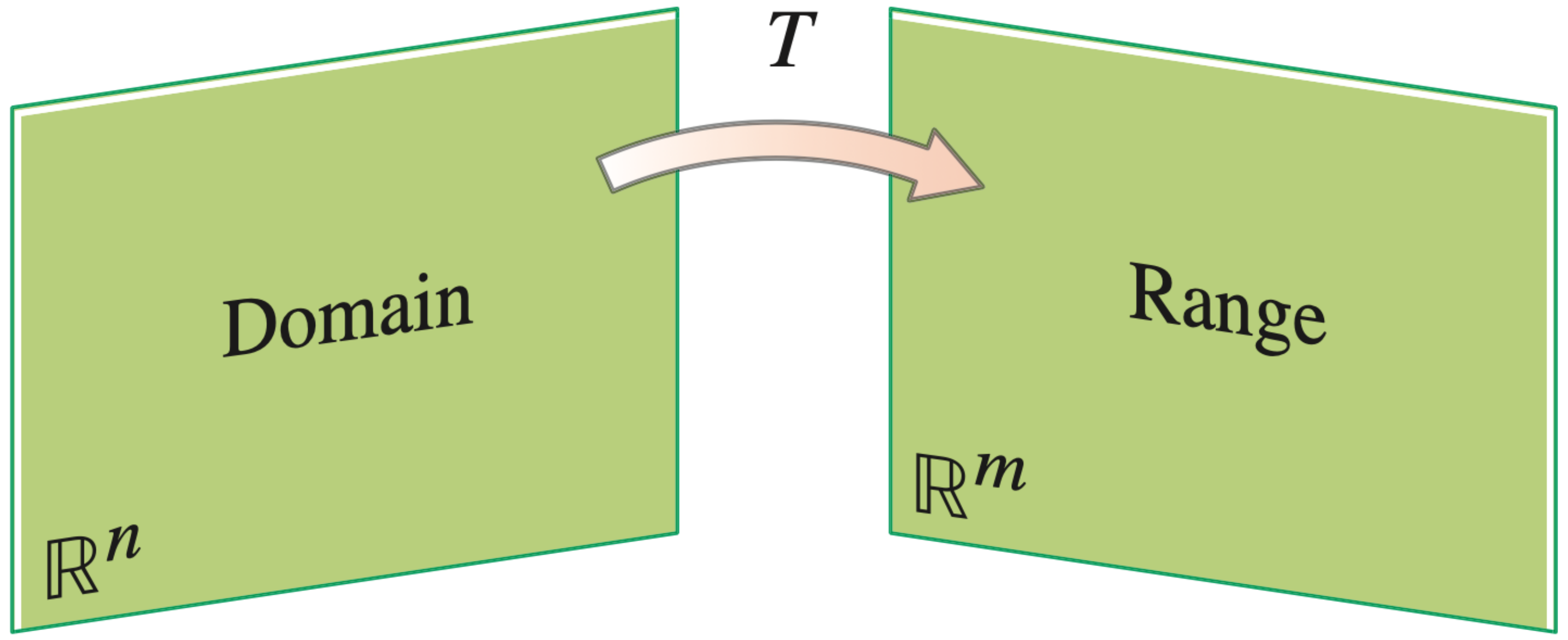
Definition. A transformation $T: \mathbb{R}^n \to \mathbb{R}^m$ is ***onto*** if any vector $\mathbf{b}$ in $\mathbb{R}^m$ is the image of at least one vector $\mathbf{v}$ in $\mathbb{R}^n$ (where $T(\mathbf{v}) = \mathbf{b}$).

Onto Transformations

Definition. A transformation $T: \mathbb{R}^n \to \mathbb{R}^m$ is ***onto*** if any vector $\mathbf{b}$ in $\mathbb{R}^m$ is the image of at least one vector $\mathbf{v}$ in $\mathbb{R}^n$ (where $T(\mathbf{v}) = \mathbf{b}$).



T is *not* onto $\mathbb{R}^m$

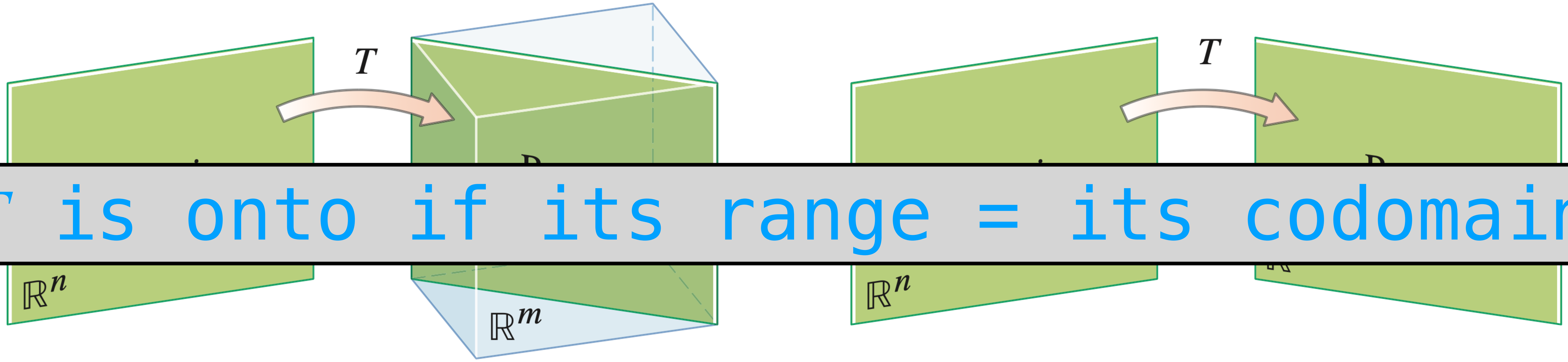


T is onto $\mathbb{R}^m$

image source: Linear Algebra and its Applications. Lay, Lay, and McDonald

Onto Transformations

Definition. A transformation $T : \mathbb{R}^n \to \mathbb{R}^m$ is ***onto*** if any vector $\mathbf{b}$ in $\mathbb{R}^m$ is the image of at least one vector $\mathbf{v}$ in $\mathbb{R}^n$ (where $T(\mathbf{v}) = \mathbf{b}$).



T is onto if its range = its codomain

T is *not* onto $\mathbb{R}^m$ $\qquad$ T is onto $\mathbb{R}^m$

image source: Linear Algebra and its Applications. Lay, Lay, and McDonald

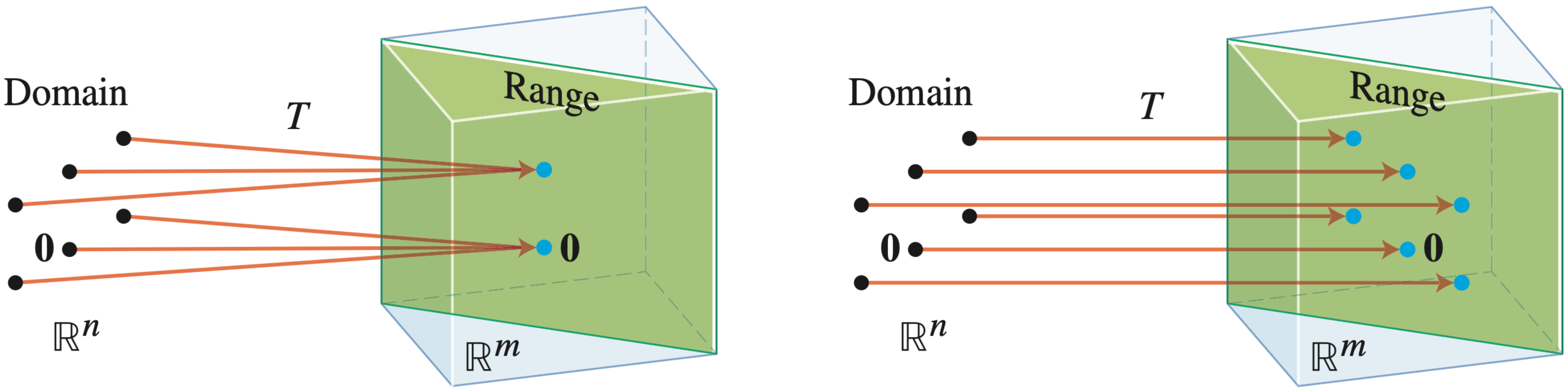
One-to-one Transformations

One-to-one Transformations

Definition. A transformation $T: \mathbb{R}^n \to \mathbb{R}^m$ is ***one-to-one*** if any vector $\mathbf{b}$ in $\mathbb{R}^m$ is the image of at most one vector $\mathbf{v}$ in $\mathbb{R}^n$ (where $T(\mathbf{v}) = \mathbf{b}$).

One-to-one Transformations

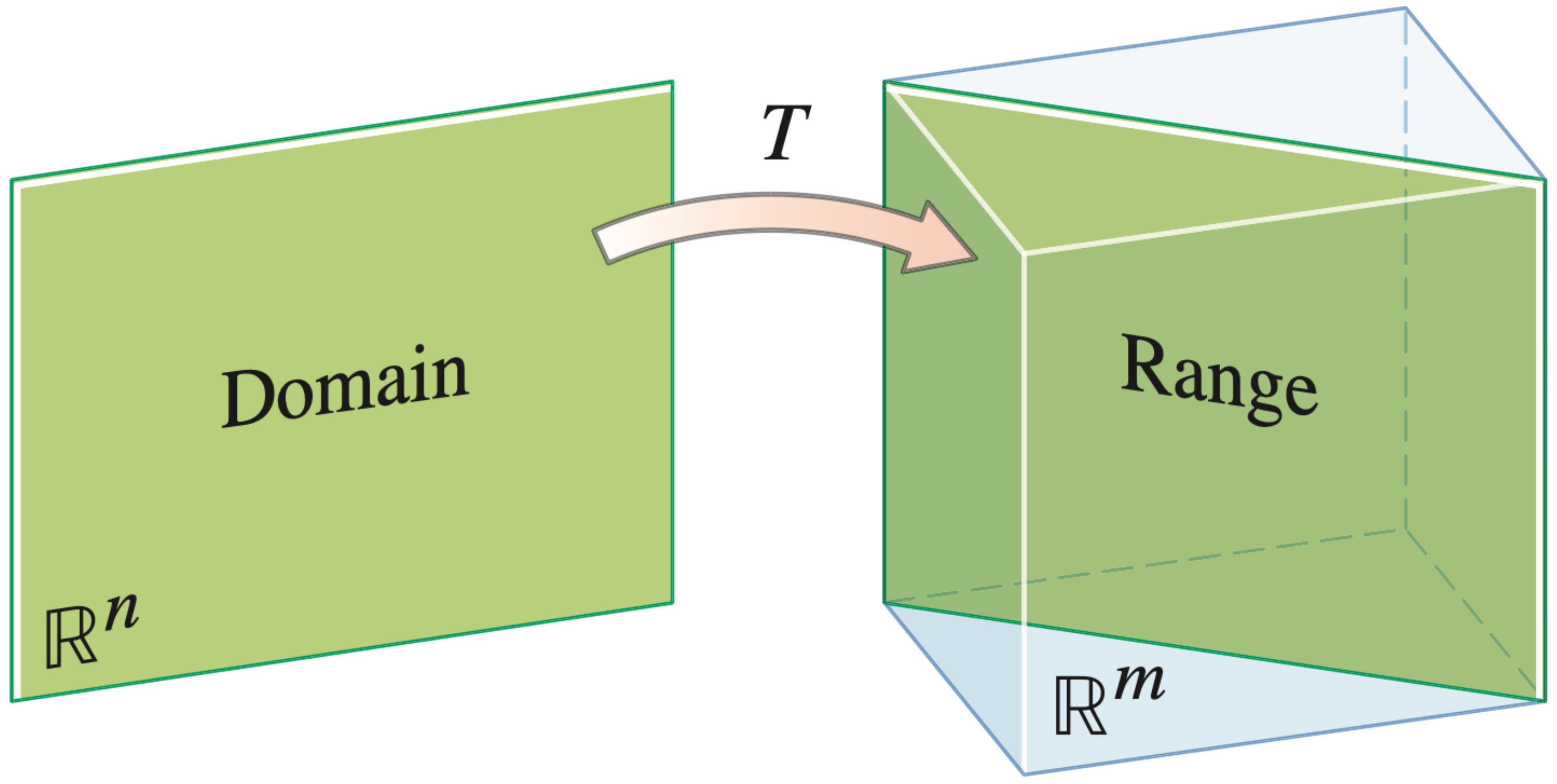
Definition. A transformation $T:\mathbb{R}^n \to \mathbb{R}^m$ is ***one-to-one*** if any vector $\mathbf{b}$ in $\mathbb{R}^m$ is the image of at most one vector $\mathbf{v}$ in $\mathbb{R}^n$ (where $T(\mathbf{v}) = \mathbf{b}$).



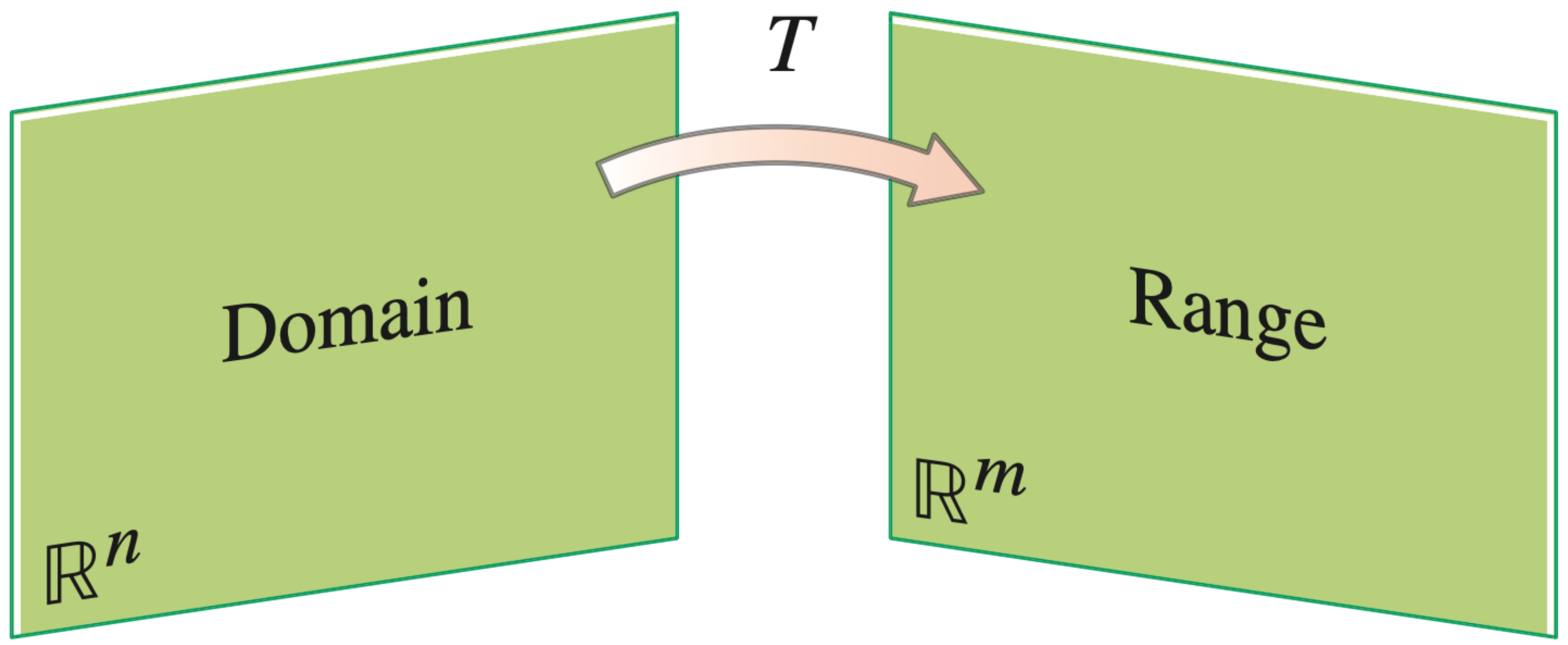
T is *not* one-to-one   *T* is one-to-one

image source: Linear Algebra and its Applications. Lay, Lay, and McDonald

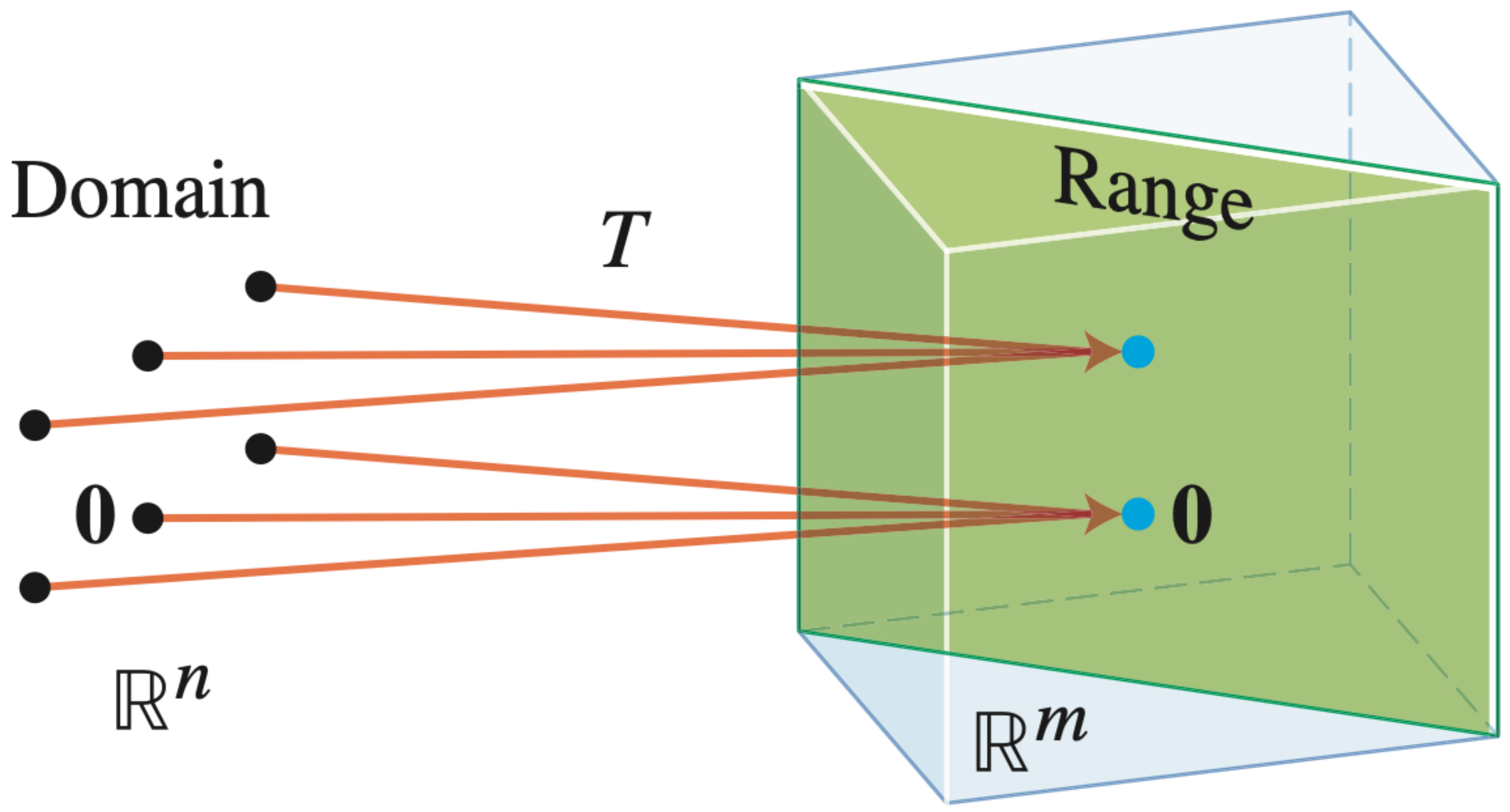
Comparing Pictures



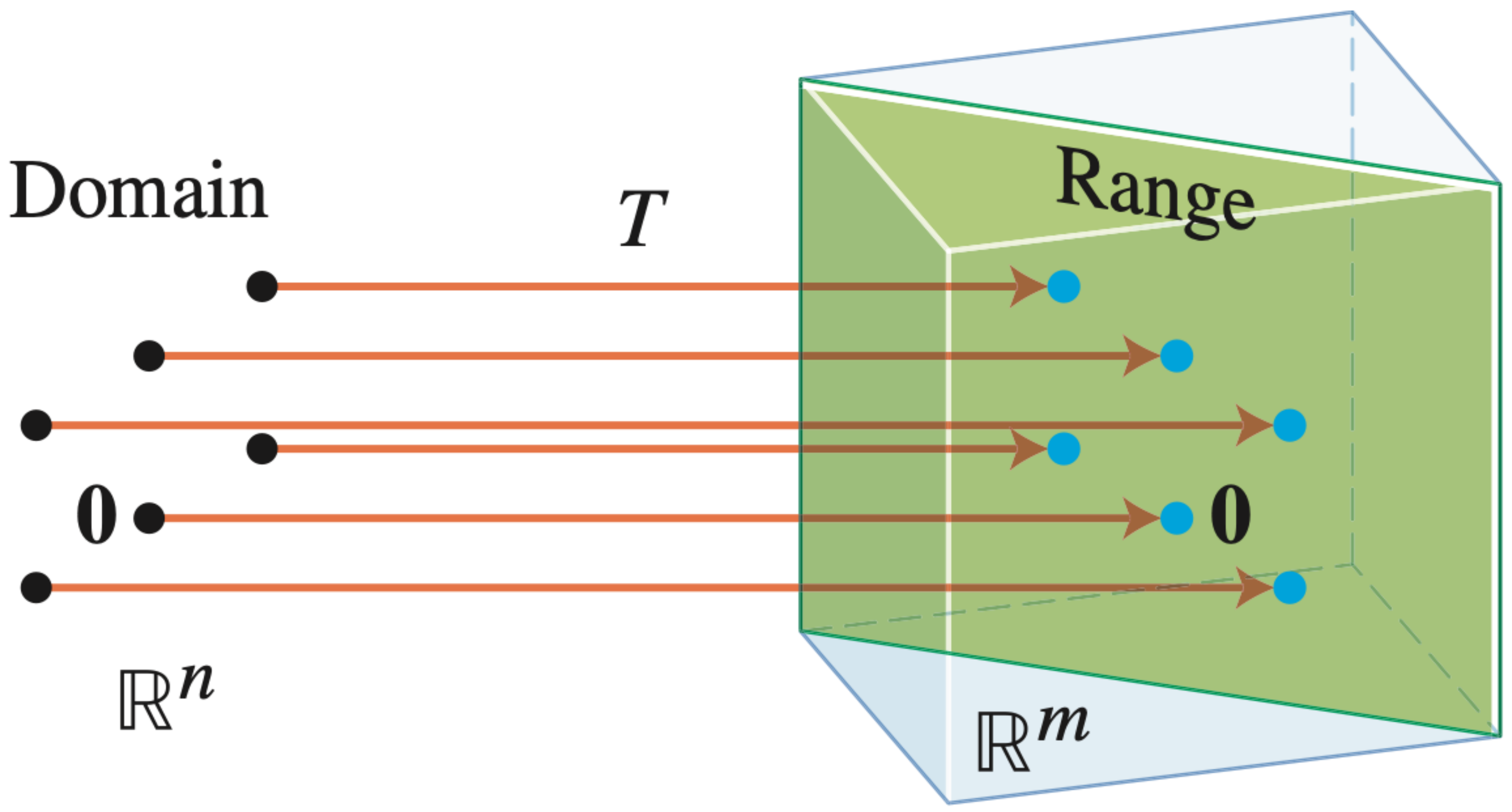
T is *not* onto $\mathbb{R}^m$



T is onto $\mathbb{R}^m$



T is *not* one-to-one



T is one-to-one

Example: both 1-1 and onto

Rotation about the origin:

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

why?:

Example: 1-1, not onto

Lifting:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \mapsto \begin{bmatrix} x_1 \\ x_2 \\ x_1 + x_2 \end{bmatrix}$$

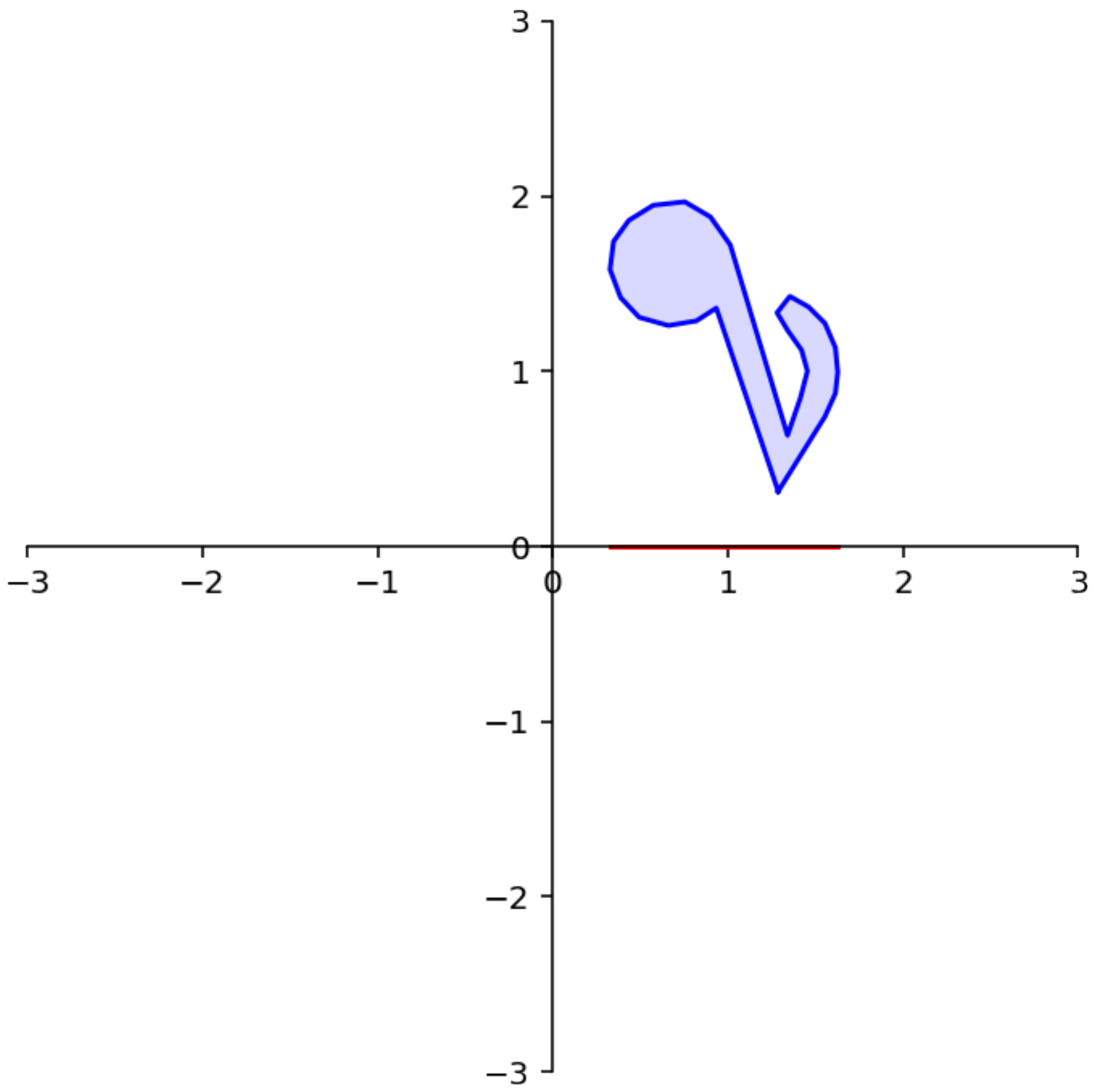
why?:

Example: not 1-1, not onto

Projection onto the x_1 axis:

$$\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

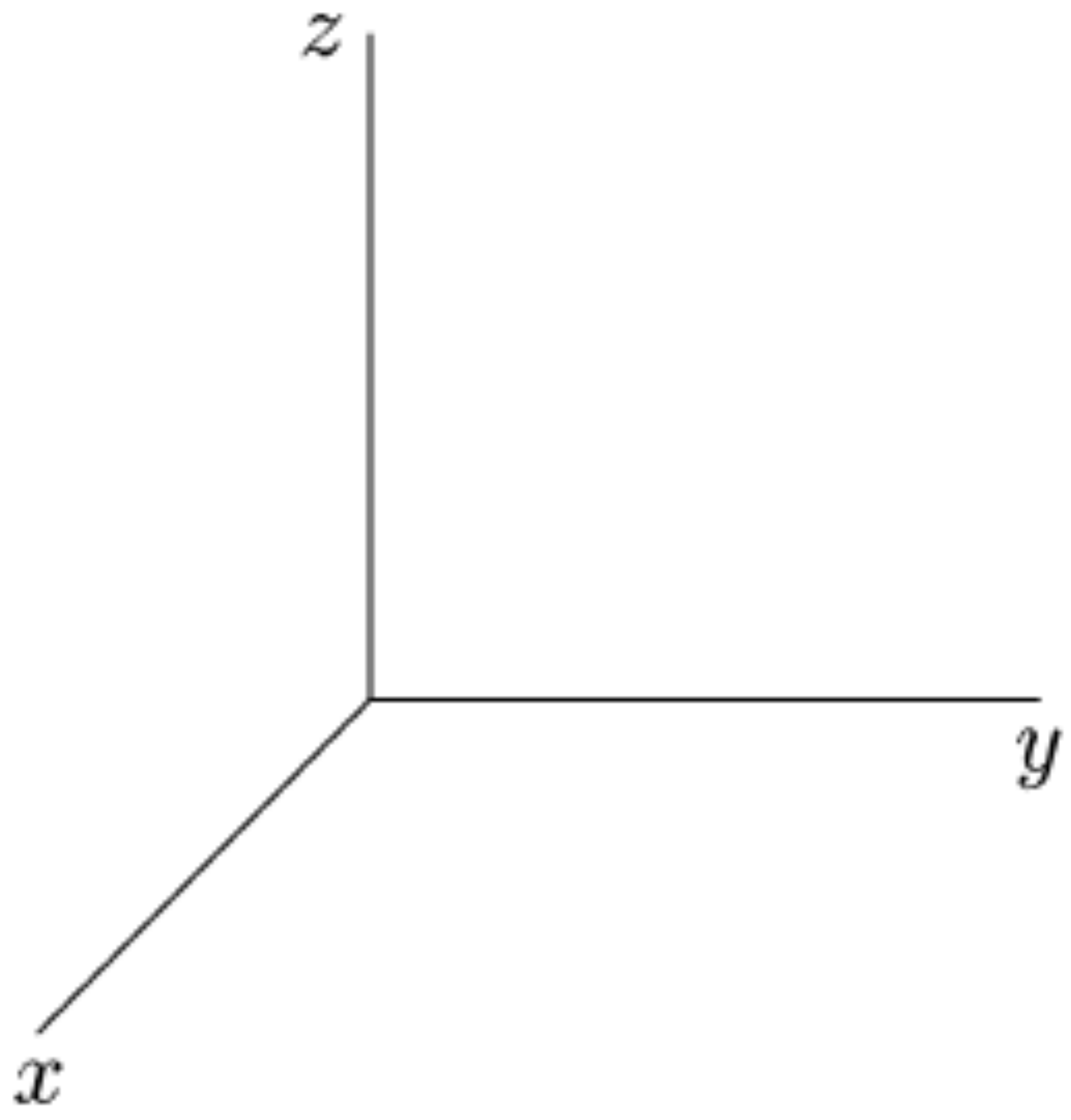
why?:



Example: onto, not 1-1

Projection from $\mathbb{R}^3$ to $\mathbb{R}^2$.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \mapsto \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$



why?:

Taking Stock: Onto

Theorem. The following are logically equivalent for the linear transformation $T : \mathbb{R}^n \to \mathbb{R}^m$ implemented by the matrix A.

- » T is onto
- » $A\mathbf{x} = \mathbf{b}$ has a solution for any choice of $\mathbf{b}$
- » $\text{range}(T) = \text{codomain}(T)$
- » the columns of A span $\mathbb{R}^m$
- » A has a pivot position in every row

Taking Stock: One-to-One

Theorem. The following are logically equivalent for the linear transformation $T : \mathbb{R}^n \to \mathbb{R}^m$ implemented by the matrix A.

- » T is one-to-one
- » $A\mathbf{x} = \mathbf{b}$ has at most one solution for any $\mathbf{b}$
- » $A\mathbf{x} = \mathbf{0}$ has only the trivial solution
- » The columns of A are linearly independent
- » A has a pivot position in every column

How To: One-to-One and Onto

Question. Show that the linear transformation T is one-to-one/onto.

Solution. (one approach) Find the matrix which implements T and see if it has a pivot in every column/row.

Warning: this is not the only way. Always try to think if you can solve it using *any* of the perspectives

Example: both 1-1 and onto

Rotation about the origin:

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

why?:

Example: 1-1, not onto

`Lifting:`

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \mapsto \begin{bmatrix} x_1 \\ x_2 \\ x_1 + x_2 \end{bmatrix}$$

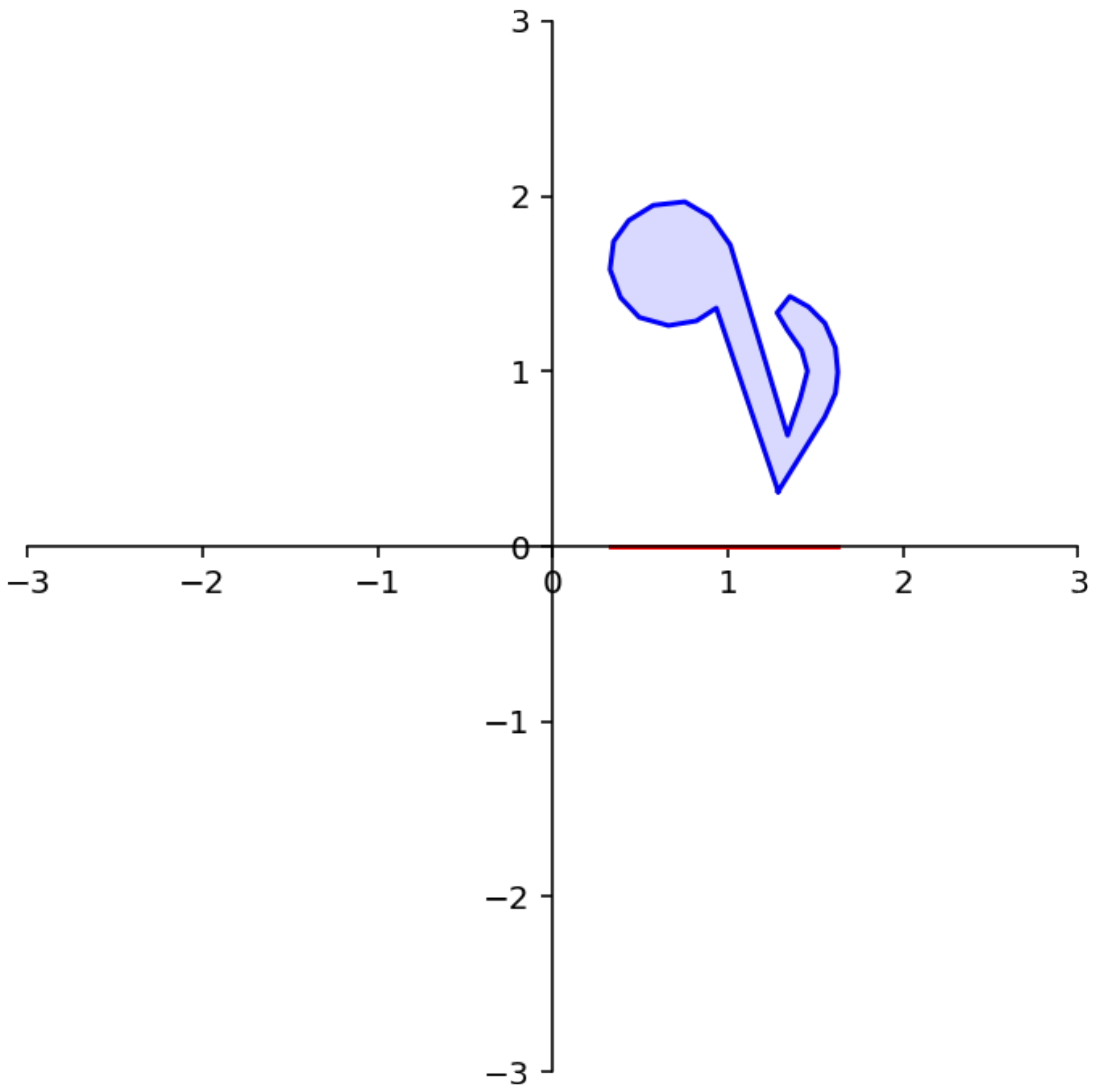
`why?:`

Example: not 1-1, not onto

Projection onto the x_1 axis:

$$\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

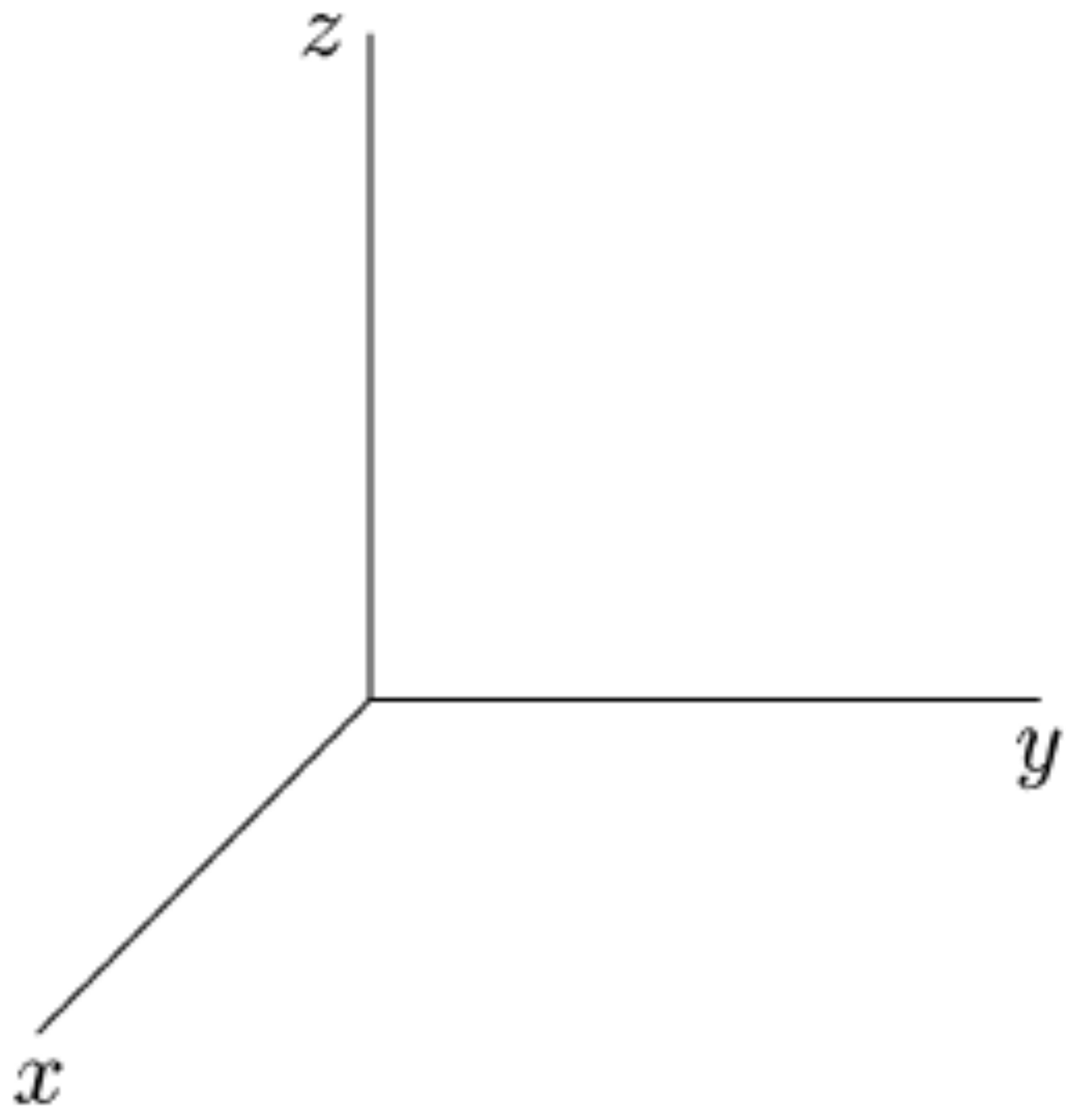
why?:



Example: onto, not 1-1

Projection from $\mathbb{R}^3$ to $\mathbb{R}^2$.

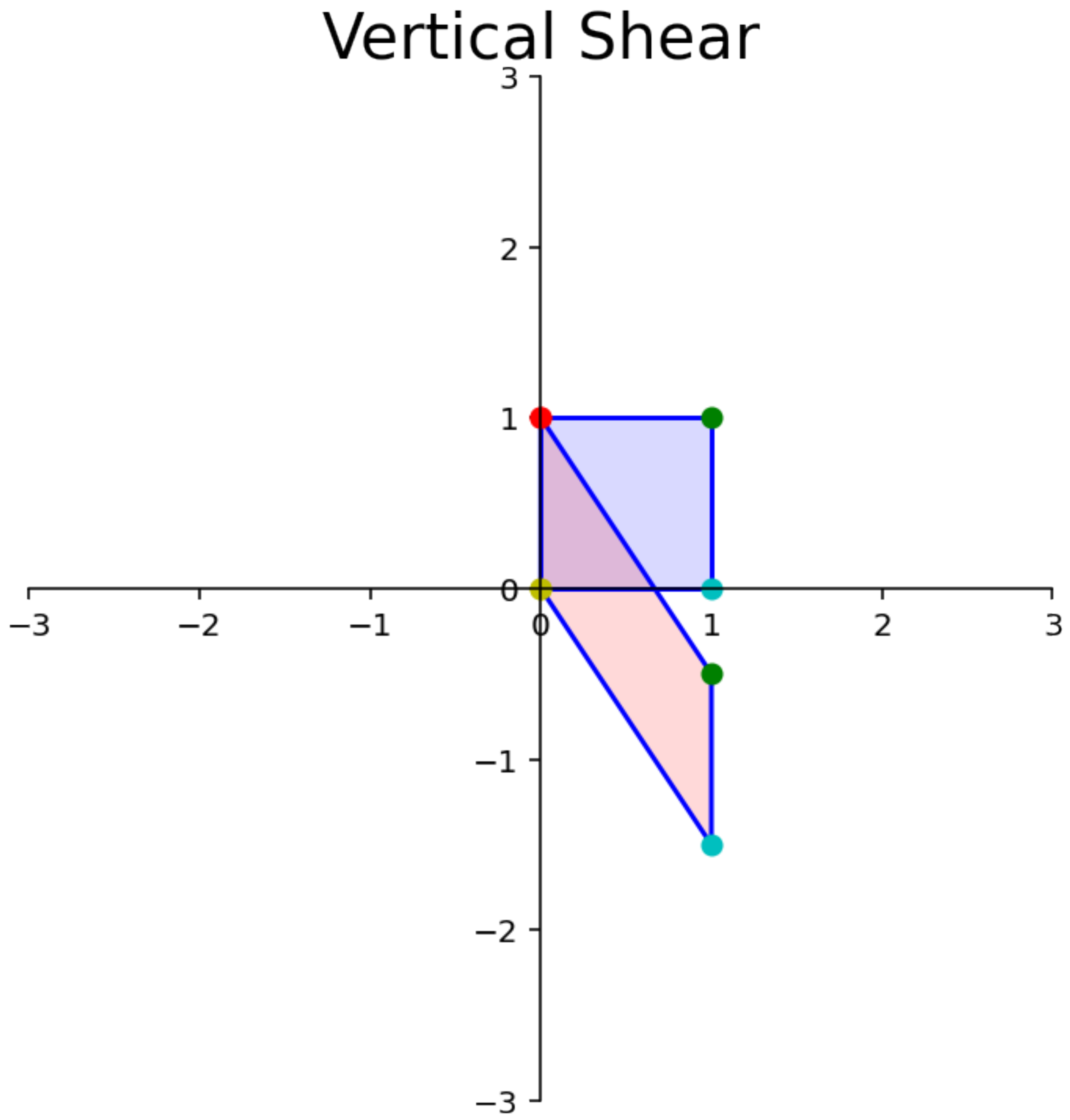
$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \mapsto \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$



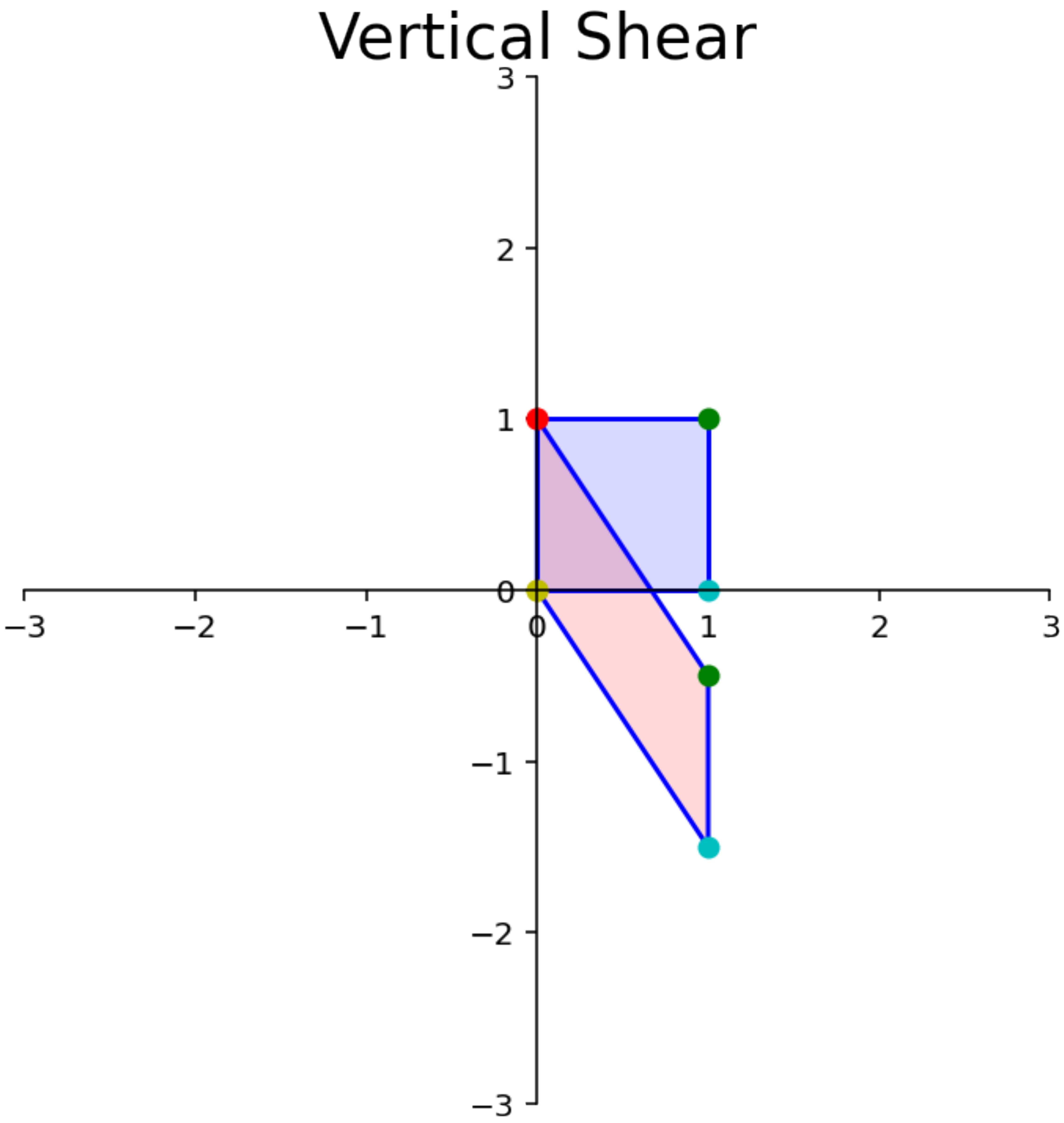
why?:

Question

Is vertical shearing a 1–1 transformation? Justify your answer.

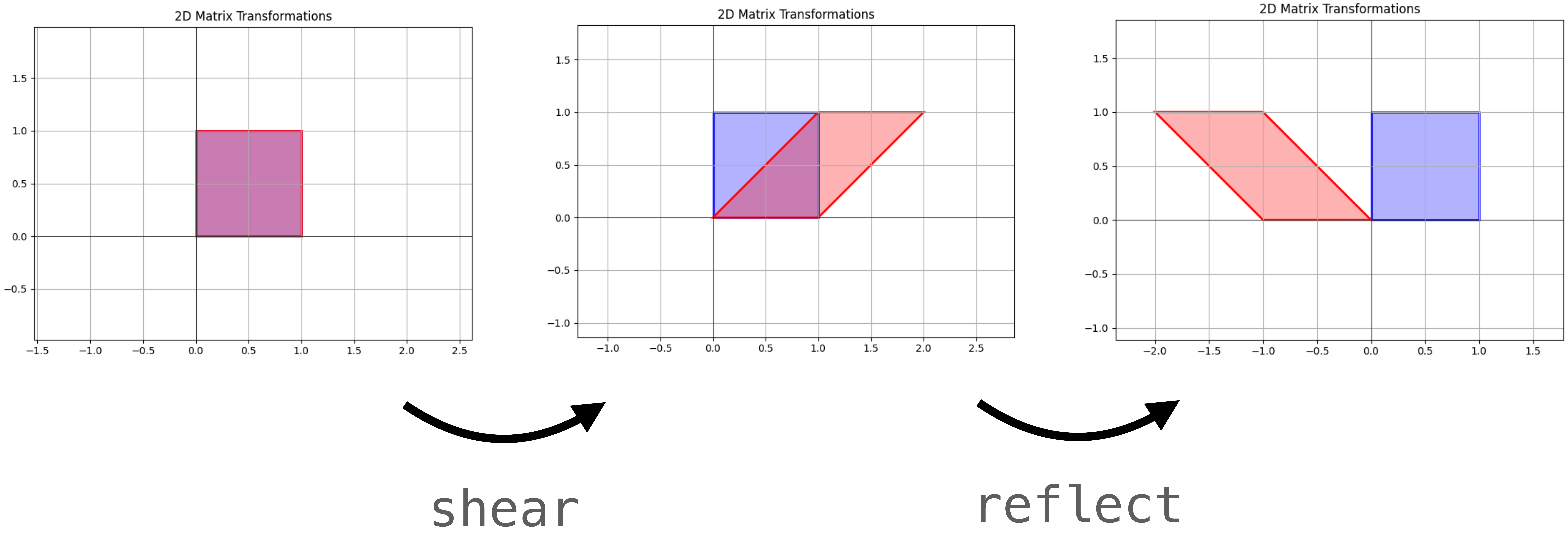


Answer: Yes



Composing Linear Transformations

Shearing and Reflecting (Geometrically)

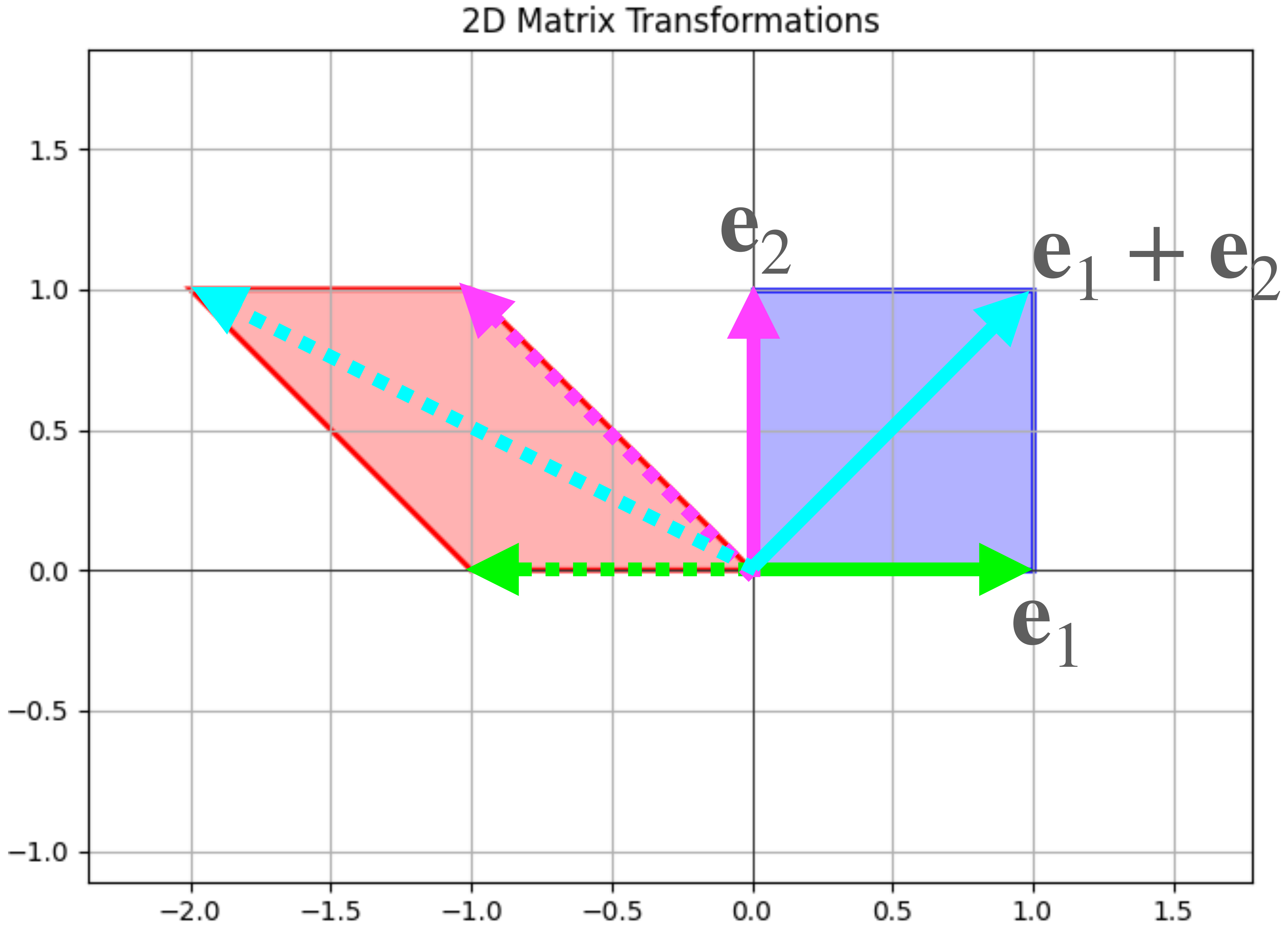


Shearing and Reflecting Matrix

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} \mapsto$$

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix} \mapsto$$

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \mapsto$$



Shearing and Reflecting (Algebraically)

$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \left(\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right)$$

reflect shear

First multiply by shear matrix, then multiply by reflection matrix

Shearing and Reflecting (Algebraically)

$$\underbrace{\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}}_{\text{reflect}} \left(\underbrace{\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}}_{\text{shear}} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right)$$

First multiply by shear matrix, then multiply by reflection matrix

This gives us the same transformation.

Shearing and Reflecting

$$\begin{bmatrix} -1 & -1 \\ 0 & 1 \end{bmatrix} \mathbf{x} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \left(\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \mathbf{x} \right)$$

The Key Fact

The Key Fact

Fact. The composition of two linear transformation is a linear transformation.

The Key Fact

Fact. The composition of two linear transformation is a linear transformation.

Verify:

The Key Fact

Fact. The composition of two linear transformation is a linear transformation.

Verify:

This means the composition of two matrix transformation can be represented as a *single* matrix.

The Key Question

Given two linear transformations, how to we compute the matrix which implements their composition?

The Key Question

Given two linear transformations, how to we compute the matrix which implements their composition?

Matrix Multiplication

Matrix Multiplication

Shearing and Reflecting

$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \left(\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right) =$$

General Composition (2D)

$$A\left(\begin{bmatrix}\mathbf{b}_1 & \mathbf{b}_2\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}\right) =$$

Matrix Multiplication

Definition. For a $m \times n$ matrix A and a $n \times p$ matrix B with columns $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_p$ the product AB is the $m \times p$ matrix given by

$$AB = A\begin{bmatrix}\mathbf{b}_1 & \mathbf{b}_2 & \ldots & \mathbf{b}_p\end{bmatrix} = \begin{bmatrix}A\mathbf{b}_1 & A\mathbf{b}_2 & \ldots & A\mathbf{b}_p\end{bmatrix}$$

Replace each column of B with A multiplied by that column.

Tracking Dimensions

this only works if the number of columns of the left matrix matches the number of rows of the right matrix

$$m\overset{n}{\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}} \; n\overset{k}{\begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}} = m\overset{k}{\begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}}$$

$(m \times n)$ $(n \times k)$ $(m \times k)$

Important Note

Even if AB is defined, it may be that BA is <u>not</u> defined

Non-Example

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \end{bmatrix} \begin{bmatrix} 2 \\ 4 \end{bmatrix} = \left[\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \end{bmatrix} \quad \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \begin{bmatrix} 2 \\ 4 \end{bmatrix} \right]$$

Non-Example

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \end{bmatrix} & \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \begin{bmatrix} 2 \\ 4 \end{bmatrix} \end{bmatrix}$$

These are not defined.

Example

$$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\end{bmatrix} = \left[\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}\begin{bmatrix}1\\4\end{bmatrix} \quad \begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}\begin{bmatrix}2\\5\end{bmatrix} \quad \begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}\begin{bmatrix}3\\6\end{bmatrix}\right]$$

The Key Fact (Restated)

For any matrices A and B (such that AB is defined) and any vector $\mathbf{v}$

$$A(B\mathbf{v}) = (AB)\mathbf{v}$$

The matrix implementing the composition is the product of the two underlying matrices.

Row-Column Rule

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik}B_{kj}$$

Given a $m \times n$ matrix A and a $n \times p$ matrix B, the entry in row i and column j of AB is defined above.

Example

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

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik}B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik}B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik}B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

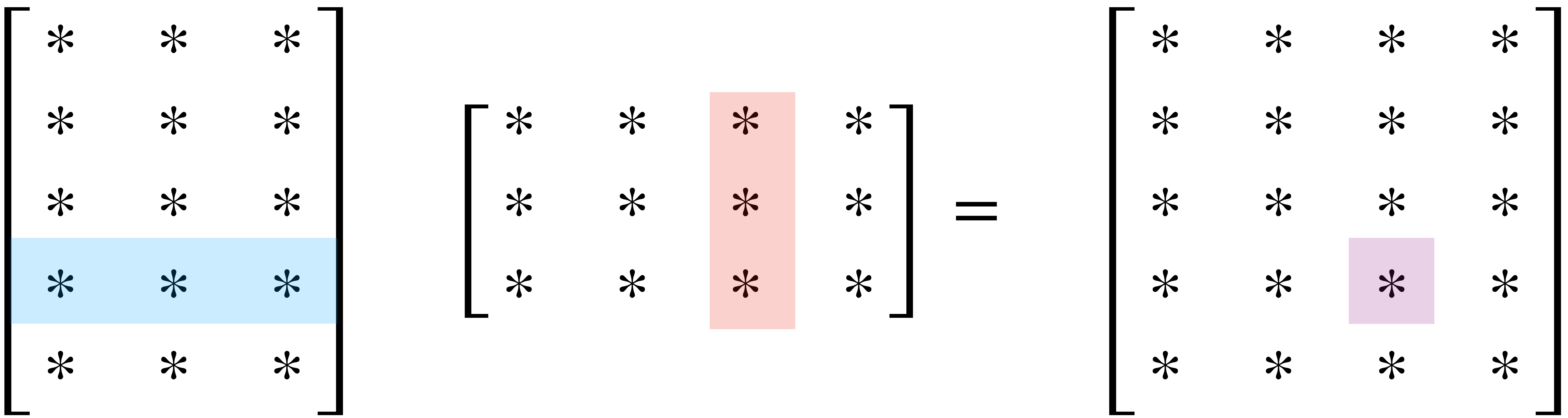
$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Row-Column Rule (Pictorially)



$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik}B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik}B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik}B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik}B_{kj}$$

Row-Column Rule (Pictorially)

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$$

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Question

Compute $\begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 2 & 0 \\ -1 & 2 \end{bmatrix}$

short version: What is the entry in the 2nd row and 2nd column?

Answer

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

Matrix Operations

Connection with Matrix-Vector Multiplication

Connection with Matrix-Vector Multiplication

`What about when the right matrix is a single column?`

Connection with Matrix-Vector Multiplication

`What about when the right matrix is a single column?`

$$A[\mathbf{b}_1] = [A\mathbf{b}_1] = A\mathbf{b}_1$$

Connection with Matrix-Vector Multiplication

What about when the right matrix is a single column?

$$A[\mathbf{b}_1] = [A\mathbf{b}_1] = A\mathbf{b}_1$$

This is just vector multiplication.

Connection with Matrix-Vector Multiplication

What about when the right matrix is a single column?

$$A[\mathbf{b}_1] = [A\mathbf{b}_1] = A\mathbf{b}_1$$

This is just vector multiplication.

We can think of $[A\mathbf{b}_1 \;\; A\mathbf{b}_2 \;\; \ldots \;\; A\mathbf{b}_p]$ as collection of simultaneous matrix–vector multiplications

Matrix "Interface"

multiplication — what does AB mean when A and B are matrices?

addition — what does $A + B$ mean when A and B are matrices?

scaling — what does cA mean when A is matrix and c is a real number?

Matrix "Interface"

multiplication — what does AB mean when A and B are matrices?

addition — what does $A + B$ mean when A and B are matrices?

scaling — what does cA mean when A is matrix and c is a real number?

These should be consistent with matrix-vector interface and vector interface

Matrix Addition

$$[\mathbf{a}_1 \quad \dots \quad \mathbf{a}_n] + \left[\mathbf{b}_1 \dots \quad \mathbf{b}_n\right] = \left[(\mathbf{a}_1 + \mathbf{b}_1) \quad \dots \quad (\mathbf{a}_n + \mathbf{b}_n)\right]$$

Addition is done column-wise (or equivalently, element-wise)

e.g. $$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 2 & 3 \\ -2 & -3 \end{bmatrix} = \begin{bmatrix} (1+2) & (2+3) \\ (3-2) & (4-3) \end{bmatrix} = \begin{bmatrix} 3 & 5 \\ 1 & 1 \end{bmatrix}$$

Matrix Addition

$$[\mathbf{a}_1 \quad \dots \quad \mathbf{a}_n] + [\mathbf{b}_1 \dots \quad \mathbf{b}_n] = [(\mathbf{a}_1 + \mathbf{b}_1) \quad \dots \quad (\mathbf{a}_n + \mathbf{b}_n)]$$

Addition is done column-wise (or equivalently, element-wise)

e.g. $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 2 & 3 \\ -2 & -3 \end{bmatrix} = \begin{bmatrix} (1+2) & (2+3) \\ (3-2) & (4-3) \end{bmatrix} = \begin{bmatrix} 3 & 5 \\ 1 & 1 \end{bmatrix}$

This is exactly the same as vector addition, but for matrices.

Matrix Addition and Scaling

$$c\,[\mathbf{a}_1 \quad \mathbf{a}_2 \quad \cdots \quad \mathbf{a}_n] = [c\mathbf{a}_1 \quad c\mathbf{a}_2 \quad \cdots \quad c\mathbf{a}_n]$$

Scaling and adding happen element-wise (or, equivalently, column-wise).

e.g. $2\begin{bmatrix} 1 & 2 \\ -1 & 3 \end{bmatrix} = \begin{bmatrix} 2(1) & 2(2) \\ 2(-1) & 2(3) \end{bmatrix} = \begin{bmatrix} 2 & 4 \\ -2 & 6 \end{bmatrix}$

Matrix Addition and Scaling

$$c\,[\mathbf{a}_1 \quad \mathbf{a}_2 \quad \cdots \quad \mathbf{a}_n] = [c\mathbf{a}_1 \quad c\mathbf{a}_2 \quad \cdots \quad c\mathbf{a}_n]$$

Scaling and adding happen element-wise (or, equivalently, column-wise).

e.g. $2\begin{bmatrix} 1 & 2 \\ -1 & 3 \end{bmatrix} = \begin{bmatrix} 2(1) & 2(2) \\ 2(-1) & 2(3) \end{bmatrix} = \begin{bmatrix} 2 & 4 \\ -2 & 6 \end{bmatrix}$

This is exactly the same as vector scaling, but for matrices.

Algebraic Properties (Addition and Scaling)

In these properties A, B, and C are matrices of the same size and r and s are scalars ($\mathbb{R}$)

$$A + B = B + A$$

$$(A + B) + C = A + (B + C)$$

$$A + 0 = A$$

$$r(A + B) = rA + rB$$

$$(r + s)A = rA + sA$$

$$r(sA) = (rs)A$$

We need to know/memorize these.

Algebraic Properties (Addition and Scaling)

In these properties A, B, and C are matrices of the appropriate size so that everything is defined, and r is a scalar

$$A(BC) = (AB)C$$

$$A(B + C) = AB + AC$$

$$(B + C)A = BC + CA$$

$$r(AB) = (rA)B = A(rB)$$

$$I_m A = A = AI_n$$

We need to know/memorize these.

Matrix Multiplication is not Commutative

Important. AB may not be the same as BA

(it may not even be defined)

Question (Conceptual)

Find a pair of 2D linear transformations T_1 and T_2 such that T_1 followed by T_2 is not the same as T_2 followed by T_1.

(also find a pair where they are the same)

Answer: Rotation and Reflection

Computational Aspects of Matrix Multiplication

Matrix Operations in Numpy

Let a and b be 2D numpy arrays and let c be a floating point number.

- » `a @ b` (matrix multiplication)
- » `a + b` (matrix addition)
- » `c * a` (matrix scaling)

We've seen these, we've used them a bit, we'll use them much more.

Analyzing Linear Algebra Algorithms

Analyzing Linear Algebra Algorithms

`We will not use` $O(\cdot)$ `notation!`

Analyzing Linear Algebra Algorithms

We will not use $O(\cdot)$ notation!

For numerics, we care about number of **FL**oating-oint **OP**erations (FLOPs):

>> addition
>> subtraction
>> multiplication
>> division
>> square root

Analyzing Linear Algebra Algorithms

We will not use $O(\cdot)$ notation!

For numerics, we care about number of **FL**oating-oint **OP**erations (FLOPs):

>> addition
>> subtraction
>> multiplication
>> division
>> square root

$2n$ vs. n is very different when $n \sim 10^{20}$

Dominant Terms

Dominant Terms

`that said, we don't care about` *`exact`* `bounds`

Dominant Terms

that said, we don't care about *exact* bounds

A function $f(n)$ is ***asymptotically equivalent*** to $g(n)$ if

$$\lim_{i \to \infty} \frac{f(i)}{g(i)} = 1$$

Dominant Terms

that said, we don't care about *exact* bounds

A function $f(n)$ is ***asymptotically equivalent*** to $g(n)$ if

$$\lim_{i \to \infty} \frac{f(i)}{g(i)} = 1$$

for polynomials, they are equivalent to their dominant term

Dominant Terms

the dominant term of a polynomial is the monomial with the highest degree

$$\lim_{i \to \infty} \frac{3x^3 + 100000x^2}{3x^3} = 1$$

$3x^3$ dominates the function even though the coefficient for x^2 is so large

A Note on Complexity

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Suppose A and B are $n \times n$ matrices.

This operations takes n multiplications and n divisions ($2n$ FLOPS total)

Repeating for each entry gives $\sim 2n^3$ FLOPS

A Note on Parallelization

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik}B_{kj}$$

```
The main part of this procedure is highly
parallelizable.
```

A Note on Parallelization

```
a = np.array(...)
b = np.array(...)
prod = np.zeros([a.shape[0], b.shape[1]])
for i in range(a.shape[0]):
    for j in range(b.shape[1]):
        prod[i, j] = np.dot(a[i], b[:,j])
```

The main part of this procedure is highly parallelizable.

One processor per entry gets you to $\sim 2n$ FLOPS

A Note on Libraries

There are a lot of other considerations for doing linear algebra on computers.

Best leave it to experts (or do research in the area).

LAPACK is the state of the art library for matrix operations.

numpy uses LAPACK

Summary

We can reason about matrix equations by reasoning directly about properties of linear transformations.

Matrix multiplication coincides with composition of linear transformations.

There is an algebra of matrices which is consistent with the algebra of vectors.