# Linear Algebra Primer

Mahtab Bigverdi

# Outline

- Vectors and matrices
  - Basic Matrix Operations
  - Determinants, norms, trace
  - Special Matrices
- Transformation Matrices
  - Homogeneous coordinates
  - Translation
- Matrix inverse
- Matrix rank
- Eigenvalues and Eigenvectors
- Matrix Calculus

# Outline

- Vectors and matrices
  - Basic Matrix Operations
  - Determinants, norms, trace
  - Special Matrices
- Transformation Matrices
  - Homogeneous coordinates
  - Translation
- Matrix inverse
- Matrix rank
- Eigenvalues and Eigenvectors
- Matrix Calculus

Vectors and matrices are just collections of ordered numbers that represent something: movements in space, scaling factors, pixel brightness, etc. We'll define some common uses and standard operations on them.

# Vector

- A column vector $\mathbf{v} \in \mathbb{R}^{n \times 1}$ where

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

- A row vector $\mathbf{v}^T \in \mathbb{R}^{1 \times n}$ where

$$\mathbf{v}^T = \begin{bmatrix} v_1 & v_2 & \dots & v_n \end{bmatrix}$$

$T$ denotes the transpose operation

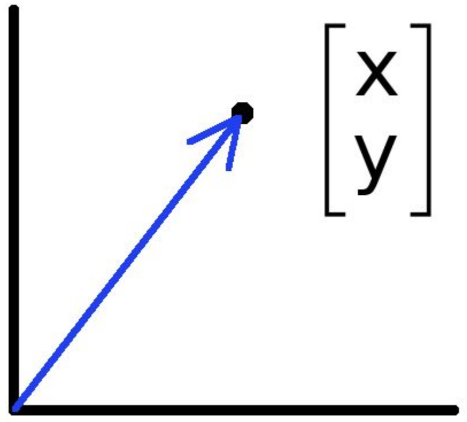
# Vector

- We'll default to column vectors in this class

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

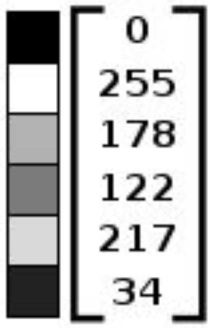
- You'll want to keep track of the orientation of your vectors when programming in python
- You can transpose a vector *V* in python by writing **V.t**. (But in class materials, we will **always** use $V^T$ to indicate transpose, and we will use V' to mean "V prime")

# Vectors have two main uses




- Vectors can represent an offset in 2D or 3D space.
- Points are just vectors from the origin.

- Data (pixels, gradients at an image keypoint, etc) can also be treated as a vector.




- Such vectors don’t have a geometric interpretation, but calculations like “distance” can still have value.

# Matrix

- A matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ is an array of numbers with size by , i.e. m rows and n columns.

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ \vdots & & & & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{bmatrix}$$

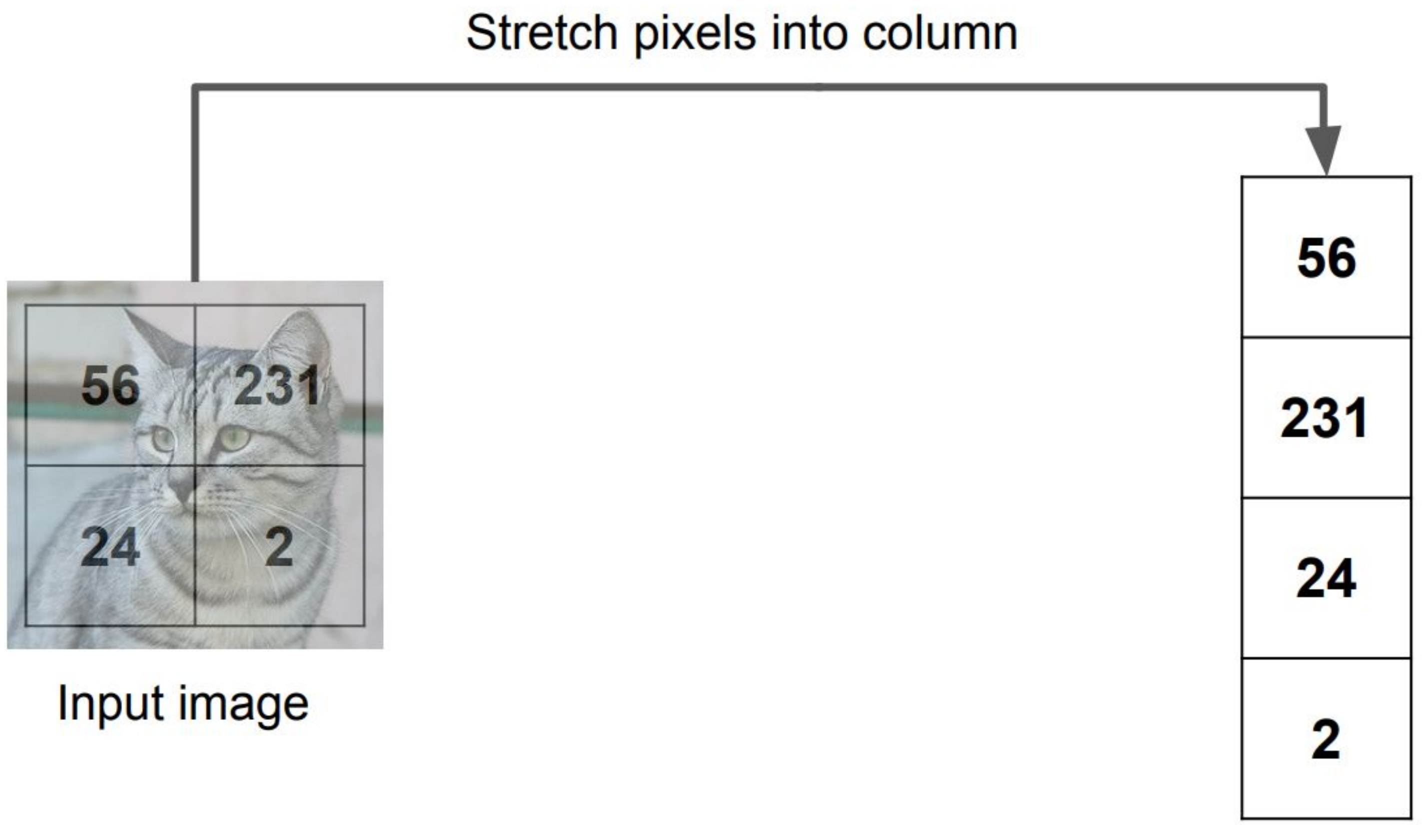
- If $m = n$, we say that $\mathbf{A}$ is square.

# Images

$$= \begin{bmatrix} 193 & 180 & 210 & 112 & 125 \\ 189 & 8 & 177 & 97 & 114 \\ 100 & 71 & 81 & 195 & 165 \\ 167 & 12 & 242 & 203 & 181 \\ 44 & 25 & 9 & 48 & 192 \end{bmatrix}$$

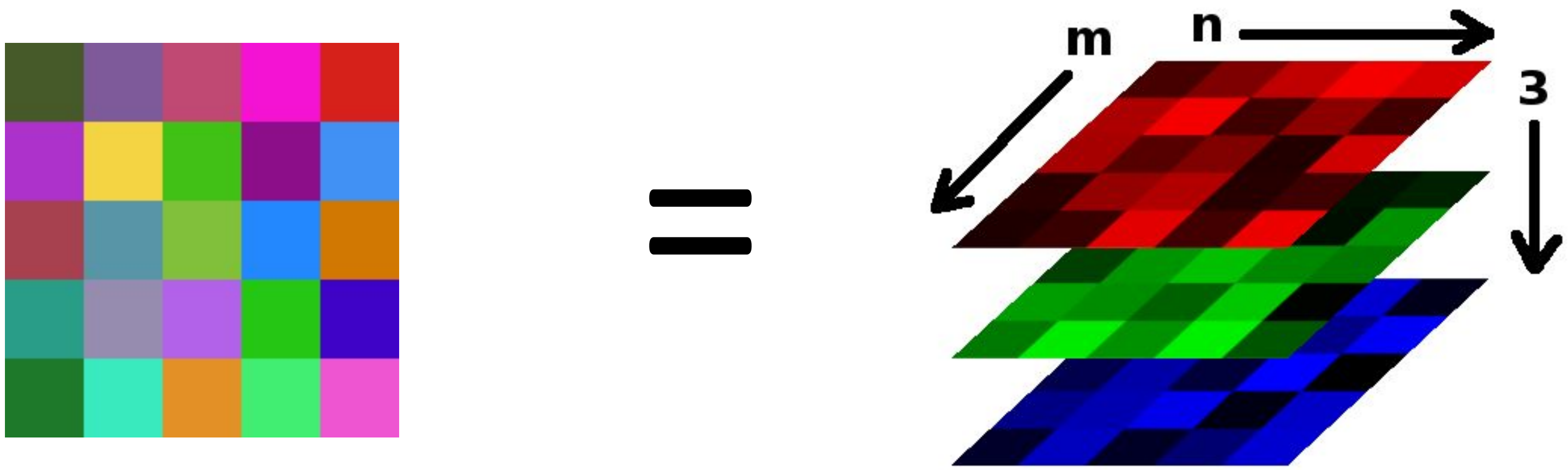
- Python represents an image as a matrix of pixel brightnesses
- Note that the upper left corner is [y,x] = (0,0)

# Images as both a **matrix** as well as a **vector**



# Color Images

- Grayscale images have one number per pixel, and are stored as an m × n matrix.
- Color images have 3 numbers per pixel – red, green, and blue brightnesses (RGB)
- Stored as an m × n × 3 matrix



# Basic Matrix Operations

- We will discuss:
  - Addition
  - Scaling
  - Dot product
  - Multiplication
  - Transpose
  - Inverse / pseudoinverse
  - Determinant / trace

# Matrix Operations

- Addition

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} + \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} a+1 & b+2 \\ c+3 & d+4 \end{bmatrix}$$

  – Can only add a matrix with matching dimensions, or a scalar.

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} + 7 = \begin{bmatrix} a+7 & b+7 \\ c+7 & d+7 \end{bmatrix}$$

- Scaling

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \times 3 = \begin{bmatrix} 3a & 3b \\ 3c & 3d \end{bmatrix}$$

# Vectors

- **Norm** $\|x\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}.$

- More formally, a norm is any function $f : \mathbb{R}^n \to \mathbb{R}$ that satisfies 4 properties:

- **Non-negativity:** For all $x \in \mathbb{R}^n,\ f(x) \geq 0$
- **Definiteness**: f(x) = 0 if and only if x = 0.
- **Homogeneity**: For all $x \in \mathbb{R}^n,\ t \in \mathbb{R},\ f(tx) = |t| f(x)$
- **Triangle inequality**: For all $x, y \in \mathbb{R}^n,\ f(x+y) \leq f(x) + f(y)$

# Norms

- **Example Norms**

$$\|x\|_1 = \sum_{i=1}^{n} |x_i| \qquad\qquad \|x\|_\infty = \max_i |x_i|.$$

- General $\ell_p$ norms:

$$\|x\|_p = \left( \sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

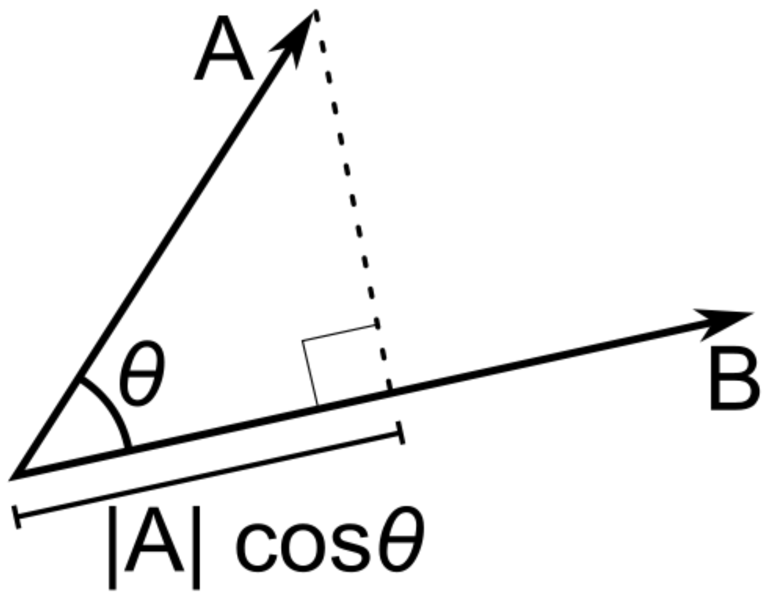
# Matrix Operations

- Inner product (dot product) of vectors
  - Multiply corresponding entries of two vectors and add up the result
  - x·y is also |x||y|Cos( *the angle between x and y* )

$$\mathbf{x}^T\mathbf{y} = \begin{bmatrix} x_1 & \dots & x_n \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \sum_{i=1}^{n} x_i y_i \qquad \text{(scalar)}$$

# Matrix Operations

- Inner product (dot product) of vectors
  - If B is a unit vector, then A·B gives the length of A which lies in the direction of B



# Matrix Operations

- The product of two matrices

$$A \in \mathbb{R}^{m \times n}$$

$$B \in \mathbb{R}^{n \times p}$$

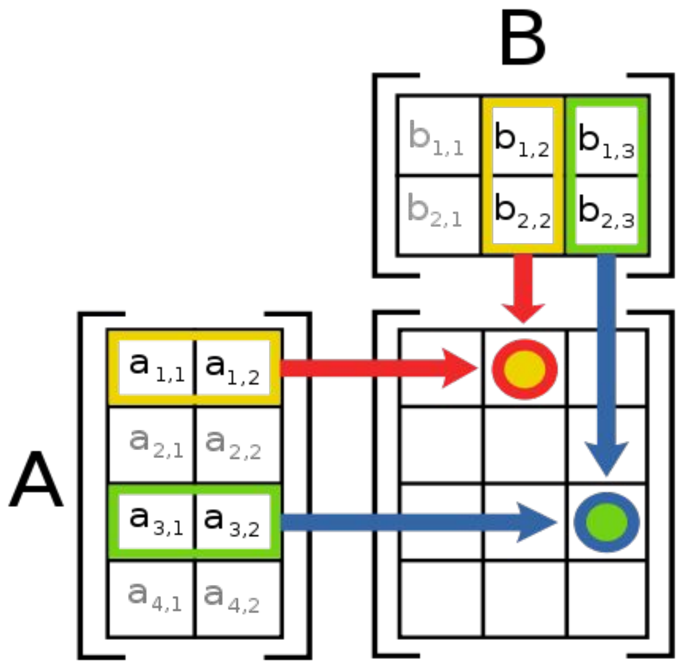
$$C = AB \in \mathbb{R}^{m \times p}$$

$$C_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

$$C = AB = \begin{bmatrix} — & a_1^T & — \\ — & a_2^T & — \\ & \vdots & \\ — & a_m^T & — \end{bmatrix} \begin{bmatrix} | & | & & | \\ b_1 & b_2 & \cdots & b_p \\ | & | & & | \end{bmatrix} = \begin{bmatrix} a_1^T b_1 & a_1^T b_2 & \cdots & a_1^T b_p \\ a_2^T b_1 & a_2^T b_2 & \cdots & a_2^T b_p \\ \vdots & \vdots & \ddots & \vdots \\ a_m^T b_1 & a_m^T b_2 & \cdots & a_m^T b_p \end{bmatrix}.$$

# Matrix Operations

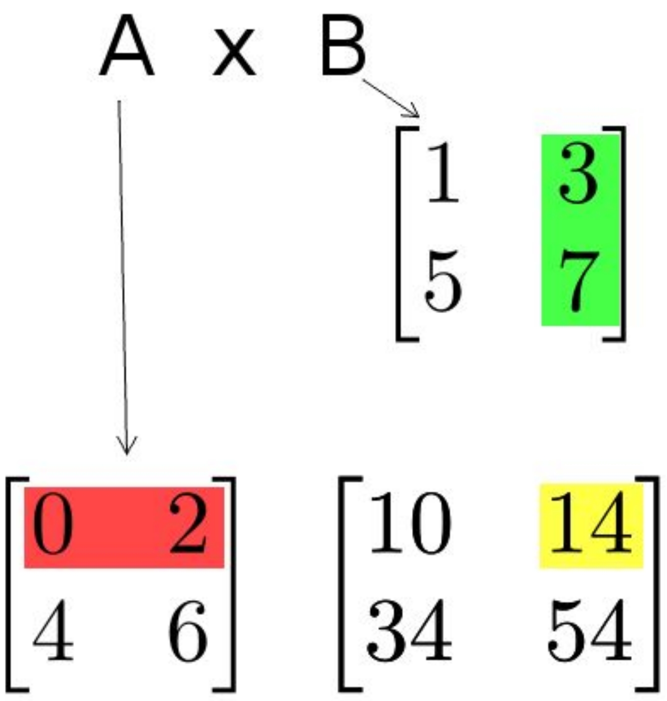
- Multiplication
- The product AB is:




- Each entry in the result is (that row of A) dot product with (that column of B)
- Many uses, which will be covered later

# Matrix Operations

- Multiplication example:




$$0 \cdot 3 + 2 \cdot 7 = 14$$

  – Each entry of the matrix product is made by taking the dot product of the corresponding row in the left matrix, with the corresponding column in the right one.

# Matrix Operations

- The product of two matrices

Matrix multiplication is associative: $(AB)C = A(BC)$.

Matrix multiplication is distributive: $A(B + C) = AB + AC$.

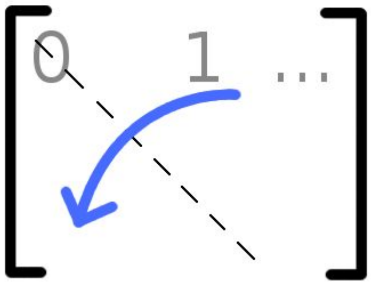
Matrix multiplication is, in general, *not* commutative; that is, it can be the case that $AB \neq BA$. (For example, if $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{n\times q}$, the matrix product $BA$ does not even exist if $m$ and $q$ are not equal!)

# Matrix Operations

- Powers
  - By convention, we can refer to the matrix product AA as $A^2$, and AAA as $A^3$, etc.
  - Obviously only square matrices can be multiplied that way

# Matrix Operations

- Transpose – flip matrix, so row 1 becomes column 1




$$\begin{bmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{bmatrix}^T = \begin{bmatrix} 0 & 2 & 4 \\ 1 & 3 & 5 \end{bmatrix}$$

- A useful identity:

$$(ABC)^T = C^T B^T A^T$$

# Matrix Operations

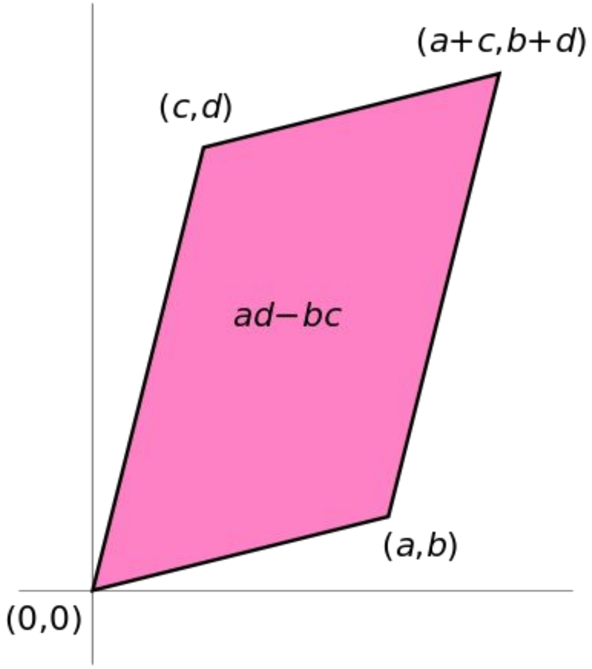
- **Determinant**
  - $\det(\mathbf{A})$ returns a scalar
  - Represents area (or volume) of the parallelogram described by the vectors in the rows of the matrix
  - For $\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$, $\det(\mathbf{A}) = ad - bc$
  - Properties:

$$\det(\mathbf{AB}) = \det(\mathbf{BA})$$

$$\det(\mathbf{A}^{-1}) = \frac{1}{\det(\mathbf{A})}$$

$$\det(\mathbf{A}^T) = \det(\mathbf{A})$$

$$\det(\mathbf{A}) = 0 \Leftrightarrow \mathbf{A} \text{ is singular}$$



# Matrix Operations

- **Trace**

  $\mathrm{tr}(\mathbf{A}) = \text{sum of diagonal elements}$

  $\mathbf{tr}(\begin{bmatrix} 1 & 3 \\ 5 & 7 \end{bmatrix}) = 1 + 7 = 8$

  - Invariant to a lot of transformations, so it's used sometimes in proofs. (Rarely in this class though.)
  - Properties:

$$\mathrm{tr}(\mathbf{AB}) = \mathrm{tr}(\mathbf{BA})$$
$$\mathrm{tr}(\mathbf{A} + \mathbf{B}) = \mathrm{tr}(\mathbf{A}) + \mathrm{tr}(\mathbf{B})$$

# Matrix Operations

- **Vector Norms**

$$\|x\|_1 = \sum_{i=1}^{n} |x_i| \qquad \|x\|_\infty = \max_i |x_i|.$$

$$\|x\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}. \qquad \|x\|_p = \left( \sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

- **Matrix norms:** Norms can also be defined for matrices, such as

$$\|A\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} A_{ij}^2} = \sqrt{\mathrm{tr}(A^T A)}.$$

# Special Matrices

- Identity matrix **`I`**
  - Square matrix, 1's along diagonal, 0's elsewhere
  - **`I`** · [another matrix] = [that matrix]

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

- Diagonal matrix
  - Square matrix with numbers along diagonal, 0's elsewhere
  - A diagonal · [another matrix] scales the rows of that matrix

$$\begin{bmatrix} 3 & 0 & 0 \\ 0 & 7 & 0 \\ 0 & 0 & 2.5 \end{bmatrix}$$

# Special Matrices

- Symmetric matrix

$$\mathbf{A}^T = \mathbf{A}$$

$$\begin{bmatrix} 1 & 2 & 5 \\ 2 & 1 & 7 \\ 5 & 7 & 1 \end{bmatrix}$$

- Skew-symmetric matrix

$$\mathbf{A}^T = -\mathbf{A}$$

$$\begin{bmatrix} 0 & -2 & -5 \\ 2 & 0 & -7 \\ 5 & 7 & 0 \end{bmatrix}$$

# Outline

- Vectors and matrices
  - Basic Matrix Operations
  - Determinants, norms, trace
  - Special Matrices
- Transformation Matrices
  - Homogeneous coordinates
  - Translation
- Matrix inverse
- Matrix rank
- Eigenvalues and Eigenvectors
- Matrix Calculus

Matrix multiplication can be used to transform vectors. A matrix used in this way is called a transformation matrix.

# Transformation

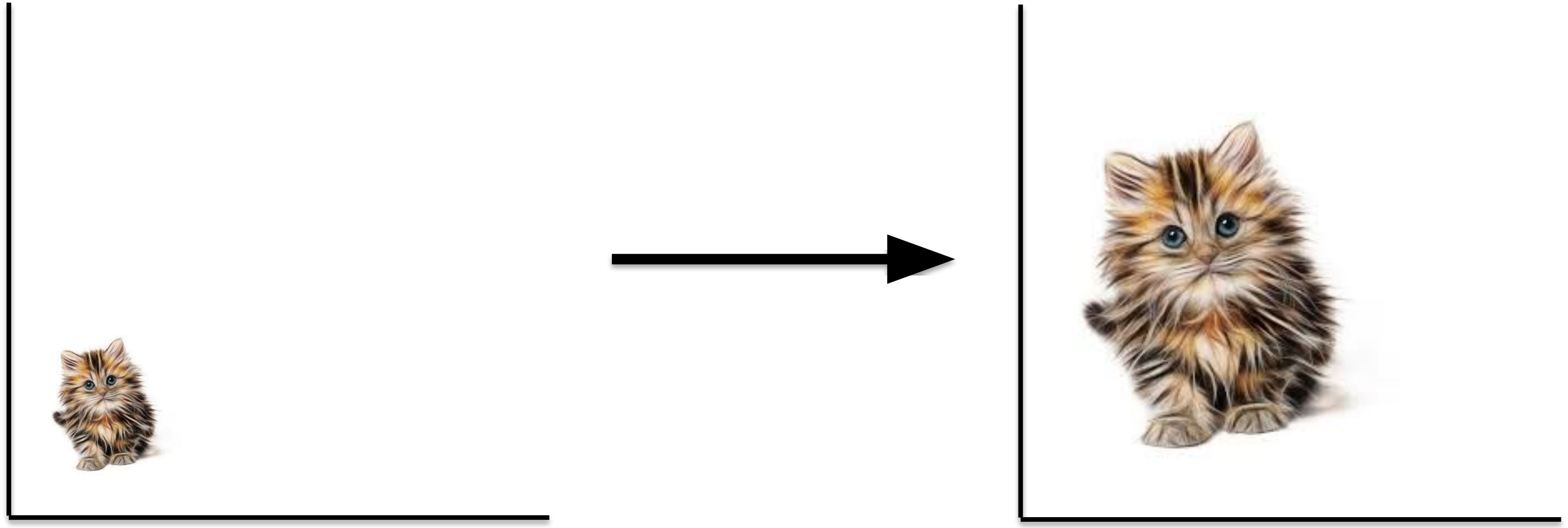
- Matrices can be used to transform vectors in useful ways, through multiplication: x'= Ax
- Simplest is scaling:

$$\begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \times \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} s_x x \\ s_y y \end{bmatrix}$$

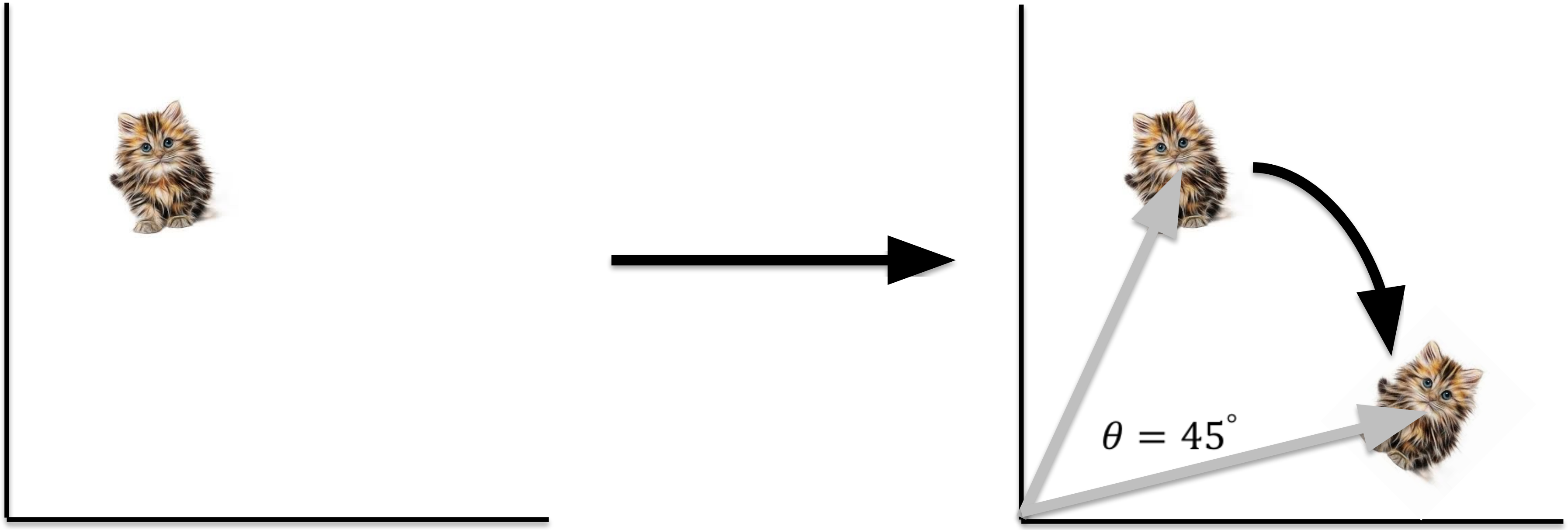
(Verify to yourself that the matrix multiplication works out this way)

# Transformation

$$\begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \times \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} s_x x \\ s_y y \end{bmatrix}$$

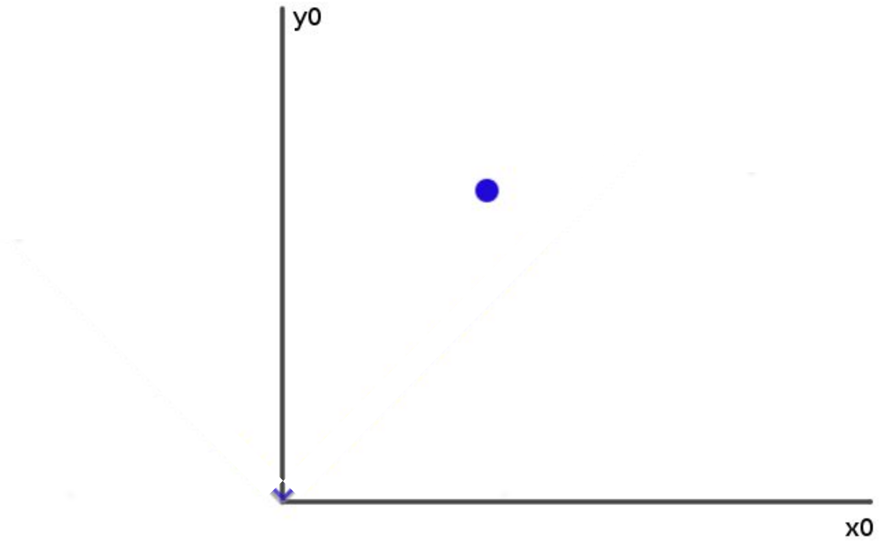


# Rotation



# Rotation

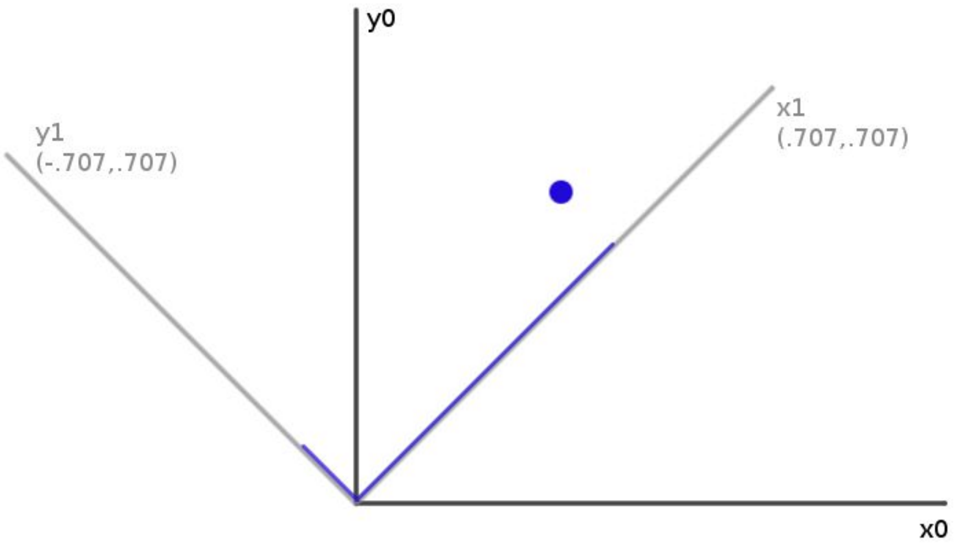
- How can you convert a vector represented in frame "0" to a new, rotated coordinate frame "1"?



# Rotation

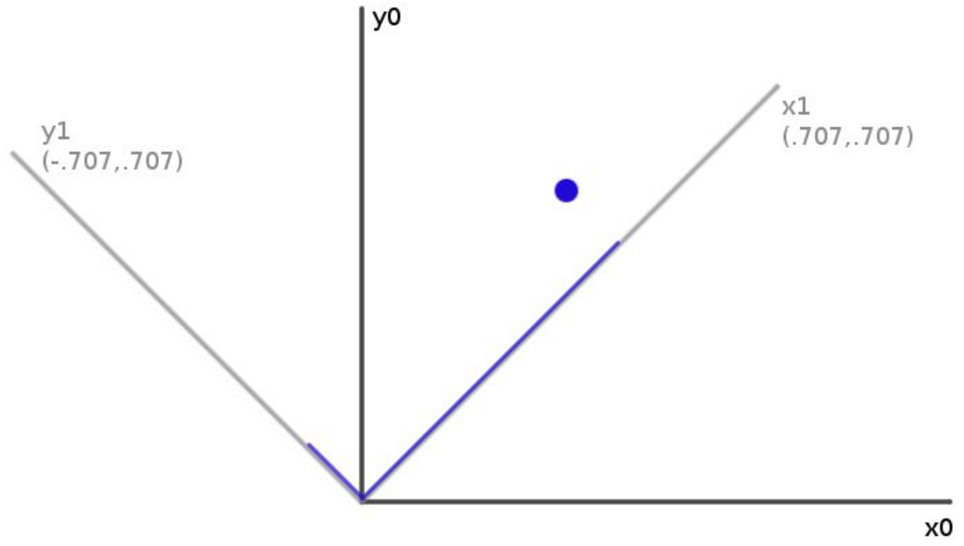
- How can you convert a vector represented in frame "0" to a new, rotated coordinate frame "1"?
- Remember what a vector is:

  [component in direction of the frame's x axis, component in direction of y axis]



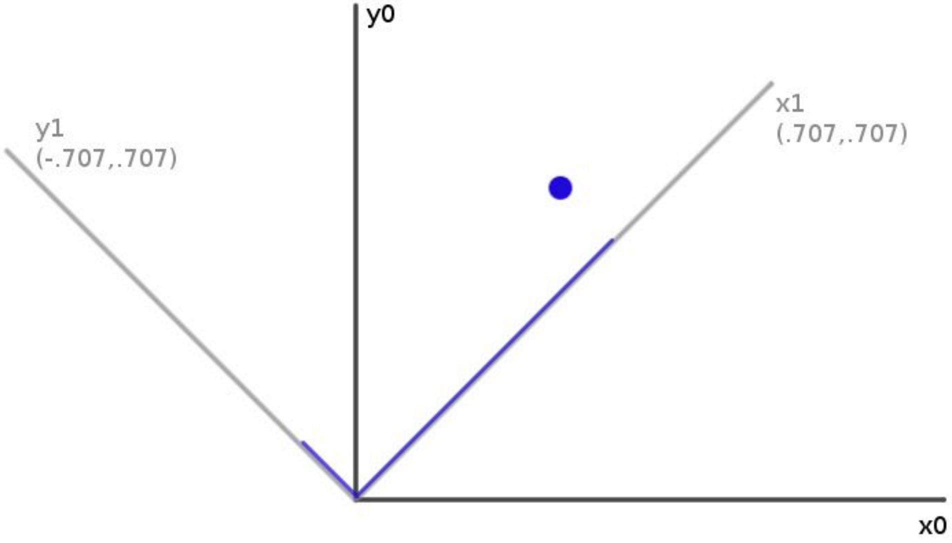
# Rotation

- So to rotate it we must produce this vector:
  [component in direction of **new** x axis, component in direction of **new** y axis]
- We can do this easily with dot products!
- New x coordinate is [original vector] **dot** [the new x axis]
- New y coordinate is [original vector] **dot** [the new y axis]



# Rotation

- Insight: this is what happens in a matrix*vector multiplication
  - Result x coordinate is:
    [original vector] **dot** [matrix row 1]
  - So matrix multiplication can rotate a vector p:

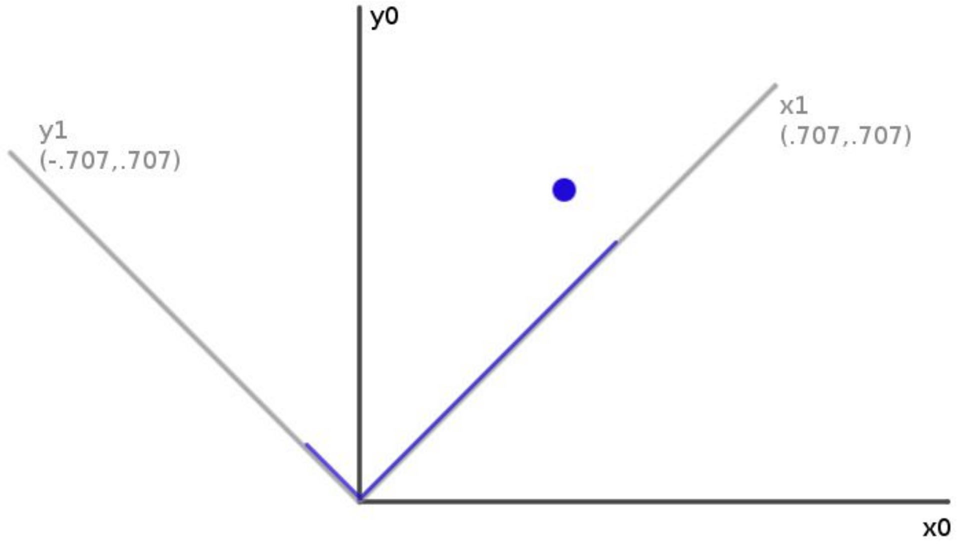



R x p =
rotated p'

$$\overset{R}{\begin{bmatrix} .707 & .707 \\ -.707 & .707 \end{bmatrix}} \qquad \overset{p}{\begin{bmatrix} px \\ py \end{bmatrix}} \qquad \begin{bmatrix} px' \\ py' \end{bmatrix}$$

# Rotation

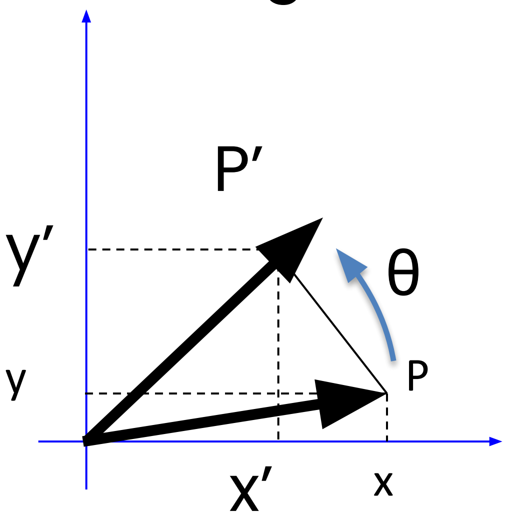
- Suppose we express a point in the new coordinate system which is rotated left
- If we plot the result in the **original** coordinate system, we have rotated the point right




  – Thus, rotation matrices can be used to rotate vectors. We'll usually think of them in that sense-- as operators to rotate vectors

# 2D Rotation Matrix Formula

Counter-clockwise rotation by an angle θ




$$x' = \cos\theta \, x - \sin\theta \, y$$

$$y' = \cos\theta \, y + \sin\theta \, x$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\mathbf{P'} = \mathbf{R} \; \mathbf{P}$$

# Transformation Matrices

- Multiple transformation matrices can be used to transform a point:
  $p' = R_2 R_1 S p$
- The effect of this is to apply their transformations one after the other, from **right to left**.
- In the example above, the result is
  $(R_2 (R_1 (S p)))$
- The result is exactly the same if we multiply the matrices first, to form a single transformation matrix:
  $p' = (R_2 R_1 S) p$

# Homogeneous system

- In general, a matrix multiplication lets us linearly combine componen ts of a vector

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

  – This is sufficient for scale, rotate, skew transformations.
  – But notice, we can't add a constant! ☹

# Homogeneous system

– The (somewhat hacky) solution? Stick a "1" at the end of every vector:

$$\begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} ax + by + c \\ dx + ey + f \\ 1 \end{bmatrix}$$

– Now we can rotate, scale, and skew like before, AND translate (note how the multiplication works out, above)

– This is called "homogeneous coordinates"

# Homogeneous system

– In homogeneous coordinates, the multiplication works out so the rightmost column of the matrix is a vector that gets added.

$$\begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} ax + by + c \\ dx + ey + f \\ 1 \end{bmatrix}$$

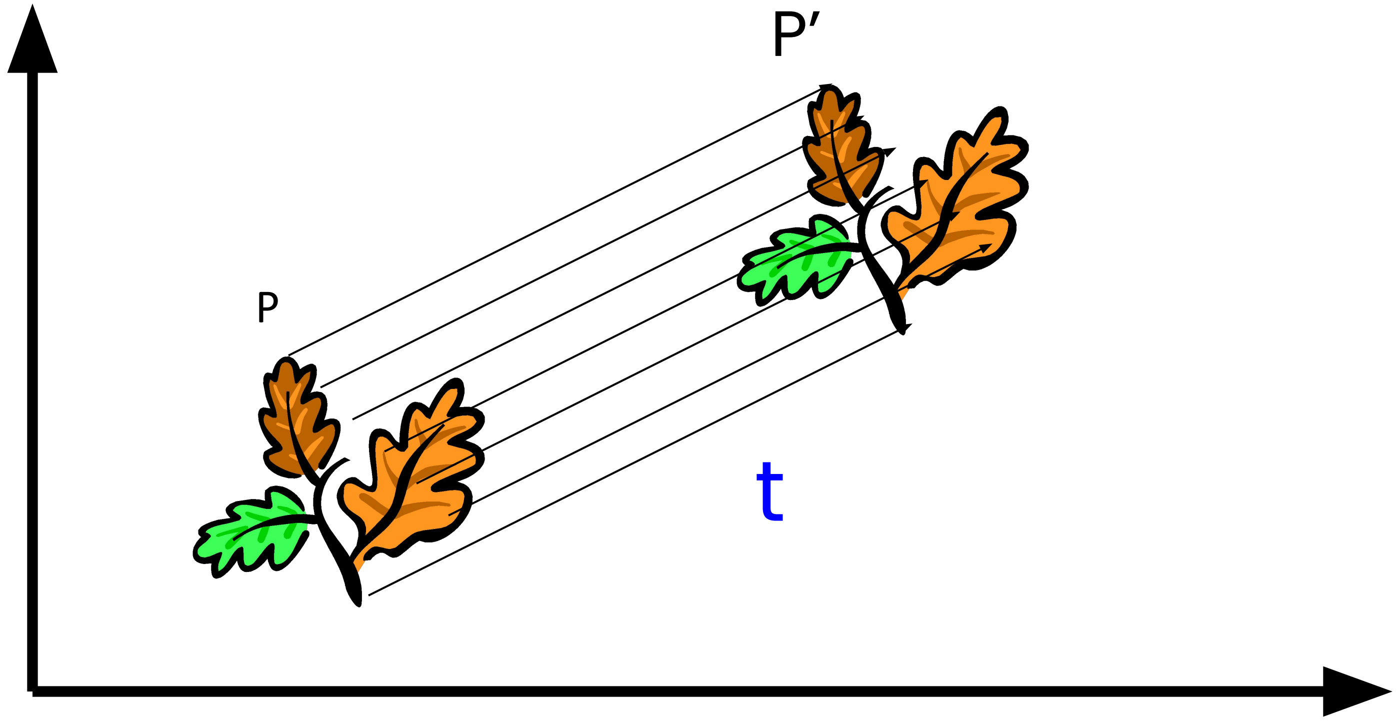
– Generally, a homogeneous transformation matrix will have a bottom row of [0 0 1], so that the result has a “1” at the bottom too.

# Homogeneous system

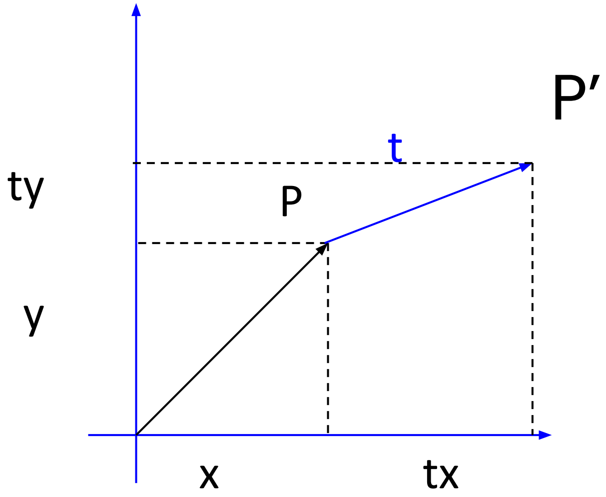
- One more thing we might want: to divide the result by something
  - For example, we may want to divide by a coordinate, to make things scale down as they get farther away in a camera image
  - Matrix multiplication can't actually divide
  - So, **by convention**, in homogeneous coordinates, we'll divide the result by its last coordinate after doing a matrix multiplication

$$\begin{bmatrix} x \\ y \\ 7 \end{bmatrix} \Rightarrow \begin{bmatrix} x/7 \\ y/7 \\ 1 \end{bmatrix}$$

# 2D Translation



# 2D Translation using Homogeneous Coordinates



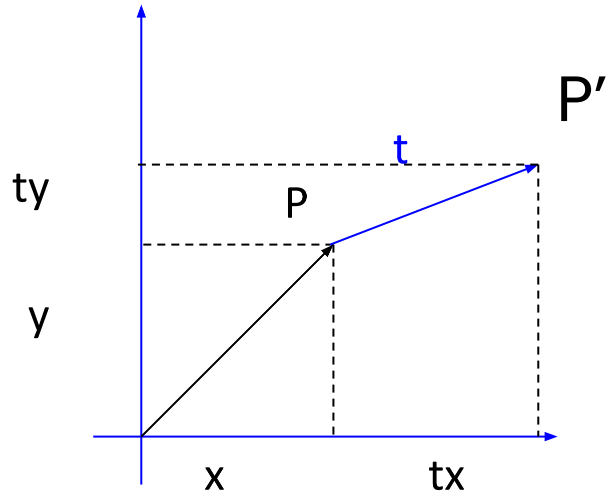

$$\mathbf{P} = (x, y) \rightarrow (x, y, 1)$$

$$\mathbf{t} = (t_x, t_y) \rightarrow (t_x, t_y, 1)$$

P

$$\mathbf{P'} \rightarrow \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix} = \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

# 2D Translation using Homogeneous Coordinates



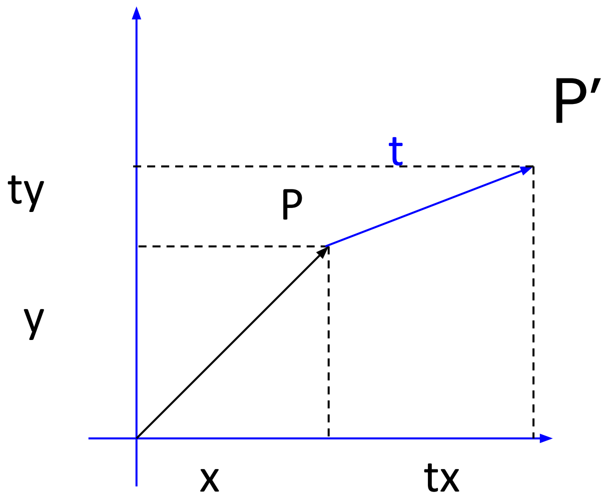

$$\mathbf{P} = (x, y) \rightarrow (x, y, 1)$$

$$\mathbf{t} = (t_x, t_y) \rightarrow (t_x, t_y, 1)$$

P

$$\mathbf{P'} \rightarrow \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & & \\ & & \\ & & \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

# 2D Translation using Homogeneous Coordinates

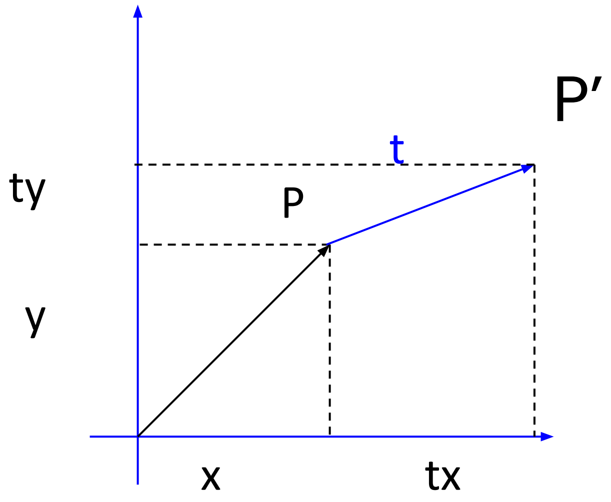



$$\mathbf{P} = (x, y) \rightarrow (x, y, 1)$$

$$\mathbf{t} = (t_x, t_y) \rightarrow (t_x, t_y, 1)$$

$$\mathbf{P'} \rightarrow \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \\ & & \\ & & \end{bmatrix} \cdot \overset{P}{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}$$

# 2D Translation using Homogeneous Coordinates

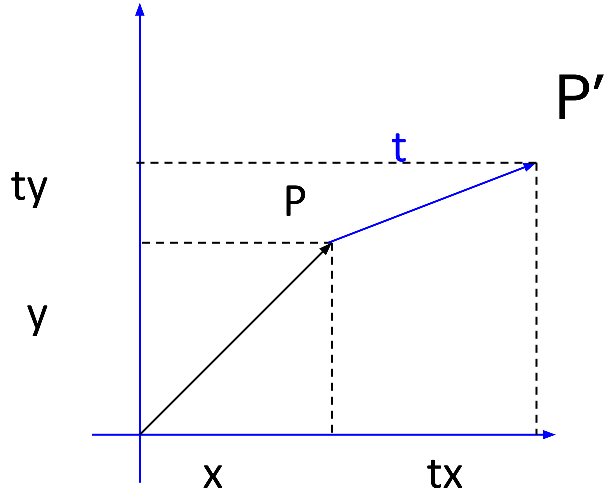



$$\mathbf{P} = (x, y) \rightarrow (x, y, 1)$$

$$\mathbf{t} = (t_x, t_y) \rightarrow (t_x, t_y, 1)$$

$$\mathbf{P}' \rightarrow \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \boxed{t_x} \\ & & \\ & & \end{bmatrix} \cdot \overset{\text{P}}{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}$$

# 2D Translation using Homogeneous Coordinates




$$\mathbf{P} = (x, y) \rightarrow (x, y, 1)$$

$$\mathbf{t} = (t_x, t_y) \rightarrow (t_x, t_y, 1)$$

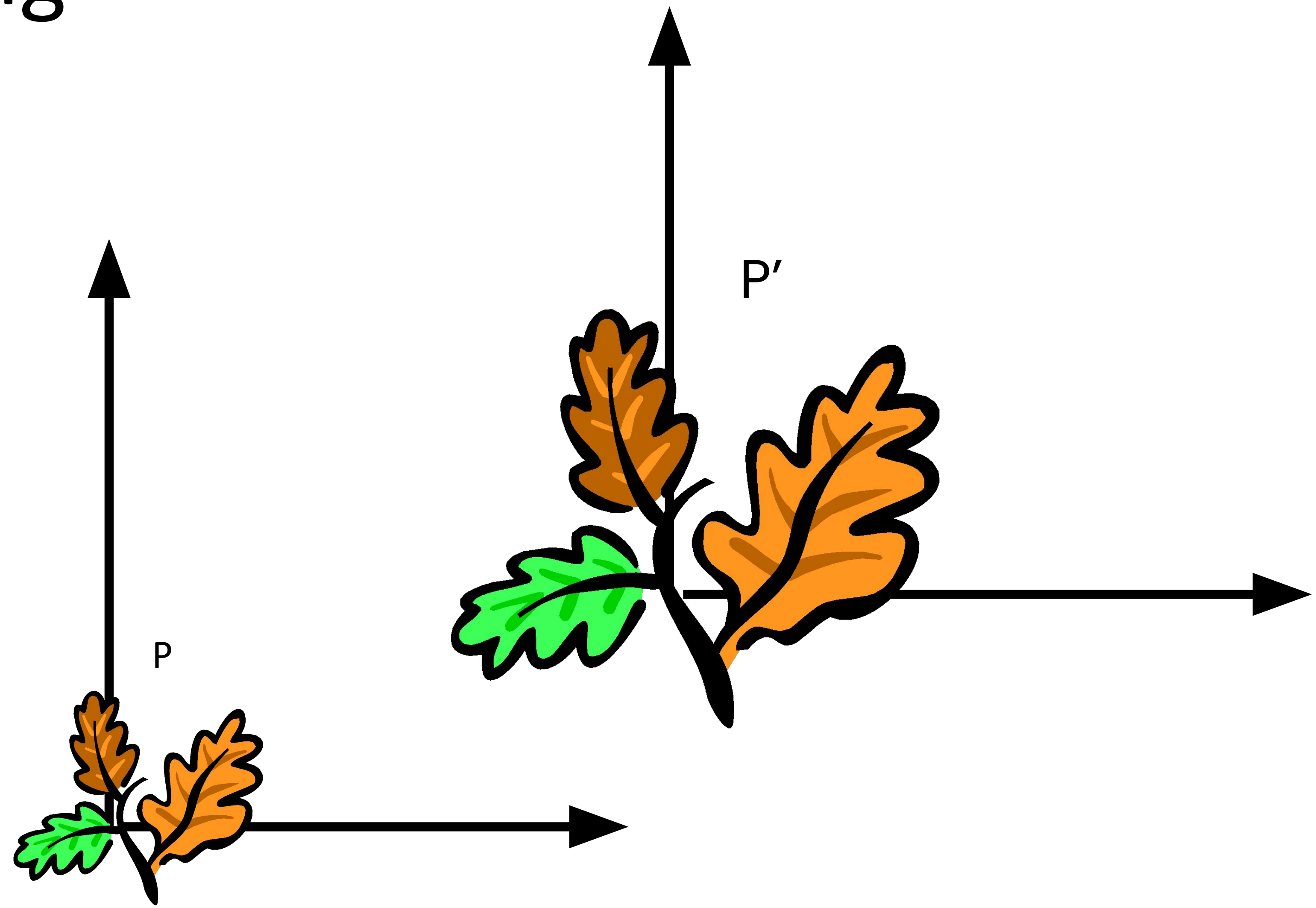
t

P

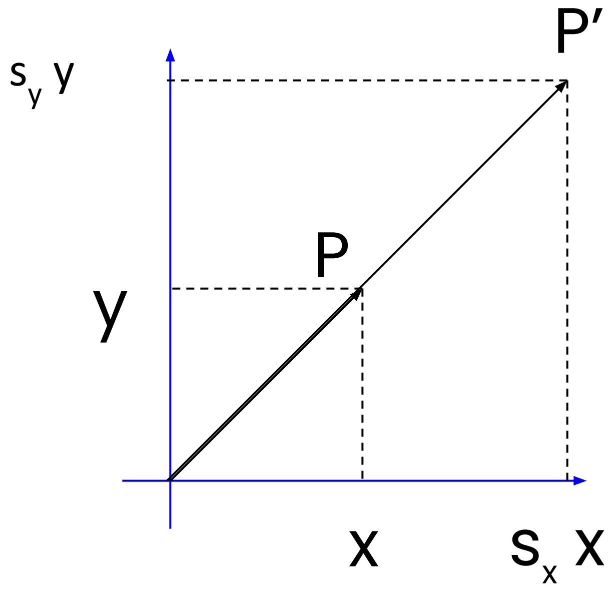
$$\mathbf{P'} \rightarrow \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \cdot \mathbf{P} = \mathbf{T} \cdot \mathbf{P}$$

# Scaling



# Scaling Equation

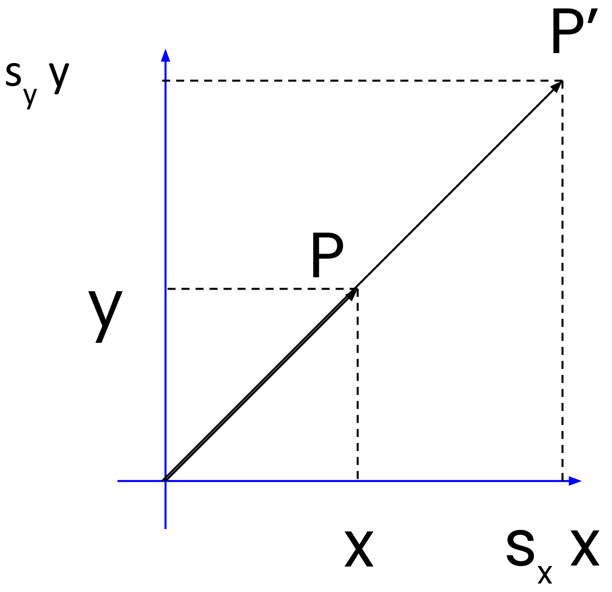



$$\mathbf{P} = (\mathrm{x}, \mathrm{y}) \rightarrow \mathbf{P'} = (\mathrm{s}_{\mathrm{x}}\mathrm{x}, \mathrm{s}_{\mathrm{y}}\mathrm{y})$$

$$\mathbf{P} = (x, y) \rightarrow (x, y, 1)$$

$$\mathbf{P'} = (s_x x, s_y y) \rightarrow (s_x x, s_y y, 1)$$

# Scaling Equation



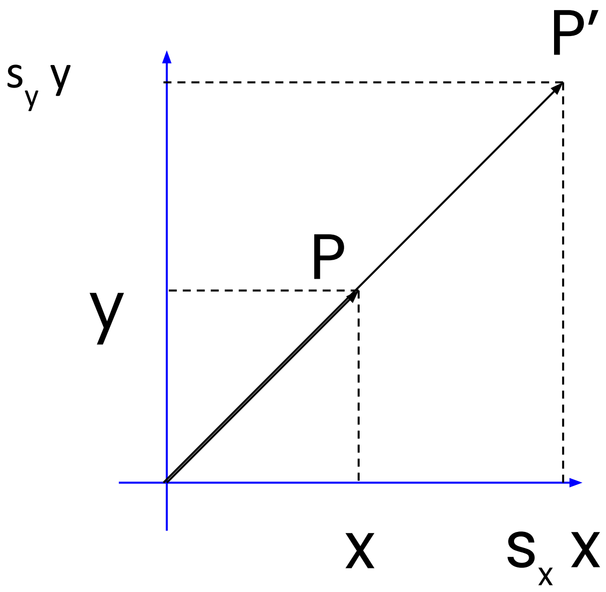

$$\mathbf{P} = (x, y) \rightarrow \mathbf{P}' = (s_x x, s_y y)$$

$$\mathbf{P} = (x, y) \rightarrow (x, y, 1)$$

$$\mathbf{P}' = (s_x x, s_y y) \rightarrow (s_x x, s_y y, 1)$$

$$\mathbf{P}' \rightarrow \begin{bmatrix} s_x x \\ s_y y \\ 1 \end{bmatrix} = \begin{bmatrix} \quad & \quad & \quad \\ & & \\ & & \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

# Scaling Equation



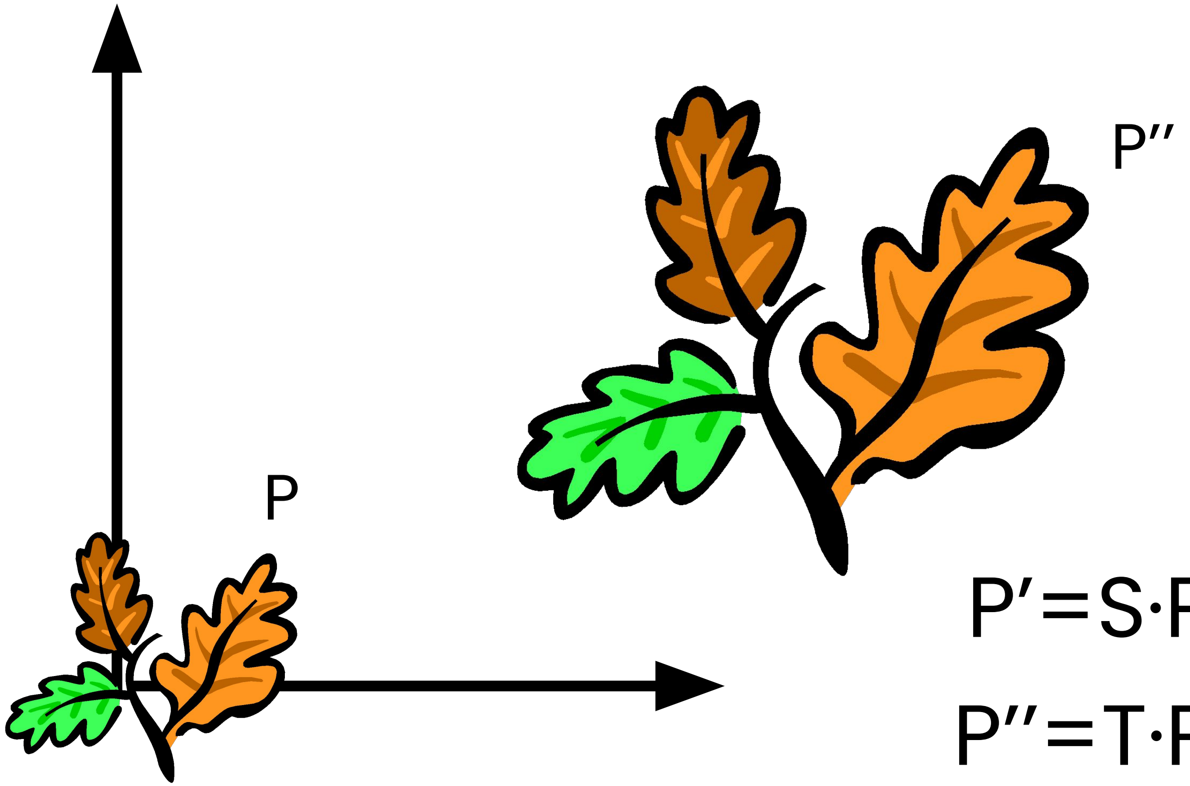

$$\mathbf{P} = (\mathrm{x}, \mathrm{y}) \rightarrow \mathbf{P'} = (\mathrm{s_x x}, \mathrm{s_y y})$$

$$\mathbf{P} = (x, y) \rightarrow (x, y, 1)$$

$$\mathbf{P'} = (s_x x, s_y y) \rightarrow (s_x x, s_y y, 1)$$

$$\mathbf{P'} \rightarrow \begin{bmatrix} s_x x \\ s_y y \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{S}} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{S'} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \cdot \mathbf{P} = \mathbf{S} \cdot \mathbf{P}$$

# Scaling & Translating




$P' = S \cdot P$

$P'' = T \cdot P'$

$P'' = T \cdot P' = T \cdot (S \cdot P) = T \cdot S \cdot P$

# Scaling & Translating

$$\mathbf{P}'' = \mathbf{T} \cdot \mathbf{S} \cdot \mathbf{P} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

# Scaling & Translating

$$\mathbf{P}'' = \mathbf{T} \cdot \mathbf{S} \cdot \mathbf{P} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} =$$

$$= \begin{bmatrix} s_x & 0 & t_x \\ 0 & s_y & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x x + t_x \\ s_y y + t_y \\ 1 \end{bmatrix} = \begin{bmatrix} S & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

# Translating & Scaling != Scaling & Translating

$$\mathbf{P}''' = \mathbf{T} \cdot \mathbf{S} \cdot \mathbf{P} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & t_x \\ 0 & s_y & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x x + t_x \\ s_y y + t_y \\ 1 \end{bmatrix}$$

# Translating & Scaling != Scaling & Translating

$$\mathbf{P}''' = \mathbf{T} \cdot \mathbf{S} \cdot \mathbf{P} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & t_x \\ 0 & s_y & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x x + t_x \\ s_y y + t_y \\ 1 \end{bmatrix}$$

$$\mathbf{P}''' = \mathbf{S} \cdot \mathbf{T} \cdot \mathbf{P} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} =$$

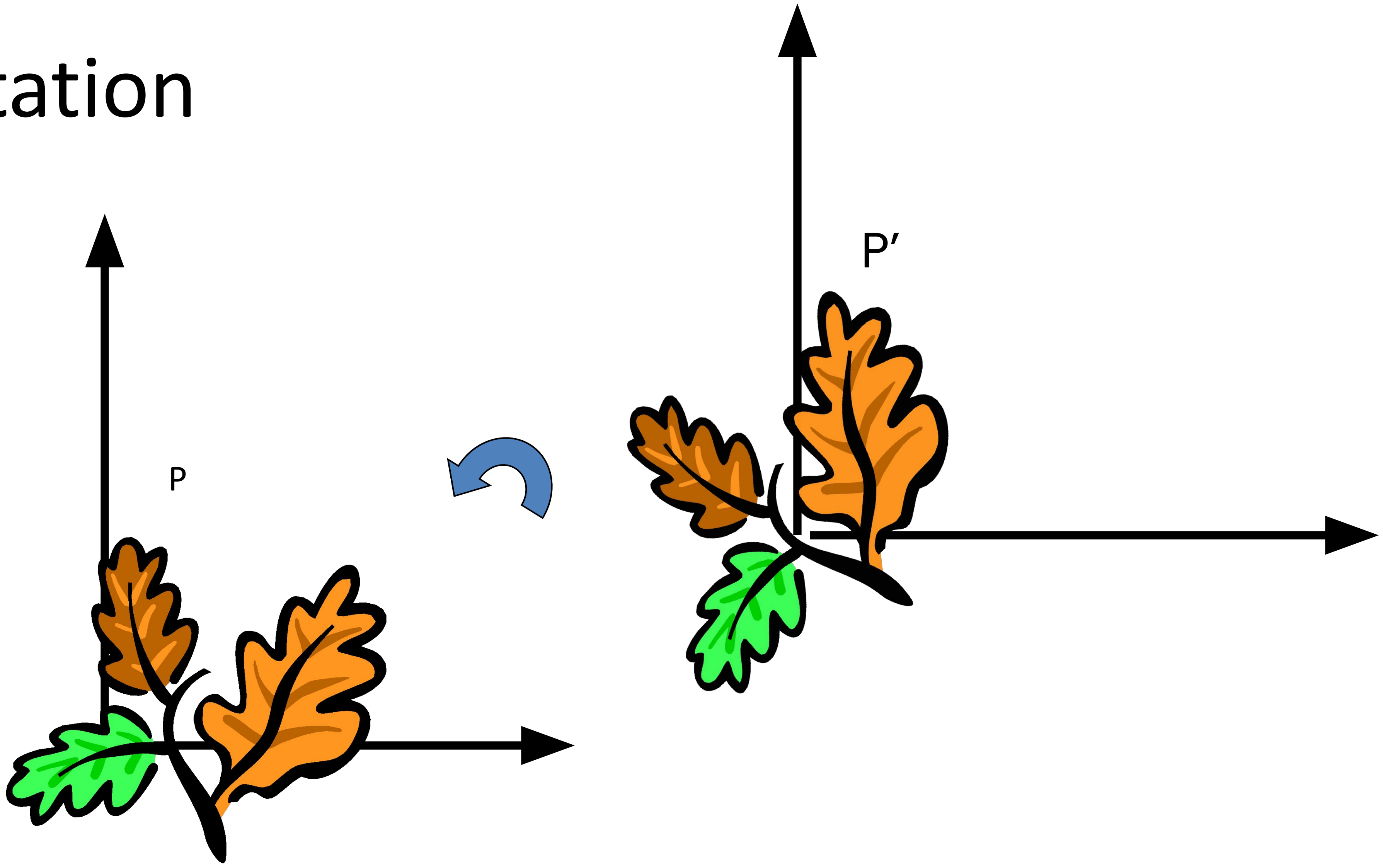
# Translating & Scaling != Scaling & Translating

$$\mathbf{P}''' = \mathbf{T}\cdot\mathbf{S}\cdot\mathbf{P} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & t_x \\ 0 & s_y & t_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x x + t_x \\ s_y y + t_y \\ 1 \end{bmatrix}$$

$$\mathbf{P}''' = \mathbf{S}\cdot\mathbf{T}\cdot\mathbf{P} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} =$$

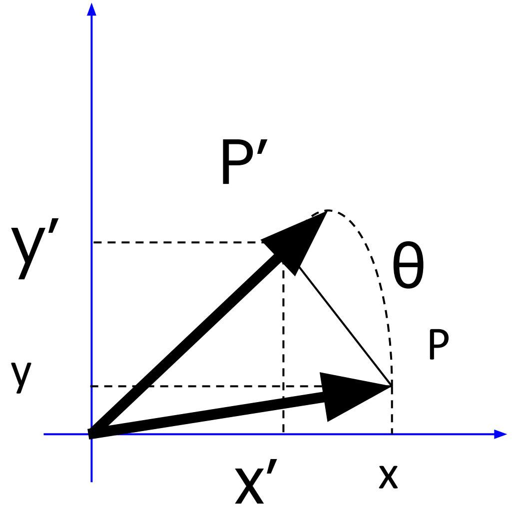
$$= \begin{bmatrix} s_x & 0 & s_x t_x \\ 0 & s_y & s_y t_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x x + s_x t_x \\ s_y y + s_y t_y \\ 1 \end{bmatrix}$$

# Rotation



# Rotation Equations

Counter-clockwise rotation by an angle θ




$$x' = \cos\theta\, x - \sin\theta\, y$$
$$y' = \cos\theta\, y + \sin\theta\, x$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\mathbf{P'} = \mathbf{R}\ \mathbf{P}$$

# Rotation Matrix Properties

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

A 2D rotation matrix is 2x2

Note: R belongs to the category of *normal* matrices
and satisfies many interesting properties:

$$\mathbf{R} \cdot \mathbf{R}^{\mathrm{T}} = \mathbf{R}^{\mathrm{T}} \cdot \mathbf{R} = \mathbf{I}$$

$$\det(\mathbf{R}) = 1$$

# Rotation Matrix Properties

- Transpose of a rotation matrix produces a rotation in the opposite direction

$$\mathbf{R} \cdot \mathbf{R}^{\mathrm{T}} = \mathbf{R}^{\mathrm{T}} \cdot \mathbf{R} = \mathbf{I}$$

$$\det(\mathbf{R}) = 1$$

- The rows of a rotation matrix are always mutually perpendicular (a.k.a. orthogonal) unit vectors
  – (and so are its columns)

# Scaling + Rotation + Translation

P'= (T R S) P

$$\mathbf{P'} = \mathbf{T} \cdot \mathrm{R} \cdot \mathbf{S} \cdot \mathbf{P} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} =$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} =$$

$$= \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} S & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \boxed{\begin{bmatrix} R\,S & t \\ 0 & 1 \end{bmatrix}} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

This is the form of the general-purpose transformation matrix

# Outline

- Vectors and matrices
  - Basic Matrix Operations
  - Determinants, norms, trace
  - Special Matrices
- Transformation Matrices
  - Homogeneous coordinates
  - Translation
- **Matrix inverse**
- Matrix rank
- Eigenvalues and Eigenvectors
- Matrix Calculate

The inverse of a transformation matrix reverses its effect

# Inverse

- Given a matrix **A**, its inverse $\mathbf{A}^{-1}$ is a matrix such that $\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}$
- E.g. $\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}^{-1} = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{3} \end{bmatrix}$
- Inverse does not always exist. If $\mathbf{A}^{-1}$ exists, **A** is *invertible* or *non-singular*. Otherwise, it's *singular*.
- Useful identities, for matrices that are invertible:

$$(\mathbf{A}^{-1})^{-1} = \mathbf{A}$$

$$(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}$$

$$\mathbf{A}^{-T} \triangleq (\mathbf{A}^{T})^{-1} = (\mathbf{A}^{-1})^{T}$$

# Matrix Operations

- Pseudoinverse
  – Fortunately, there are workarounds to solve AX=B in these situations. And python can do them!
  – Instead of taking an inverse, directly ask python to solve for X in AX=B, by typing **np.linalg.solve(A, B)**
  – Python will try several appropriate numerical methods (including the pseudoinverse if the inverse doesn’t exist)
  – Python will return the value of X which solves the equation
    - If there is no exact solution, it will return the closest one
    - If there are many solutions, it will return the smallest one

# Matrix Operations

- Python example:

$$AX = B$$

$$A = \begin{bmatrix} 2 & 2 \\ 3 & 4 \end{bmatrix}, B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

```
>> import numpy as np
>> x = np.linalg.solve(A,B)
x =
    1.0000
   -0.5000
```

# Outline

- Vectors and matrices
  - Basic Matrix Operations
  - Determinants, norms, trace
  - Special Matrices
- Transformation Matrices
  - Homogeneous coordinates
  - Translation
- Matrix inverse
- Matrix rank
- Eigenvalues and Eigenvectors
- Matrix Calculate

The rank of a transformation matrix tells you how many dimensions it transforms a vector to.

# Linear independence

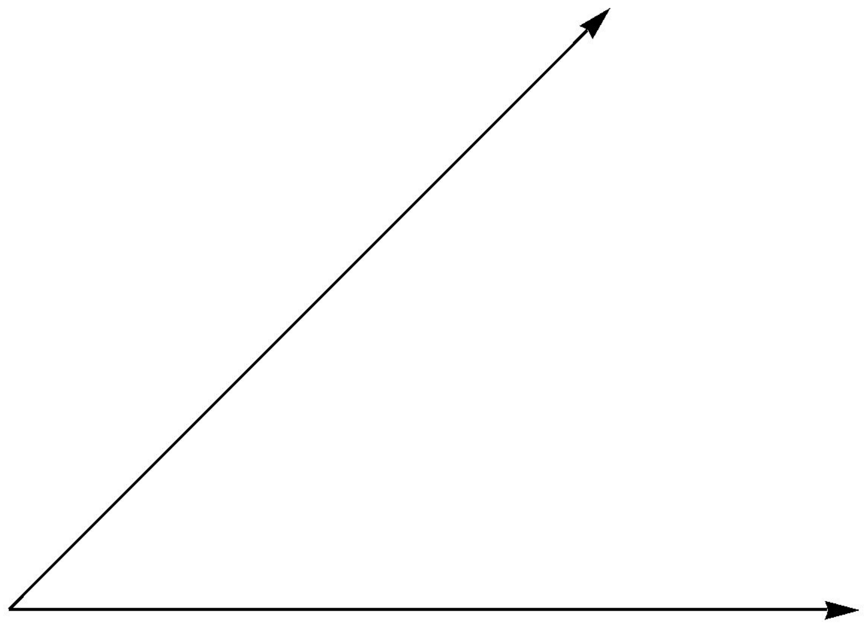
- Suppose we have a set of vectors $\mathbf{v_1, \ldots, v_n}$
- If we can express $\mathbf{v}_1$ as a linear combination of the other vectors $\mathbf{v}_2 \ldots \mathbf{v}_n$, then $\mathbf{v}_1$ is linearly *dependent* on the other vectors.
  - The direction $\mathbf{v}_1$ can be expressed as a combination of the directions $\mathbf{v}_2 \ldots \mathbf{v}_n$. (E.g. $\mathbf{v}_1 = .7\ \mathbf{v}_2 - .7\ \mathbf{v}_4$)

# Linear independence

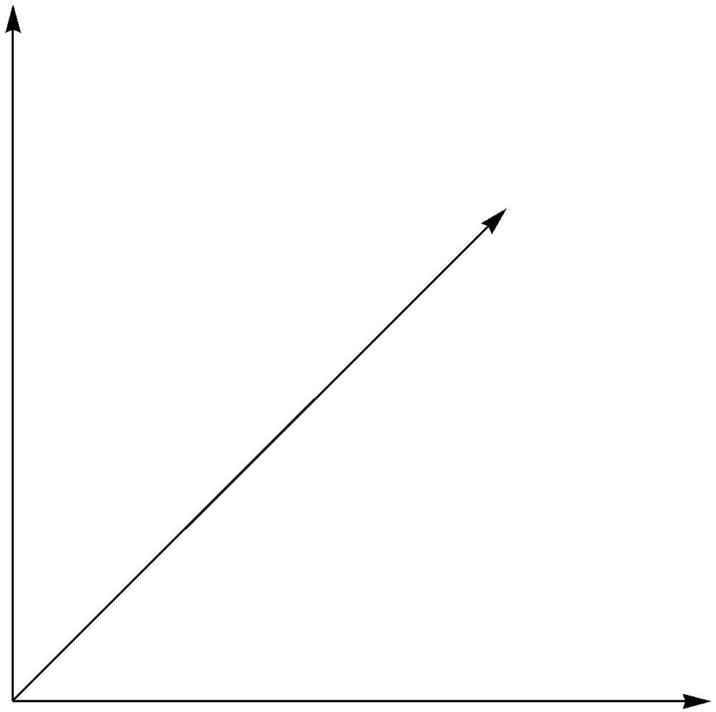
- Suppose we have a set of vectors $\mathbf{v_1}, \ldots, \mathbf{v_n}$
- If we can express $\mathbf{v}_1$ as a linear combination of the other vectors $\mathbf{v}_2 \ldots \mathbf{v}_n$, then $\mathbf{v}_1$ is linearly *dependent* on the other vectors.
  - The direction $\mathbf{v}_1$ can be expressed as a combination of the directions $\mathbf{v}_2 \ldots \mathbf{v}_n$. (E.g. $\mathbf{v}_1 = .7\, \mathbf{v}_2 - .7\, \mathbf{v}_4$)
- If no vector is linearly dependent on the rest of the set, the set is linearly *independent*.
  - Common case: a set of vectors $\mathbf{v_1}, \ldots, \mathbf{v_n}$ is always linearly independent if each vector is perpendicular to every other vector (and non-zero)

# Linear independence

Linearly independent set



Not linearly independent



# Matrix rank

- Column/row rank

[redacted]

  – Column rank always equals row rank

- Matrix rank

$$\mathrm{rank}(\mathbf{A}) \triangleq \text{col-rank}(\mathbf{A}) = \text{row-rank}(\mathbf{A})$$

# Matrix rank

- For transformation matrices, the rank tells you the dimensions of the output
- E.g. if rank of **A** is 1, then the transformation

**p'=A**p

maps points onto a line.

- Here's a matrix with rank 1:

$$\begin{bmatrix} 1 & 1 \\ 2 & 2 \end{bmatrix} \times \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + y \\ 2x + 2y \end{bmatrix}$$

All points get mapped to the line y=2x

# Matrix rank

- If an $m$ x $m$ matrix is rank $m$, we say it's "full rank"
  - Maps an $m$ x 1 vector uniquely to another $m$ x 1 vector
  - An inverse matrix can be found
- If rank < $m$, we say it's "singular"
  - At least one dimension is getting collapsed. No way to look at the result and tell what the input was
  - Inverse does not exist
- Inverse also doesn't exist for non-square matrices

# Outline

- Vectors and matrices
  - Basic Matrix Operations
  - Determinants, norms, trace
  - Special Matrices
- Transformation Matrices
  - Homogeneous coordinates
  - Translation
- Matrix inverse
- Matrix rank
- Eigenvalues and Eigenvectors(SVD)
- Matrix Calculus

# Eigenvector and Eigenvalue

- An eigenvector **x** of a linear transformation *A* is a non-zero vector that, when *A* is applied to it, does not change direction.

$$Ax = \lambda x, \quad x \neq 0.$$

# Eigenvector and Eigenvalue

- An eigenvector **x** of a linear transformation *A* is a non-zero vector that, when *A* is applied to it, does not change direction.
- Applying *A* to the eigenvector only scales the eigenvector by the scalar value *λ*, called an eigenvalue.

$$Ax = \lambda x, \quad x \neq 0.$$

# Eigenvector and Eigenvalue

- We want to find all the eigenvalues of A:

$$Ax = \lambda x, \quad x \neq 0.$$

- Which can we written as:

$$Ax = (\lambda I)x \quad x \neq 0.$$

- Therefore:

$$(\lambda I - A)x = 0, \quad x \neq 0.$$

# Eigenvector and Eigenvalue

- We can solve for eigenvalues by solving:

$$(\lambda I - A)x = 0, \quad x \neq 0.$$

- Since we are looking for non-zero **x**, we can instead solve the above equation as:

$$|(\lambda I - A)| = 0.$$

# Properties

- The trace of a A is equal to the sum of its eigenvalues:

$$\mathrm{tr}A = \sum_{i=1}^{n} \lambda_i.$$

- The determinant of A is equal to the product of its eigenvalues

$$|A| = \prod_{i=1}^{n} \lambda_i.$$

- The rank of A is equal to the number of non-zero eigenvalues of A.
- The eigenvalues of a diagonal matrix D = diag(d1, . . . dn) are just the diagonal entries d1, . . . dn

# Spectral theory

- We call an eigenvalue λ and an associated eigenvector an **eigenpair**.
- The space of vectors where (A − λI) = 0 is often called the **eigenspace** of A associated with the eigenvalue λ.
- The set of all eigenvalues of A is called its **spectrum**:

$$\sigma(A) = \{\lambda \in \mathbb{C} : \lambda I - A \text{ is singular}\}.$$

# Spectral theory

- The magnitude of the largest eigenvalue (in magnitude) is called the spectral radius

$$\rho(A) = \max\{|\lambda_1|, \ldots, |\lambda_n|\}$$

  – Where C is the space of all eigenvalues of A

# Spectral theory

- The spectral radius is bounded by infinity norm of a matrix: $$\rho(A) = \lim_{k \to \infty} \|A^k\|^{1/k}$$
- Proof: Turn to a partner and prove this!

# Spectral theory

- The spectral radius is bounded by infinity norm of a matrix: $$\rho(A) = \lim_{k\to\infty} \|A^k\|^{1/k}$$
- Proof: Let λ and v be an eigenpair of A:

$$|\lambda|^k \|\mathbf{v}\| = \|\lambda^k \mathbf{v}\| = \|A^k \mathbf{v}\| \leq \|A^k\| \cdot \|\mathbf{v}\|$$

and since $\mathbf{v} \neq 0$ we have

$$|\lambda|^k \leq \|A^k\|$$

and therefore

$$\rho(A) \leq \|A^k\|^{\frac{1}{k}}.$$

# Diagonalization

- An $n \times n$ matrix A is diagonalizable if it has n linearly independent eigenvectors.
- Most square matrices (in a sense that can be made mathematically rigorous) are diagonalizable:
  - Normal matrices are diagonalizable
  - Matrices with n distinct eigenvalues are diagonalizable

**Lemma**: Eigenvectors associated with distinct eigenvalues are linearly independent.

# Diagonalization

- An $n \times n$ matrix A is diagonalizable if it has n linearly independent eigenvectors.
- Most square matrices are diagonalizable:
  - Normal matrices are diagonalizable
  - Matrices with n distinct eigenvalues are diagonalizable

**Lemma**: Eigenvectors associated with distinct eigenvalues are linearly independent.

# Diagonalization

- Eigenvalue equation:

$$AV = VD$$
$$A = VDV^{-1}$$

  – Where D is a diagonal matrix of the eigenvalues

$$\begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix}$$

# Diagonalization

- Eigenvalue equation:

$$AV = VD$$

$$A = VDV^{-1}$$

- Assuming all $\lambda_i$'s are unique:

$$A = VDV^{T}$$

  - Remember that the inverse of an orthogonal matrix is just its transpose and the eigenvectors are orthogonal

# Symmetric matrices

- Properties:
  - For a symmetric matrix A, all the eigenvalues are real.
  - The eigenvectors of A are orthonormal.

$$A = VDV^T$$

# Symmetric matrices

- Therefore:

$$x^T A x = x^T V D V^T x = y^T D y \;= \sum_{i=1}^{n} \lambda_i y_i^2$$

  - where $y = V^T x$

- So, what can you say about the vector x that satisfies the following optimization? $\max_{x \in \mathbb{R}^n} \; x^T A x \quad \text{subject to } \|x\|_2^2 = 1$

# Symmetric matrices

- Therefore:

$$x^T A x = x^T V D V^T x = y^T D y \; = \sum_{i=1}^{n} \lambda_i y_i^2$$

  – where $\quad y = V^T x$

- So, what can you say about the vector x that satisfies the following optimization? $\max_{x \in \mathbb{R}^n} \; x^T A x \quad \text{subject to } \|x\|_2^2 = 1$
  – Is the same as finding the eigenvector that corresponds to the largest eigenvalue of A.

# Some applications of Eigenvalues

- PageRank
- Schrodinger’s equation
- PCA

- We are going to use it to compress images in future classes

# Outline

- Vectors and matrices
  - Basic Matrix Operations
  - Determinants, norms, trace
  - Special Matrices
- Transformation Matrices
  - Homogeneous coordinates
  - Translation
- Matrix inverse
- Matrix rank
- Eigenvalues and Eigenvectors(SVD)
- Matrix Calculus

# Matrix Calculus – The Gradient

- Let a function $f : \mathbb{R}^{m \times n} \to \mathbb{R}$ take as input a matrix A of size m × n and return a real value.
- Then the **gradient** of **f:**

$$\nabla_A f(A) \in \mathbb{R}^{m \times n} = \begin{bmatrix} \frac{\partial f(A)}{\partial A_{11}} & \frac{\partial f(A)}{\partial A_{12}} & \cdots & \frac{\partial f(A)}{\partial A_{1n}} \\ \frac{\partial f(A)}{\partial A_{21}} & \frac{\partial f(A)}{\partial A_{22}} & \cdots & \frac{\partial f(A)}{\partial A_{2n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f(A)}{\partial A_{m1}} & \frac{\partial f(A)}{\partial A_{m2}} & \cdots & \frac{\partial f(A)}{\partial A_{mn}} \end{bmatrix}$$

# Matrix Calculus – The Gradient

- Every entry in the matrix is: $\nabla_A f(A))_{ij} = \frac{\partial f(A)}{\partial A_{ij}}.$
- the size of $\nabla_A f(A)$ is always the same as the size of A. So if A is just a vector x:

$$\nabla_x f(x) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \frac{\partial f(x)}{\partial x_2} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix}$$

# Exercise

- Example:

For $x \in \mathbb{R}^n$, let $f(x) = b^T x$ for some known vector $b \in \mathbb{R}^n$

$$f(x) = \begin{bmatrix} b_1 & b_2 & \dots & b_n \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

- Find:

$$\frac{\partial f(x)}{\partial x_k} = ?$$

$$\nabla_x f(x) = ?$$

# Exercise

- Example:

For $x \in \mathbb{R}^n$, let $f(x) = b^T x$ for some known vector $b \in \mathbb{R}^n$

$$f(x) = \sum_{i=1}^{n} b_i x_i$$

$$\frac{\partial f(x)}{\partial x_k} = \frac{\partial}{\partial x_k} \sum_{i=1}^{n} b_i x_i = b_k.$$

- From this we can conclude that: $\nabla_x b^T x = b.$

# Matrix Calculus – The Gradient

- Properties

  - $\nabla_x(f(x) + g(x)) = \nabla_x f(x) + \nabla_x g(x)$.
  - For $t \in \mathbb{R}$, $\nabla_x(t\ f(x)) = t\nabla_x f(x)$.

# Matrix Calculus – The Hessian

- The Hessian matrix with respect to x, written $\nabla_x^2 f(x)$ or simply as H:

$$\nabla_x^2 f(x) \in \mathbb{R}^{n\times n} = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x)}{\partial x_2^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \frac{\partial^2 f(x)}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix}$$

- The Hessian of n-dimensional vector is the n × n matrix.

# Matrix Calculus – The Hessian

- Each entry can be written as: $$\nabla_x^2 f(x))_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}.$$

- Exercise: Why is the Hessian always symmetric?

# Matrix Calculus – The Hessian

- Each entry can be written as: $$\nabla_x^2 f(x))_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}.$$

- The Hessian is always symmetric, because

$$\frac{\partial^2 f(x)}{\partial x_i \partial x_j} = \frac{\partial^2 f(x)}{\partial x_j \partial x_i}.$$

- This is known as Schwarz's theorem: The order of partial derivatives don’t matter as long as the second derivative exists and is continuous.

# Matrix Calculus – The Hessian

- Note that the hessian is not the gradient of whole gradient of a vector (this is not defined). It is actually the gradient of **every entry** of the gradient of the vector.

$$\nabla_x^2 f(x) \in \mathbb{R}^{n \times n} = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x)}{\partial x_2^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \frac{\partial^2 f(x)}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix}$$

# Matrix Calculus – The Hessian

- Eg, the first column is the gradient of $\frac{\partial f(x)}{\partial x_1}$

$$\nabla_x^2 f(x) \in \mathbb{R}^{n\times n} = \begin{bmatrix} \boxed{\begin{matrix} \frac{\partial^2 f(x)}{\partial x_1^2} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1} \\ \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} \end{matrix}} & \begin{matrix} \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f(x)}{\partial x_2^2} \\ \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_2} \end{matrix} & \begin{matrix} \cdots \\ \cdots \\ \ddots \\ \cdots \end{matrix} & \begin{matrix} \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \vdots \\ \frac{\partial^2 f(x)}{\partial x_n^2} \end{matrix} \end{bmatrix}$$

# Exercise

- Example:

consider the quadratic function $f(x) = x^T Ax$

$$f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j$$

$$\frac{\partial f(x)}{\partial x_k} = \frac{\partial}{\partial x_k} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j$$

# Exercise

$$\frac{\partial f(x)}{\partial x_k} \quad = \quad \frac{\partial}{\partial x_k} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j$$

# Exercise

$$
\begin{aligned}
\frac{\partial f(x)}{\partial x_k} &= \frac{\partial}{\partial x_k} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j \\
&= \frac{\partial}{\partial x_k} \left[ \sum_{i \neq k} \sum_{j \neq k} A_{ij} x_i x_j + \sum_{i \neq k} A_{ik} x_i x_k + \sum_{j \neq k} A_{kj} x_k x_j + A_{kk} x_k^2 \right]
\end{aligned}
$$

Divide the summation into 3 parts depending on whether:

- i == k or
- j == k

# Exercise

$$
\begin{aligned}
\frac{\partial f(x)}{\partial x_k} &= \frac{\partial}{\partial x_k} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j \\
&= \frac{\partial}{\partial x_k} \left[ \boxed{\sum_{i \neq k} \sum_{j \neq k} A_{ij} x_i x_j} + \sum_{i \neq k} A_{ik} x_i x_k + \sum_{j \neq k} A_{kj} x_k x_j + A_{kk} x_k^2 \right] \\
&= \sum_{i \neq k} A_{ik} x_i + \sum_{j \neq k} A_{kj} x_j + 2 A_{kk} x_k
\end{aligned}
$$

# Exercise

$$
\begin{aligned}
\frac{\partial f(x)}{\partial x_k} &= \frac{\partial}{\partial x_k} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j \\
&= \frac{\partial}{\partial x_k} \left[ \sum_{i \neq k} \sum_{j \neq k} A_{ij} x_i x_j + \boxed{\sum_{i \neq k} A_{ik} x_i x_k} + \sum_{j \neq k} A_{kj} x_k x_j + A_{kk} x_k^2 \right] \\
&= \boxed{\sum_{i \neq k} A_{ik} x_i} + \sum_{j \neq k} A_{kj} x_j + 2 A_{kk} x_k
\end{aligned}
$$

# Exercise

$$
\begin{aligned}
\frac{\partial f(x)}{\partial x_k} &= \frac{\partial}{\partial x_k} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j \\
&= \frac{\partial}{\partial x_k} \left[ \sum_{i \neq k} \sum_{j \neq k} A_{ij} x_i x_j + \sum_{i \neq k} A_{ik} x_i x_k + \boxed{\sum_{j \neq k} A_{kj} x_k x_j} + A_{kk} x_k^2 \right] \\
&= \sum_{i \neq k} A_{ik} x_i + \boxed{\sum_{j \neq k} A_{kj} x_j} + 2 A_{kk} x_k
\end{aligned}
$$

# Exercise

$$
\begin{aligned}
\frac{\partial f(x)}{\partial x_k} &= \frac{\partial}{\partial x_k} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j \\
&= \frac{\partial}{\partial x_k} \left[ \sum_{i \neq k} \sum_{j \neq k} A_{ij} x_i x_j + \sum_{i \neq k} A_{ik} x_i x_k + \sum_{j \neq k} A_{kj} x_k x_j + \boxed{A_{kk} x_k^2} \right] \\
&= \sum_{i \neq k} A_{ik} x_i + \sum_{j \neq k} A_{kj} x_j + \boxed{2 A_{kk} x_k}
\end{aligned}
$$

# Exercise

$$
\begin{aligned}
\frac{\partial f(x)}{\partial x_k} &= \frac{\partial}{\partial x_k} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j \\
&= \frac{\partial}{\partial x_k} \left[ \sum_{i \neq k} \sum_{j \neq k} A_{ij} x_i x_j + \sum_{i \neq k} A_{ik} x_i x_k + \sum_{j \neq k} A_{kj} x_k x_j + A_{kk} x_k^2 \right] \\
&= \sum_{i \neq k} A_{ik} x_i + \sum_{j \neq k} A_{kj} x_j + 2 A_{kk} x_k
\end{aligned}
$$

# Exercise

$$
\begin{aligned}
\frac{\partial f(x)}{\partial x_k} &= \frac{\partial}{\partial x_k} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j \\
&= \frac{\partial}{\partial x_k} \left[ \sum_{i \neq k} \sum_{j \neq k} A_{ij} x_i x_j + \sum_{i \neq k} A_{ik} x_i x_k + \sum_{j \neq k} A_{kj} x_k x_j + A_{kk} x_k^2 \right] \\
&= \sum_{i \neq k} A_{ik} x_i + \sum_{j \neq k} A_{kj} x_j + 2 A_{kk} x_k \\
&= \sum_{i=1}^{n} A_{ik} x_i + \sum_{j=1}^{n} A_{kj} x_j = 2 \sum_{i=1}^{n} A_{ki} x_i,
\end{aligned}
$$

# Exercise

$$f(x) = x^T A x$$

$$f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j$$

$$\frac{\partial^2 f(x)}{\partial x_k \partial x_\ell} = \frac{\partial}{\partial x_k}\left[\frac{\partial f(x)}{\partial x_\ell}\right] = \frac{\partial}{\partial x_k}\left[\sum_{i=1}^{n} A_{\ell i} x_i\right]$$

# Exercise

$$f(x) = x^T A x$$

$$f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j$$

$$\frac{\partial^2 f(x)}{\partial x_k \partial x_\ell} = \frac{\partial}{\partial x_k} \left[ \frac{\partial f(x)}{\partial x_\ell} \right] = \frac{\partial}{\partial x_k} \left[ \sum_{i=1}^{n} A_{\ell i} x_i \right]$$

$$= 2A_{\ell k} = 2A_{k\ell}.$$

# Exercise

$$f(x) = x^T A x$$

$$f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j$$

$$\frac{\partial^2 f(x)}{\partial x_k \partial x_\ell} = \frac{\partial}{\partial x_k}\left[\frac{\partial f(x)}{\partial x_\ell}\right] = \frac{\partial}{\partial x_k}\left[\sum_{i=1}^{n} 2A_{\ell i} x_i\right]$$

$$= 2A_{\ell k} = 2A_{k\ell}.$$

$$\nabla_x^2 f(x) = 2A$$

# What we have learned

- Vectors and matrices
  - Basic Matrix Operations
  - Special Matrices
- Transformation Matrices
  - Homogeneous coordinates
  - Translation
- Matrix inverse
- Matrix rank
- Eigenvalues and Eigenvectors
- Matrix Calculate