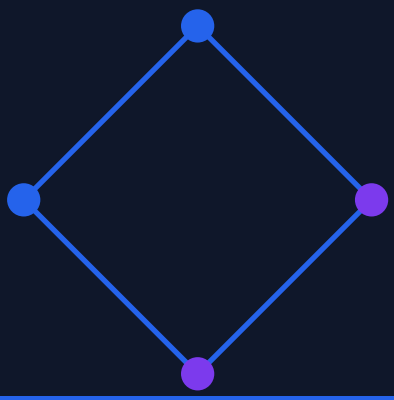
PMAI LAB

Mathematics • Python • Artificial Intelligence



LINEAR ALGEBRA FOR MACHINE LEARNING

Lecture 04

Matrices as Linear Transformations, Composition, Inverses, and Geometric Effects

Lecture focus Interpreting matrices as functions that transform data, combining transformations, reversing invertible mappings, and applying these ideas in graphics and machine learning.

Dr. Hira Benish

Associate Professor of Mathematics

Student Learning Handout | Includes NumPy practice and review exercises

1 Lecture Overview

A matrix is not merely a table of numbers. It represents a rule that transforms an input vector into an output vector. This viewpoint unifies coordinate changes, image transformations, feature engineering, neural-network layers, and many other computational operations.

This lecture develops matrix operations, linear transformations, geometric mappings, composition, inverse transformations, determinants, and affine transformations. It also explains the row-versus-column data convention used in Python.

Learning Outcomes

After completing this lecture, students should be able to:

- interpret an $m \times n$ matrix as a mapping from $\mathbb{R}^n$ to $\mathbb{R}^m$;
- verify whether a transformation is linear;
- describe scaling, reflection, rotation, and shear using matrices;
- compose transformations in the correct order;
- state when an inverse matrix exists and interpret its meaning;
- distinguish between linear and affine transformations;
- apply transformations to ML datasets stored by rows; and
- implement and verify transformations using NumPy.

1.1 Transformation Viewpoint

Object	Interpretation
Input vector x	Original observation or representation
Matrix A	Linear rule or learned weights
Output A**x**	Transformed representation
Matrix product BA	Composition: first A, then B
Inverse A^{-1}	Reversal of an invertible transformation
Bias b	Translation added after a linear mapping

2 Matrices as Mappings

For $A \in \mathbb{R}^{m\times n}$, multiplication defines a transformation

$$T : \mathbb{R}^n \to \mathbb{R}^m, \qquad T(\mathbf{x}) = A\mathbf{x}.$$

The input has n components and the output has m components.

2.1 Column Interpretation

If

$$A = \begin{bmatrix} | & & | \\ \mathbf{a}_1 & \cdots & \mathbf{a}_n \\ | & & | \end{bmatrix}, \qquad \mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix},$$

then

$$A\mathbf{x} = x_1\mathbf{a}_1 + \cdots + x_n\mathbf{a}_n.$$

The columns of A are the images of the standard basis vectors. Therefore, knowing where a transformation sends the basis completely determines the transformation.

2.2 Definition of Linearity

A transformation T is linear when, for all vectors $\mathbf{u}, \mathbf{v}$ and scalars c,

$$T(\mathbf{u}+\mathbf{v}) = T(\mathbf{u}) + T(\mathbf{v}), \qquad T(c\mathbf{u}) = cT(\mathbf{u}).$$

Every matrix transformation $T(\mathbf{x}) = A\mathbf{x}$ is linear. A linear transformation must satisfy $T(\mathbf{0}) = \mathbf{0}$.

2.3 Worked Example

Let

$$A = \begin{bmatrix} 2 & 1 \\ 0 & 3 \end{bmatrix}, \qquad \mathbf{x} = \begin{bmatrix} 4 \\ -1 \end{bmatrix}.$$

Then

$$A\mathbf{x} = \begin{bmatrix} 2(4) + 1(-1) \\ 0(4) + 3(-1) \end{bmatrix} = \begin{bmatrix} 7 \\ -3 \end{bmatrix}.$$

Machine-Learning Connection

A learned feature layer forms new features as linear combinations of the original features. Each row of the weight matrix defines one output feature, while each column shows how one input feature contributes across outputs.

3 Geometric Transformations in Two Dimensions

Matrices can change the length, direction, orientation, and area of geometric objects.

3.1 Common Transformation Matrices

Transformation	Matrix	Effect
Scaling	$\begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$	Stretches or compresses each axis
Reflection in x-axis	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	Reverses the vertical coordinate
Rotation by θ	$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$	Rotates counterclockwise about the origin
Horizontal shear	$\begin{bmatrix} 1 & k \\ 0 & 1 \end{bmatrix}$	Shifts x according to y
Projection onto x-axis	$\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$	Removes the vertical component

3.2 Rotation Example

A counterclockwise rotation through 90° uses

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

Thus

$$R \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 2 \end{bmatrix}.$$

3.3 Determinant as Geometric Scaling

For a square transformation matrix, $|\det(A)|$ is the factor by which areas in $\mathbb{R}^2$ or volumes in $\mathbb{R}^3$ are scaled.

- $|\det(A)| > 1$: expansion;
- $0 < |\det(A)| < 1$: contraction;
- $\det(A) < 0$: orientation is reversed; and
- $\det(A) = 0$: dimension collapses and information is lost.

> A projection matrix can map a plane onto a line. Its determinant is zero because no inverse can recover the discarded direction.

4 Composition of Transformations

If $T_A(\mathbf{x}) = A\mathbf{x}$ and $T_B(\mathbf{x}) = B\mathbf{x}$, then applying A first and B second gives

$$T_B(T_A(\mathbf{x})) = B(A\mathbf{x}) = (BA)\mathbf{x}.$$

The matrix of the combined transformation is BA.

Order Matters

Matrix multiplication is generally not commutative:

$$BA \neq AB.$$

The rightmost matrix acts first. Therefore, "scale then rotate" is represented by RS, whereas "rotate then scale" is represented by SR.

4.1 Worked Composition

Let

$$S = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, \qquad R = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}.$$

Then

$$RS = \begin{bmatrix} 0 & -1 \\ 2 & 0 \end{bmatrix}, \qquad SR = \begin{bmatrix} 0 & -2 \\ 1 & 0 \end{bmatrix}.$$

For $\mathbf{x} = [1, 1]^T$,

$$(RS)\mathbf{x} = \begin{bmatrix} -1 \\ 2 \end{bmatrix}, \qquad (SR)\mathbf{x} = \begin{bmatrix} -2 \\ 1 \end{bmatrix}.$$

The different outputs confirm that the order changes the result.

4.2 Identity Transformation

The identity matrix I satisfies

$$I\mathbf{x} = \mathbf{x}, \qquad AI = IA = A.$$

It is the transformation that leaves every vector unchanged.

Machine-Learning Connection

A deep neural network composes many transformations. Without activation functions, a sequence of linear layers collapses to one linear matrix product. Nonlinear activations are what allow deep networks to represent nonlinear relationships.

5 Inverse Transformations

For a square matrix A, an inverse matrix A^{-1} satisfies

$$A^{-1}A = AA^{-1} = I.$$

If $\mathbf{y} = A\mathbf{x}$, then

$$\mathbf{x} = A^{-1}\mathbf{y}.$$

The inverse reverses the effect of the original transformation.

5.1 Equivalent Conditions for Invertibility

For $A \in \mathbb{R}^{n\times n}$, the following statements are equivalent:

- A^{-1} exists;
- $\det(A) \neq 0$;
- $\operatorname{rank}(A) = n$;
- the columns of A form a basis for $\mathbb{R}^n$;
- $A\mathbf{x} = \mathbf{0}$ has only the zero solution; and
- $A\mathbf{x} = \mathbf{b}$ has a unique solution for every $\mathbf{b}$.

5.2 Two-by-Two Formula

For

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \qquad ad - bc \neq 0,$$

$$A^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

5.3 Example

For $A = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$, $\det(A) = 1$ and

$$A^{-1} = \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix}.$$

> In numerical work, do not normally calculate A^{-1} merely to solve $A\mathbf{x} = \mathbf{b}$. Use `np.linalg.solve(A, b)` because it is more efficient and usually more stable.

Machine-Learning Connection

Standardization can be reversed when the original scale parameters are retained. By contrast, dimensionality-reducing transformations are generally not perfectly invertible because some information is intentionally discarded.

6 Affine Transformations

An **affine transformation** has the form

$$T(\mathbf{x}) = A\mathbf{x} + \mathbf{b}.$$

It combines a linear transformation with a translation. If $\mathbf{b} \neq \mathbf{0}$, then $T(\mathbf{0}) = \mathbf{b}$, so the transformation is not linear.

6.1 Neural-Layer Interpretation

A dense neural layer calculates

$$\mathbf{z} = W\mathbf{x} + \mathbf{b}.$$

This is an affine transformation. An activation function is applied afterwards:

$$\mathbf{a} = \sigma(\mathbf{z}).$$

6.2 Translation with Homogeneous Coordinates

A translation in two dimensions cannot be represented by a 2×2 matrix alone. Introduce a homogeneous coordinate:

$$\widetilde{\mathbf{x}} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

Translation by (t_x, t_y) becomes

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

6.3 Standardization as an Affine Map

For one feature,

$$z = \frac{x - \mu}{s} = \frac{1}{s}x - \frac{\mu}{s},$$

where μ is the mean and s is the standard deviation. This is an affine transformation of the feature.

Linear versus Affine

Linear maps preserve the origin and linear combinations. Affine maps preserve lines and parallelism but may move the origin. Many operations informally called “linear layers” in ML software are mathematically affine because they include a bias.

Machine-Learning Connection

Computer graphics uses homogeneous coordinates to combine scaling, rotation, and translation into one matrix pipeline. The same composition principle is used in image preprocessing and data augmentation.

7 Python and NumPy Practice

7.1 Applying Basic Transformations

```python
import numpy as np

x = np.array([2.0, 1.0])

scale = np.array([[2.0, 0.0],
                  [0.0, 0.5]])

theta = np.deg2rad(90)
rotate = np.array([
    [np.cos(theta), -np.sin(theta)],
    [np.sin(theta),  np.cos(theta)]
])

print("Scaled:", scale @ x)
print("Rotated:", rotate @ x)
```

7.2 Verifying Composition Order

```python
scale_then_rotate = rotate @ scale
rotate_then_scale = scale @ rotate

y1 = scale_then_rotate @ x
y2 = rotate_then_scale @ x

print("Scale then rotate:", y1)
print("Rotate then scale:", y2)
print("Same result:", np.allclose(y1, y2))
```

7.3 Solving and Recovering an Input

```python
A = np.array([[2.0, 1.0],
              [1.0, 1.0]])

original = np.array([3.0, -1.0])
transformed = A @ original
recovered = np.linalg.solve(A, transformed)

print("Transformed:", transformed)
print("Recovered:", recovered)
print("Recovery correct:",
      np.allclose(original, recovered))
```

> Values such as $\cos(90^\circ)$ may appear as a tiny floating-point number rather than exact zero. Use `np.allclose` when checking numerical equality.

8 ML Case Study: Transforming a Dataset

Mathematical formulas often use column vectors, while ML datasets usually store observations as rows.

If $A\mathbf{x}$ transforms one column vector, a row-based dataset X is transformed using

$$X_{\text{new}} = XA^T.$$

8.1 Rotating Two-Dimensional Data

```
X = np.array([
    [1.0, 0.0],
    [2.0, 1.0],
    [0.0, 2.0],
    [-1.0, 1.0]
])

angle = np.deg2rad(30)
A = np.array([
    [np.cos(angle), -np.sin(angle)],
    [np.sin(angle),  np.cos(angle)]
])

X_rotated = X @ A.T
print(X_rotated)
```

8.2 Affine Feature Layer

```
W = np.array([
    [0.8, -0.3],
    [0.2,  0.6],
    [-0.5, 0.9]
])
bias = np.array([0.1, -0.2, 0.3])

# Each row of X is one observation.
Z = X @ W.T + bias
relu_output = np.maximum(0, Z)

print("Affine output shape:", Z.shape)
print("After ReLU:\n", relu_output)
```

Since X has shape 4×2 and W has shape 3×2, the output XW^T has shape 4×3: four observations and three learned features.

Machine-Learning Connection

Shape reasoning is a professional ML skill. It helps detect incorrect matrix order, misplaced transposes, and incompatible feature dimensions before training a model.

9 Review and Independent Practice

Practice Tasks

1. For $A = \begin{bmatrix} 1 & 2 \\ 3 & 1 \end{bmatrix}$ and $\mathbf{x} = [2, -1]^T$, calculate $A\mathbf{x}$ and interpret the output dimension.
2. Construct matrices for reflection in the y-axis, uniform scaling by 3, and rotation by $180°$.
3. Apply a non-uniform scaling and a $90°$ rotation in both orders to $[1, 2]^T$. Compare the results.
4. Determine whether $A = \begin{bmatrix} 2 & 4 \\ 1 & 2 \end{bmatrix}$ has an inverse. Explain geometrically.
5. Find the inverse of $\begin{bmatrix} 3 & 1 \\ 2 & 1 \end{bmatrix}$ and verify the product with NumPy.
6. Explain why $T(\mathbf{x}) = A\mathbf{x} + \mathbf{b}$ is not linear when $\mathbf{b} \neq \mathbf{0}$.
7. Create a 5×2 dataset and transform it to four features using a 4×2 weight matrix and a bias vector.

9.1 Key Takeaways

- An $m \times n$ matrix maps vectors from $\mathbb{R}^n$ to $\mathbb{R}^m$.
- Matrix columns show where the standard basis vectors are sent.
- Scaling, rotation, reflection, shear, and projection are matrix transformations.
- Composition is matrix multiplication, and the rightmost transformation acts first.
- An inverse exists exactly when a square transformation loses no direction.
- An affine map adds translation or bias to a linear transformation.
- Row-based ML datasets use XA^T when $A\mathbf{x}$ is the column-vector formula.

9.2 Key Terms

Linear transformation	Matrix mapping	Composition
Identity matrix	Inverse matrix	Determinant
Rotation	Scaling	Shear
Projection	Affine transformation	Homogeneous coordinate

Next lecture: Inner products, orthogonality, vector projections, and geometric similarity.