Introduction to Linear Algebra in Python

Part II: Matrices

BILD 62

Reminders

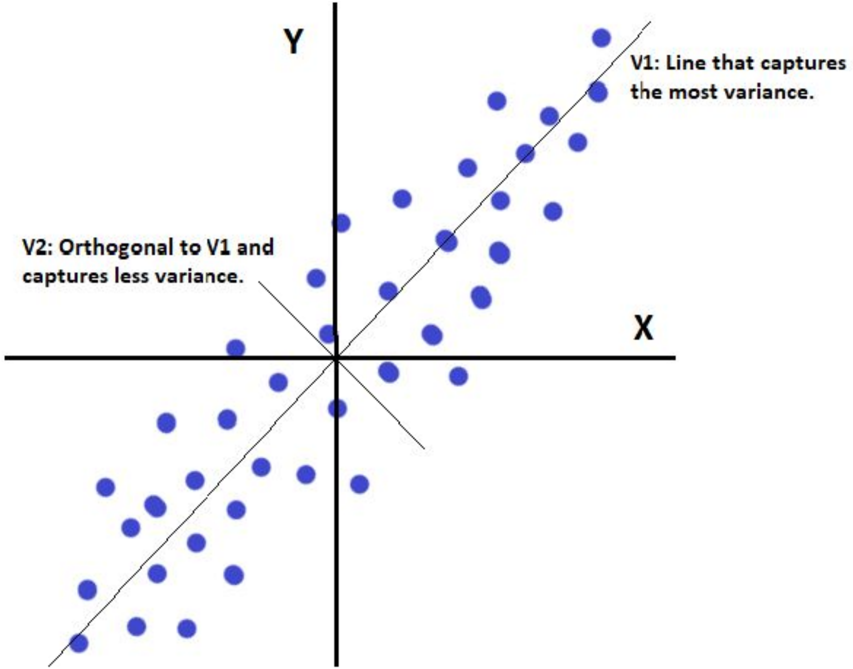
Last time we talked about:

- Different ways to conceptualize a vector, either as a list of numbers of as an arrow in space
- Different operations you can do with vectors (addition, subtraction, scalar multiplication, dot products)

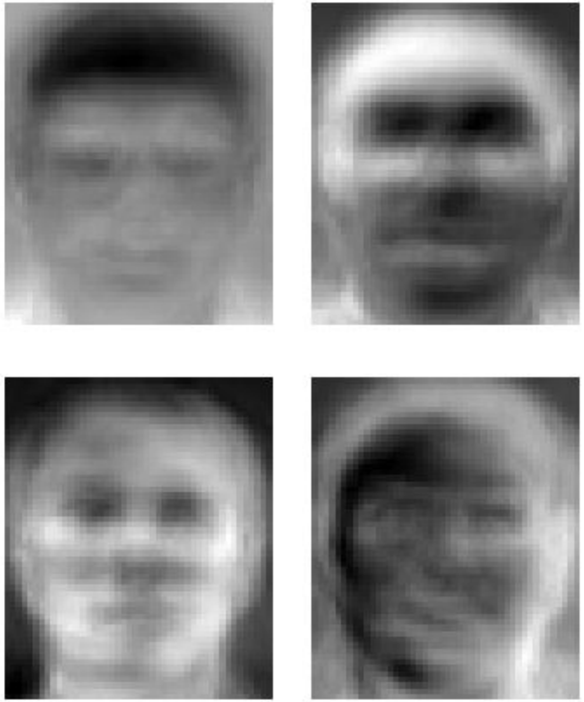
By the end of this lecture, you will be able to:

- Construct and multiply matrices in Python (and by hand)
- Create and manipulate special cases of matrices
- Explain matrices as a linear transformation and relate matrix properties to properties of that linear transformation
- Define what eigenvalues/eigenvectors are and determine them using Python

What's the fuss about **eigenvectors**?



Principal Component Analysis ([video]())



[Eigenfaces]()

A brief introduction to matrices

2 x 3 matrix

Matrix A has 2 rows and 3 columns, can be indexed as $\mathbf{A}_{i,j}$, where i is the row number and j is the column number.

For example, $\mathbf{A}_{1,2} = \mathbf{4}$ and $\mathbf{A}_{2,3} = \mathbf{3}$.

$$A = \begin{pmatrix} 2 & 4 & 8 \\ 1 & 7 & 3 \end{pmatrix}$$

Note: Be mindful of indexing differences when translating formulas into Python code!

Special matrices

Square matrices are those with the same number of row and column. m = n. For example,

$$A = \begin{pmatrix} 2 & 4 \\ 1 & 5 \end{pmatrix}, \; b = \begin{pmatrix} 2 & 4 & 8 \\ 1 & 7 & 3 \\ 2 & 5 & 6 \end{pmatrix}$$

Diagonal matrices are square matrices with only the values along the main diagonal are non-zero. For example,

$$C = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 7 & 0 \\ 0 & 0 & 6 \end{pmatrix}$$

Identity matrices are diagonal matrices where all the non-zero values are 1. For example,

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Slide: Jing Wang

Matrix transposition

Transposition flips rows and columns — each row of the original matrix becomes the corresponding column of the new matrix

$$A = \begin{pmatrix} 2 & 4 & 8 \\ 1 & 7 & 3 \end{pmatrix} \quad \Longleftrightarrow \quad A^T = \begin{pmatrix} 2 & 1 \\ 4 & 7 \\ 8 & 3 \end{pmatrix}$$

We can move between these with **transposition**

The example for today’s notebook will focus on the visual system — specifically connections from the **retina** to the **LGN**, a visual nucleus in the **thalamus**

The retina *projects* to the LGN

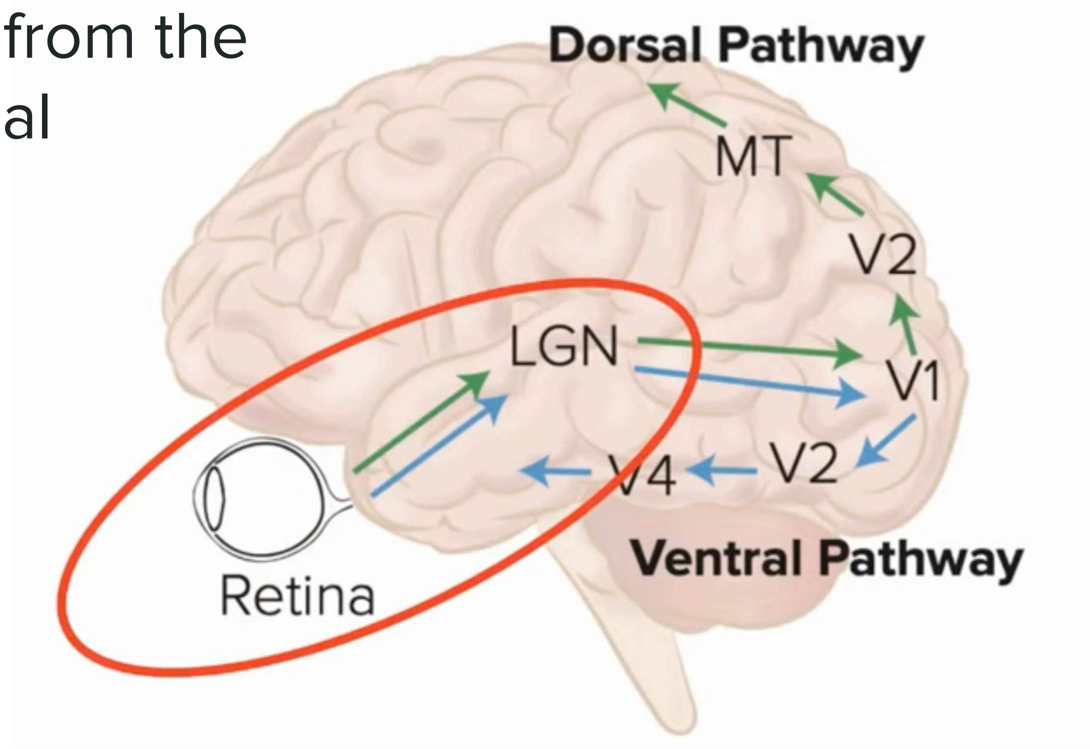
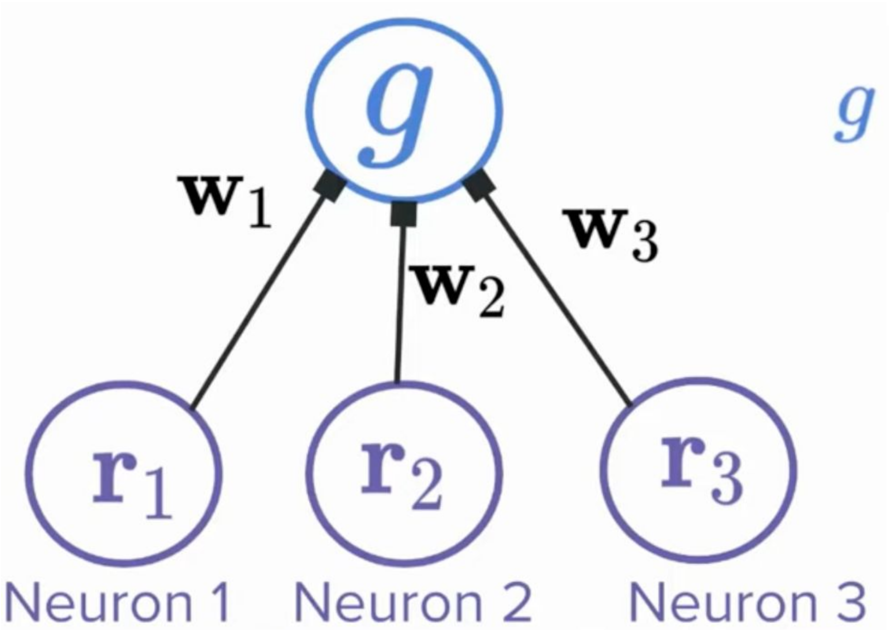


Image: Neuromatch Academy

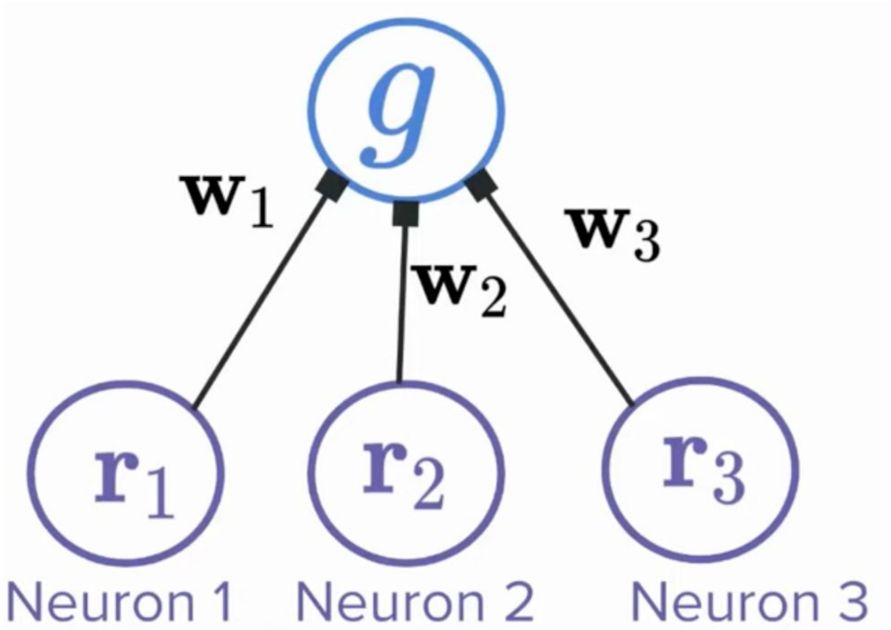
We can describe **LGN** activity as an equation, summing the weights ($\mathbf{w}$) * activities ($\mathbf{r}$) of each **retina** neuron.



$$g = w_1 r_1 + w_2 r_2 + w_3 r_3$$

We can call this a **weighted sum**

We can simplify this by using vectors, and computing the **dot product**.



$$\mathbf{w} = \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \mathbf{w}_3 \end{bmatrix} \qquad \mathbf{r} = \begin{bmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \\ \mathbf{r}_3 \end{bmatrix}$$

$$\begin{aligned} g &= \mathbf{w} \cdot \mathbf{r} \\ &= \mathbf{w}_1 \mathbf{r}_1 + \mathbf{w}_2 \mathbf{r}_2 + \mathbf{w}_3 \mathbf{r}_3 \end{aligned}$$

Building a model of neural connections using linear equations

Let’s try this in the notebook.

Thinking about the computations as linear transformations of matrices

Additional resources

https://www.youtube.com/watch?v=Ene_TYyTdNM — modeling neural connections as vectors and dot product review

http://matrixmultiplication.xyz/ — awesome visualization of matrix multiplication!

Essence of linear algebra - YouTube