

# Just Enough Spectral Theory for Machine Learning

Ng Yen Kaow

# Notations (Important)

- A vector is by default a column
  - For vectors  $x$  and  $y$ , their inner (or dot) product,  $\langle x, y \rangle = x^T y$ 
    - $\langle x + z, y \rangle = \langle x, y \rangle + \langle z, y \rangle = x^T y + z^T y$
  - Beware: some texts use row vectors and  $\langle x, y \rangle = xy^T$
- For a matrix an example is a row
  - An example (or datapoint) is a row  $x_i$  while each feature is a columns
    - Features are like fixed columns in a spreadsheet
  - For matrices  $X$  and  $Y$ ,  $\langle X, Y \rangle = XY^T$  or  $\sum_i (x_i y_i^T)$
  - Beware: some texts use column for examples and let  $\langle X, Y \rangle = X^T Y$
- So it's  $x^T x$ ,  $x^T M x$ , but  $XX^T$  and  $Q\Lambda Q^T$

# Outer product

- The **outer product** of two vectors  $x$  and  $y$  is a matrix  $M$  where the  $M_{ij} = x_i y_j$

e.g.  $\begin{pmatrix} a \\ b \end{pmatrix} \begin{pmatrix} c & d \end{pmatrix} = \begin{pmatrix} ac & ad \\ bc & bd \end{pmatrix}$

- The outer product (or Kronecker product) of two matrices is a **tensor**

- We don't deal with tensors yet

- Common uses of outer products

- Denote pairwise dot product matrix,

$$xx^T = \begin{pmatrix} x_1 x_1 & x_1 x_2 & \dots \\ x_2 x_1 & x_2 x_2 & \dots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

- Denote matrix of all ones,  $\mathbf{1}\mathbf{1}^T = \begin{bmatrix} 1 & \dots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \dots & 1 \end{bmatrix}$

# More notations

## □ Conventions

- $x_i$  from a matrix is by default a row vector
- $x_i$  from a vector is a scalar
- $x_{ij}$  from a matrix is a scalar
- $x, u_i$  (all other vectors) are by default column vectors

## □ Common expansions

$$xy^T = \sum_i x_i y_i$$

$$(XY)_{ij} = \sum_k x_{ik} y_{kj}$$

$$(x^T y)_{ij} = x_i y_j$$

$$(XY^T)_{ij} = x_i y_j^T = \sum_k x_{ik} y_{jk}$$

$$x^T M y = \sum_{ij} m_{ij} x_i y_j$$

$$(X^T Y)_{ij} = \sum_k x_{ki} y_{kj}$$

$$X^T X = \sum_i x_i^T x_i \quad (\text{used in kernel PCA})$$

# Python call for inner product

- Inner products are performed with `np. dot ()`
  - When called on two arrays, the arrays are **automatically** oriented to perform inner product
    - Note that `[[ 1 ], [ 1 ]]` is a  $1 \times 2$  matrix
  - When called on an array `x` and a matrix `X`, the array is **automatically** read as a row for `np. dot (x, X)`, and column for `np. dot (X, x)` to perform inner product
  - When called on two matrices, make sure that the matrices are oriented correctly, or you will get  $X^T X$  when you want  $XX^T$
  - Impossible to get outer product with `np. dot ()`
- If you write `x*y` or `X*Y`, what you get is an element-wise multiplication

# Eigenvectors and eigenvalues

- Only concerned with **square** matrices
  - Most matrices we consider are furthermore **symmetric** (and of only **real** values)
- A **eigenvector** for a square matrix  $M$  is vector  $u$  where  $Mu = \lambda u$ 
  - $u$  is **invariant** under transformation  $M$
  - The scaling factor  $\lambda$  is a **eigenvalue**
  - Use  $u$  to denote a column vector even when multiple  $u_i$  are collected into a matrix  $U = \begin{bmatrix} u_1 & \dots & u_k \end{bmatrix}$

# $Mu = \lambda u$ is a system of equations

- An equation such as  $Mu = \lambda u$  actually states  $n$  linear equations, namely  $\forall i, \sum_j m_{ij}u_j = \lambda u_i$

- For example

$$\begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \lambda \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$$

states the two equations

$$m_{11}u_1 + m_{12}u_2 = \lambda u_1$$

$$m_{21}u_1 + m_{22}u_2 = \lambda u_2$$

- This is important when manipulating equation by multiplying with other matrix/vector
  - For example when  $Mu = \lambda u$  is multiplied from the left by  $u^T$ , the resultant  $u^T Mu = \lambda u^T u$  becomes only one equation, that is,  $\sum_{ij} u_i m_{ij} u_j = \lambda \sum_{ij} u_i u_j$

# Eigendecomposition

- A eigendecomposition of matrix  $M$  is

$$M = Q\Lambda Q^{-1}$$

where  $\Lambda$  is **diagonal**, and  $Q$  contains (not necessarily orthogonal) **eigenvectors** of  $M$

- Any **normal**  $M$  can be eigendecomposed
- **The set of eigenvalues for  $M$  is unique**
- **There can be different eigenvectors of the same eigenvalue (hence not unique)**
  - **For **real symmetric**  $M$ , eigenvectors that correspond to distinct eigenvalues are orthogonal**
- For an orthogonal matrix  $Q$ ,  $Q^{-1} = Q^T$
- **Only consider real symmetric  $M \Rightarrow M = Q\Lambda Q^T$**



# Eigenspace

- The **eigenspace** of a matrix  $M$  is the set of all the vectors  $u$  that fulfills  $Mu = \lambda u$ 
  - The **rank** of  $M$  is its number of non-zero  $\lambda$
- A **eigenbasis** of a  $n \times n$  matrix  $M$  is a set of  $n$  **orthogonal** eigenvectors of  $M$  (including those with zero eigenvalues)
  - Any datapoint  $x_i$  in  $M$  can be written as a linear combination of the eigenbasis,  $x_i = \sum_j \langle x_i, u_j \rangle u_j$
  - Any eigenvector  $u_i$  for  $M$  can be written as a linear combination of the datapoints  $x_i$ , by solving the system of equations  $x_i = \sum_j \langle x_i, u_j \rangle u_j$

# Rayleigh Quotient

- Consider an  $n \times n$  real symmetric  $M$
- $M = Q\Lambda Q^T$ , where  $\Lambda$  is diagonal, and  $Q$  is the eigenbasis of  $M$

- Denote the eigenvalues  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$ .

- Then, for all unit vector  $u$

Min-max  
Theorem

$$\max_{\|u\|=1} \frac{u^T M u}{u^T u} = \lambda_1$$

Similarly,  $\lambda_n$  is the  
minimum of the  
Rayleigh Quotient

- And for all orthogonal matrix  $P$  and  $k \leq n$

Minimax  
Principle

$$\max_{P \in \mathbb{R}^{k \times n}, P^T P = I} \text{tr}(P^T M P) = \lambda_1 + \dots + \lambda_k$$

Similarly,  $\min_{P \in \mathbb{R}^{k \times n}, P^T P = I} \text{tr}(P^T M P) = \lambda_{n-k+1} + \dots + \lambda_n$

# Eigendecomposition applications

- Matrix inverse
- Matrix approximation
- Matrix factorization
  - Multidimensional Scaling
- Minimization or maximization through the Rayleigh Quotient
  - PCA
    - Max of covariance matrix
  - Spectral clustering
    - Min of graph Laplacian

# Singular Value Decomposition

- Any matrix can be singular value decomposed
- $M = U\Sigma V^*$ 
  - $M$  is  $m \times n$  matrix
  - $U$  is an  $m \times m$  unitary (orthogonal) matrix
  - $\Sigma$  is an  $m \times n$  diagonal matrix
  - $V$  is an  $n \times n$  unitary matrix

□ For a real  $M$ ,  $V^* = V^T$  (and  $U = U^T$ ) hence  
$$M = U\Sigma V^T$$

# SVD applications

- Solving linear equations
- Linear regression
- Pseudoinverse
- Kabsch algorithm
- Matrix approximation
- As a eigendecomposition (see next slide)

# SVD and eigendecomposition

- SVD is a eigendecomposition but not of  $M$ 
  - Given an SVD of  $M = U\Sigma V^T$
  - Then, clearly
    - $M^T M = V\Sigma^T U^T U\Sigma V^T = V(\Sigma^T \Sigma)V^T$
    - $MM^T = U\Sigma V^T V\Sigma^T U^T = U(\Sigma^T \Sigma)U^T$
  - Hence  $V$  is the eigenbasis of  $M^T M$  and  $U$  is the eigenbasis of  $MM^T$  respectively
  - That is,  $U$  and  $V$  are **eigenbases of the squared matrices of  $M$** 
    - However the eigenbasis of  $M^T M$  and  $MM^T$  are in general not the eigenbasis of  $M$