ESQC 2026

Mathematical Methods Lecture 2

By Simen Kvaal

Today's agenda:

- More on matrices
- Linear systems of equations
- Matrix decompoisitions

Brief recap

The matrix-vector product

- A matrix and a vector can be **multiplied together**

$$A \in M(m, n, \mathbb{F}) \qquad \mathbf{x} \in \mathbb{F}^n \qquad A\mathbf{x} \in \mathbb{F}^m$$

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1} & A_{m2} & \cdots & A_{mn} \end{bmatrix} \qquad (A\mathbf{x})_i = \sum_{j=1}^{n} A_{ij} x_j$$

Matrix-matrix product

$$AB = C \qquad\qquad C_{ij} = \sum_k A_{ik} B_{kj}$$

$$\begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1} & A_{m2} & \cdots & A_{mn} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} & \cdots & B_{1o} \\ B_{21} & B_{22} & \cdots & B_{2o} \\ \vdots & \vdots & \ddots & \vdots \\ B_{n1} & B_{n2} & \cdots & B_{no} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1o} \\ C_{21} & C_{22} & \cdots & C_{2o} \\ \vdots & \vdots & \ddots & \vdots \\ C_{m1} & C_{m2} & \cdots & C_{mo} \end{bmatrix}$$

Note 1: $C_{ij} = (\text{col } i \text{ of } A) \cdot (\text{row } j \text{ of } B)$

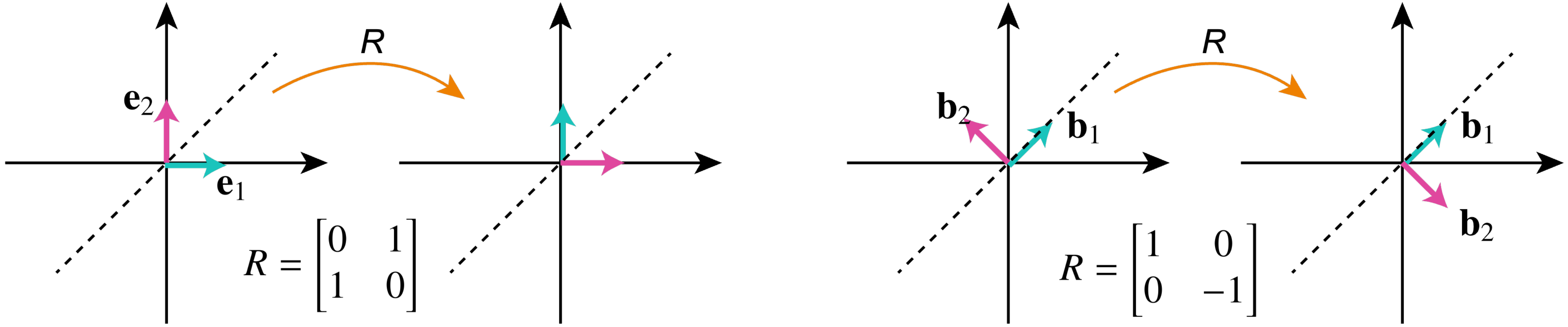
Note 2: $A \times$ a column of B = a column of C

Matrices = linear transformations

- The matrix-vector product **is linear**
- Therefore it defines a linear transformation (from n to m dimensions)
- Conversely, any linear transformation can be written as a matrix-vector product
- **A powerful result!**

The matrix of a linear transformation

- Recall: n-dimensional space has **infinitely many bases**
- The definition of a linear transformation does **not rely on a basis**
- Once a basis is **chosen** the linear transformation beomes a **matrix**



The matrix zoo

The matrix zoo

- Matrices contain a lot of information!
- Can we tame that information?
- Usually, we are interested in special problems:
  - **Define important classes of matrices with common properties**
- For a given matrix, try to say as much as possible about it
  - **Study decompositions of matrices in terms of simpler matrices**

- We will now look at some **classes of matrices**

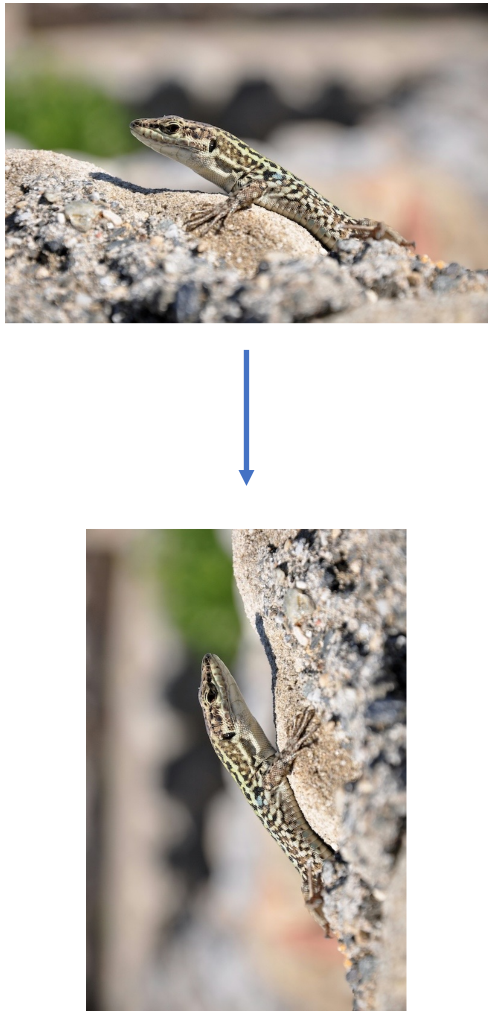
Important matrix operations

- Sometimes we need rows and columns to switch place
- **Transpose:**

$$(A^T)_{ij} = A_{ji} \qquad \begin{bmatrix} 0 & 1 \\ \mathrm{i} & 2 \\ 0 & -1 \end{bmatrix}^T = \begin{bmatrix} 0 & \mathrm{i} & 0 \\ 1 & 2 & -1 \end{bmatrix}$$

- **Hermitian adjoint:**

$$(A^H)_{ij} = \overline{A_{ji}} \qquad \begin{bmatrix} 0 & 1 \\ \mathrm{i} & 2 \\ 0 & -1 \end{bmatrix}^H = \begin{bmatrix} 0 & -\mathrm{i} & 0 \\ 1 & 2 & -1 \end{bmatrix}$$



Why this operation?

$$\langle \mathbf{x}, \mathbf{y} \rangle = x_1 y_1 + \ldots + x_n y_n = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \mathbf{x}^T \mathbf{y}$$

$$\langle \mathbf{x}, A\mathbf{y} \rangle = \langle A^T \mathbf{x}, \mathbf{y} \rangle$$

Real case

$$\langle \mathbf{x}, \mathbf{y} \rangle = x_1^* y_1 + \ldots + x_n^* y_n = \begin{bmatrix} x_1^* & \cdots & x_n^* \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \mathbf{x}^H \mathbf{y}$$

$$\langle \mathbf{x}, A\mathbf{y} \rangle = \langle A^H \mathbf{x}, \mathbf{y} \rangle$$

Complex case

Class 1: Symmetric matrices

- Square matrices that are their own **transpose**

$$A = A^T \quad \Longleftrightarrow \quad A_{ij} = A_{ji}$$

$$A = \begin{bmatrix} 0 & 2 \\ 2 & -1 \end{bmatrix} \quad \text{is symmetric}$$

$$B = \begin{bmatrix} 0 & 2 \\ -2 & -1 \end{bmatrix} \quad \text{is not symmetric}$$

Class 2: Hermitian matrices (complex)

- Square matrices that are their own **conjugate transpose**

$$A = A^H \quad \iff \quad A_{ij} = A^*_{ji}$$

$$A = \begin{bmatrix} 0 & 2 \\ 2 & -1 \end{bmatrix}$$ is Hermitian and symmetric

$$B = \begin{bmatrix} 0 & 2i \\ -2i & -1 \end{bmatrix}$$ is Hermitian but not symmetric

Hermitian matrices can be «diagonalized»

- Let A be Hermitian (real case: symmetric)
- Consider the eigenvalue problem:

$$A\mathbf{x}_k = \lambda_k \mathbf{x}_k$$

- An **orthonormal basis** of eigenvectors exist
- A becomes very simple in this basis:

$$\mathbf{y} = \sum_i y_i \mathbf{x}_i \quad A\mathbf{y} = \sum_i \lambda_i \, y_i \mathbf{x}_i$$

- **Demo:** `lecture-2-diagonalization.py`

The identity matrix is special

- We need a small aside ...
- The **identity matrix** defines the «do nothing»-transformation:

$$\mathbb{1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Class 3: Unitary/orthogonal matrices

- Char. 1: Matrices that satisfy:

$$U^H U = I$$

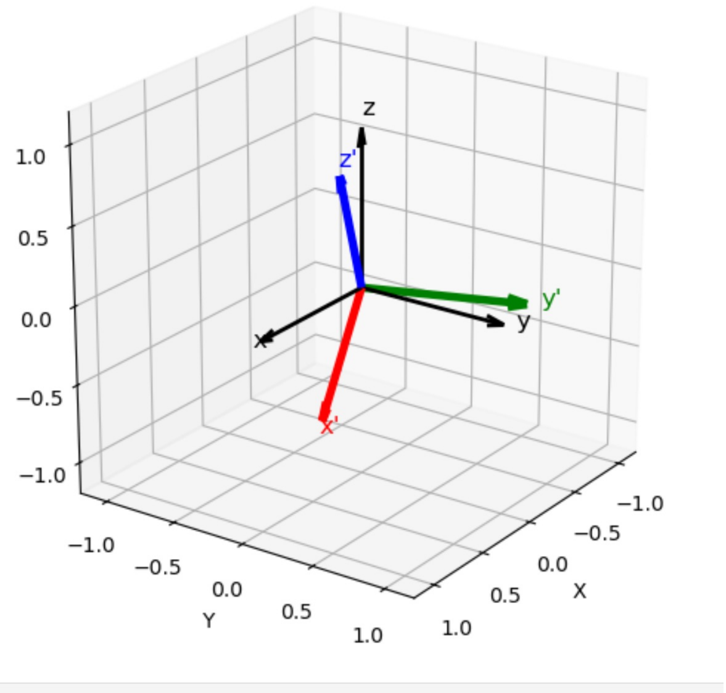
- Char. 2: Columns form **orthonormal basis**:

$$U = [\mathbf{u}_1 \ \mathbf{u}_2 \ \mathbf{u}_3 \cdots \mathbf{u}_n] = \begin{bmatrix} U_{11} & \cdots & U_{1n} \\ \vdots & \ddots & \vdots \\ U_{n1} & \cdots & U_{nn} \end{bmatrix}$$

- Char. 3: Matrices that **preserve orthogonality of vectors**
- Real case: «orthogonal», complex case: «unitary»

Demo

- Rotation in 3D
- `lecture-1-interactive-rotation.ipynb`



Linear systems of equations

- Given **b,** determine **x** such that:

$$A\mathbf{x} = \mathbf{b}$$

- Linear systems are **everywhere** in science
- Example: Balancing reactions

$$a\,\mathrm{C_2H_5OH} + b\,\mathrm{O_2} \rightarrow c\,\mathrm{CO_2} + d\,\mathrm{H_2O}$$

$$\begin{aligned} 2a - c &= 0, \\ 6a - 2d &= 0, \\ a + 2b - 2c - d &= 0. \end{aligned}$$

$$\begin{pmatrix} 2 & 0 & -1 & 0 \\ 6 & 0 & 0 & -2 \\ 1 & 2 & -2 & -1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$$(a, b, c, d) = (1, 3, 2, 3)$$

Gaussian elimination

- A method for solving linear systems
- Read about it in the lecture notes!
- Or watch some high-quality videos, e.g. https://www.youtube.com/watch?v=2GKESu5atVQ (MyWhyU)
- Of course, there is
  - `x = np.linalg.solve(A, b)`

Class 4: Invertible matrices

- If A is such that solution is unique $\Rightarrow$ existence of **inverse matrix**

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

$$AA^{-1} = A^{-1}A = \mathbb{1}$$

- **Example**: Inverse of plane rotation matrix

$$\begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}^{-1} = \begin{bmatrix} \cos(-\theta) & \sin(-\theta) \\ -\sin(-\theta) & \cos(-\theta) \end{bmatrix}$$

Inverse given
by opposite
rotation

- Non-existence of inverse: A is **singular** (e.g., balancing equation)

Matrix decompositions

Simplifying matrices with simpler matrices

Hermitian *A*: spectral decomposition

- We already saw that for Hermitian A,

$$A\mathbf{x}_k = \lambda_k \mathbf{x}_k$$

  had an orthonormal basis of eigenvectors as solution

- Then:

$$A = X\Lambda X^H$$

Unitary

Diagonal

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix} \qquad X = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \\ | & | & \cdots & | \end{bmatrix}$$

- **Demo:** Back to diagonalization notebook

Many matrices can be «diagonalized»

- It may be so that

$$A\mathbf{x}_k = \lambda_k \mathbf{x}_k$$

with a **basis of eigenvectors** – but possibly non-orthonormal

- Then:

$$A = X\Lambda X^{-1}$$

Diagonal

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix}$$

$$X = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \\ | & | & \cdots & | \end{bmatrix}$$

Invertible

- **Eigendecomposition**

Singular value decomposition (SVD):

- For **any matrix, even non-square,** we have the SVD:
  - Exists unitary matrices U, V
  - and diagonal Σ with positive entries

$$A = U\Sigma V^H$$

- Interpretation:

$$A\mathbf{x} = U\Sigma \underbrace{V^H\mathbf{x}}_{\text{1. Express } \mathbf{x} \text{ in the basis defined by V}}$$

2. Stretch the components

3. Tranform to new basis given by U

SVD continued

- **Demo:** `lecture-2-svd-of-image.ipynb`
- In which we view the parrot as a matrix – not as a vector ...

- Relevance in quantum chemistry:
  - Screening of electron repulsion integrals
  - Orthogonalization of orbitals
  - ...

LU decomposition

In general very expensive!

- We want to solve a **linear system:** find $\mathbf{x}$ such that: $A\mathbf{x} = \mathbf{b}$
- Under a mild technical condition on A there is a factorization

Lower triangular

$$A = LU$$

Upper triangular

$$L = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \qquad U = \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}$$

- This makes **repeated solves with different b super fast!**
- **Gaussian elimination** written as a matrix decoposition

Cholesky decomposition

- In many applications A satisfies:
  - **Hermitian:** $A = A^H$
  - **positive definite:** $\langle \mathbf{x}, A\mathbf{x} \rangle > 0$ for all nonzero $\mathbf{x}$

- Then LU simplifies: $A = LL^\dagger$

Cholesky decomposition

An application of Cholesky

- The **electron repulsion integral tensor** is costly to store and compute

N_{AO}^4 elements

$$(\mu\nu|\lambda\sigma) = \iint \frac{\chi_\mu(\mathbf{r}_1)\chi_\nu(\mathbf{r}_1)\chi_\lambda(\mathbf{r}_2)\chi_\sigma(\mathbf{r}_2)}{r_{12}}\, d\mathbf{r}_1\, d\mathbf{r}_2.$$

- View the tensor as a **matrix,** which is Hermitian and positive definite:

$$V_{IJ} = (I|J), \quad I = (\mu\nu), \quad J = (\lambda\sigma)$$

- V can be well approximated by an **incomplete Cholesky factorization!**

~ N_{AO}^2 elements

- **Demo:** `lecture-2-eri_svd_cholesky.ipynb`

End of lecture 2