README — Matrix & Vector Norm Visualizations (Python)

Prepared for: Kosta Ylea

November 3, 2025

Contents

1 Purpose

This project demonstrates:

1. How to **generate random** 2×2 **matrices** and compute distances between them using three matrix norms:
   - Induced 1-norm (maximum absolute column sum),
   - Induced ∞-norm (maximum absolute row sum),
   - Frobenius norm (root of sum of squares).
2. How to **visualize norm-balls** as *2D slices*:
   - A slice of the $\mathbb{R}^4$ **vector ball** $\{x : \|x - x_r\| \le r\}$ by varying two coordinates while fixing two.
   - A slice of the $\mathbb{R}^{2\times 2}$ **matrix ball** $\{A : \|A - A_r\| \le r\}$ by varying two entries while fixing two.

2 Quick Start

2.1 Requirements

Python 3.10+ with:

- `numpy`
- `matplotlib`

Install (one time):

```
pip install numpy matplotlib
# if pip isn't on PATH on Windows:
py -m pip install numpy matplotlib
```

2.2 Run

Save the provided script (e.g. `norm_slices.py`), then:

```
python norm_slices.py
```

The program will:

a) Print two random matrices A, B and the distances $\mathrm{dist}(A, B)$ under three norms.

b) Pop up two figures:

- Figure 1: 2D slice of a vector norm-ball in $\mathbb{R}^4$.
- Figure 2: 2D slice of a matrix norm-ball in $\mathbb{R}^{2\times 2}$.

3 Mathematical Background

3.1 Vector norms on $\mathbb{R}^n$

For $x \in \mathbb{R}^n$ and a reference $x_r \in \mathbb{R}^n$, define $d = x - x_r$.

$$\|d\|_1 = \sum_{i=1}^{n} |d_i|, \tag{1}$$

$$\|d\|_\infty = \max_{1\le i\le n} |d_i|, \tag{2}$$

$$\|d\|_2 = \sqrt{\sum_{i=1}^{n} d_i^2}. \tag{3}$$

The *ball* of radius $r > 0$ around x_r is $\{x : \|x - x_r\| \leq r\}$.

3.2 Matrix norms on $\mathbb{R}^{m\times n}$

For $A = [a_{ij}] \in \mathbb{R}^{2\times 2}$,

$$\|A\|_1 = \max_{1\leq j\leq 2} \sum_{i=1}^{2} |a_{ij}| \quad \text{(max column sum)}, \tag{4}$$

$$\|A\|_\infty = \max_{1\leq i\leq 2} \sum_{j=1}^{2} |a_{ij}| \quad \text{(max row sum)}, \tag{5}$$

$$\|A\|_F = \sqrt{\sum_{i=1}^{2} \sum_{j=1}^{2} a_{ij}^2} \quad \text{(Frobenius)}. \tag{6}$$

Given two matrices A, B, a **distance** can be defined by

$$d(A, B) = \|A - B\|.$$

4 Script Walkthrough (Line-by-Line Concepts)

4.1 Random matrix generation

Function: `generate_2X2_Matrix()`
Creates a 2×2 list-of-lists with random integers in $[1, 10]$.

a) Start with an empty list $\to$ will hold two rows.

b) For each row $i \in \{0, 1\}$, create a temporary `row` list.

c) For each column $j \in \{0, 1\}$, sample `randint(1,10)` and append.

d) Append the row to the matrix; return the final 2×2 structure.

4.2 Induced 1-norm of a matrix

Function: `first_norm_induced_matrix(matrix)`
Implements $\|A\|_1 = \max_j \sum_i |a_{ij}|$.

- Loop over columns j, sum $|a_{0j}| + |a_{1j}|$.
- Track the maximum column sum; return it.

4.3 Induced ∞-norm of a matrix

Function: `induced_inf_norm(matrix)`
Implements $\|A\|_\infty = \max_i \sum_j |a_{ij}|$.

- Loop over rows i, sum $|a_{i0}| + |a_{i1}|$.
- Track the maximum row sum; return it.

4.4 Matrix difference and distances

Function: `matrix_difference(A,B)`
Builds elementwise $A - B$ as a new 2×2 list-of-lists.
Function: `distance_matrix(A,B,norm=...)`
Computes

$$d(A, B) = \begin{cases} \|A - B\|_1 & \text{if } \texttt{norm='induced1'}, \\ \|A - B\|_\infty & \text{if } \texttt{norm='inducedinf'}, \\ \|A - B\|_F & \text{if } \texttt{norm='fro'}. \end{cases}$$

For the Frobenius case, we convert to a NumPy `array` and use vectorized ops to compute $\sqrt{\sum a_{ij}^2}$.

4.5 Random 4-vector and vector norms

Function: `generate_4_vector()`
Returns $[x_0, x_1, x_2, x_3]$ with entries uniformly in $[-3, 3]$.
Function: `vec_norm(x,xr,kind)`
Computes $\|x - x_r\|$ for ℓ_1, ℓ_∞, or ℓ_2 using NumPy broadcasting.

5 Plotting: 2D Slices of High-Dimensional Balls

5.1 Vector ball slice in $\mathbb{R}^4$

Function: `plot_vector_unit_ball_slice(xr, norm_kind, coords, radius, span, n)`

Idea We cannot plot a full 4-D ball. Instead, *fix two coordinates* at the reference point x_r and *vary two coordinates* on a grid. A grid point $(x[i], x[j])$ is colored "inside" if

$$\|x - x_r\|_{\texttt{norm\_kind}} \le \texttt{radius}.$$

Parameters

- `xr`: center $x_r \in \mathbb{R}^4$.
- `norm_kind` $\in \{\ell_1, \ell_\infty, \ell_2\}$: which distance.
- `coords=(i,j)`: the two coordinates to vary/plot (others fixed).
- `radius`: ball radius r.
- `span`: half-width of plotting window around $x_r[i], x_r[j]$.
- `n`: grid resolution per axis ($n \times n$ points).

Algorithm

a) Build a 2D grid (X_I, X_J) around $(x_r[i], x_r[j])$.

b) Create an $(n \times n \times 4)$ tensor X with two free coords from the grid and two fixed coords equal to x_r.

c) Compute the chosen norm of $X - x_r$ at every grid point.

d) Mark points with value $\le r$ as inside; display with `imshow`.

5.2 Matrix ball slice in $\mathbb{R}^{2\times 2}$

Function: `plot_matrix_unit_ball_slice(Ar, norm, vary, radius, span, n)`

Idea We consider the set $\{A : \|A - A_r\| \leq r\}$ where $A \in \mathbb{R}^{2\times 2}$. Fix two matrix entries equal to A_r and vary two entries over a grid. For each grid point, form the candidate matrix A and test whether it lies inside the ball.

Parameters

- `Ar`: center matrix A_r.
- `norm` $\in$ {induced1, inducedinf, fro}.
- `vary=((p,q),(r,s))`: the two entries of A to vary on axes.
- `radius, span, n`: as before.

Algorithm

a) Build 1D grids for the two chosen entries: a_{pq} and a_{rs}.

b) For each grid cell, clone A_r, set the two entries to the cell values, compute $\|A - A_r\|$ in the requested norm.

c) Mark inside ($\leq r$) and display with `imshow`.

6 Figures

Place your PNGs in the same folder as this `.tex` file.

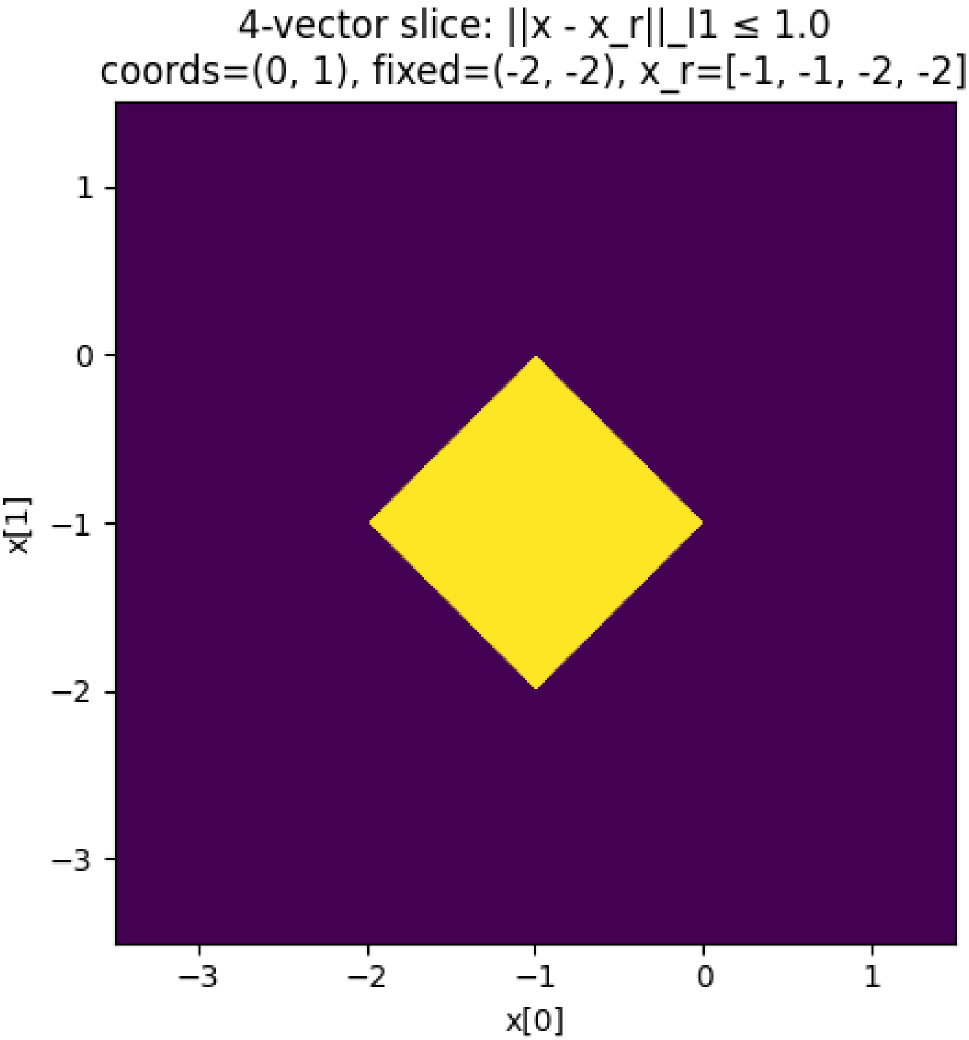

Figure 1: Vector ball slice in $\mathbb{R}^4$. Brighter region indicates points $(x[i], x[j])$ that satisfy $\|x - x_r\| \leq r$ with the two other coordinates fixed at x_r.

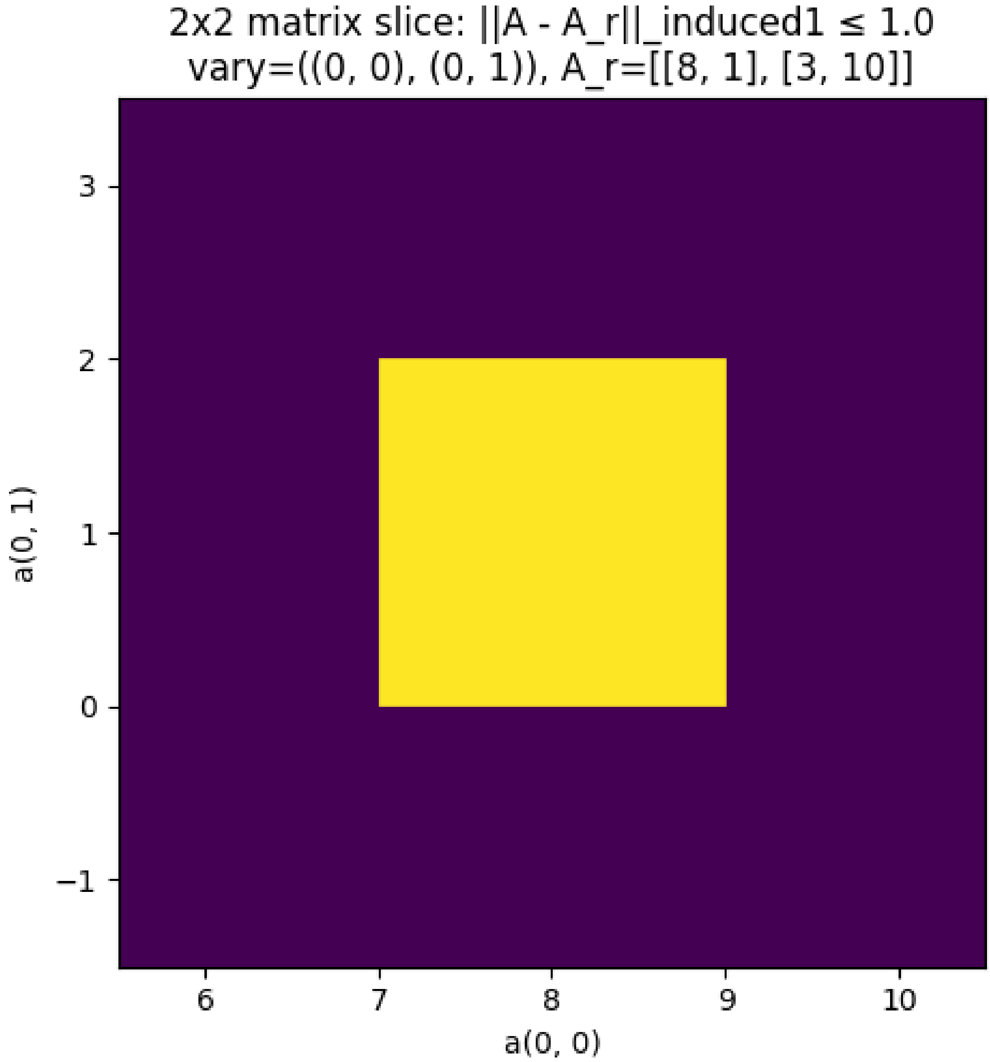

Figure 2: Matrix ball slice in $\mathbb{R}^{2\times 2}$. Each pixel corresponds to two varied entries; white = inside $\{A : \|A - A_r\| \leq r\}$.

7 Interpreting the Plots

Vector slice (Fig. 1)

- ℓ_1 (diamond-shaped cross-sections): level sets satisfy $|x_0 - x_{r,0}| + \cdots + |x_3 - x_{r,3}| = r$.
- ℓ_∞ (square cross-sections): level sets satisfy $\max_k |x_k - x_{r,k}| = r$.
- ℓ_2 (circular cross-sections): level sets satisfy $\sum_k (x_k - x_{r,k})^2 = r^2$.

Sharper corners indicate norms that penalize coordinatewise maxima more strongly; rounder shapes correspond to ℓ_2.

Matrix slice (Fig. 2)

- With induced 1- or ∞-norms, the geometry is driven by *column* or *row* sums of magnitudes, respectively.
- With Frobenius, the boundary is quadratic in all entries (spherical in $\mathbb{R}^4$ when flattening the matrix).

8 Parameters & Tuning (Cheat Sheet)

- `radius`: bigger = larger "inside" region.
- `span`: how far the axes extend from the center; increase to see more context.
- `n`: grid resolution; higher values yield smoother edges but cost more CPU.
- `coords` (vector slice): choose which two dimensions to render (e.g. `(0,1)`, `(2,3)`).
- `vary` (matrix slice): choose which entries of A to vary (e.g. `((0,0),(0,1))`).

9 Complexity & Performance Tips

- The vector slice is fully vectorized and scales roughly with $O(n^2)$ points.
- The matrix slice uses a simple double loop over n^2 grid cells; each cell computes a small constant-time norm for 2×2 (still fast).
- If plots feel slow, reduce `n` (e.g., 300→150), or lower `span`.

10 Troubleshooting

- **No module named `numpy`/`matplotlib`**: install with `pip` or `py -m pip`.
- **Plots not showing**: ensure you call the script with a Python that has GUI backend support, or run in an environment like VS Code, PyCharm, or Jupyter.
- **Images not found in LaTeX**: ensure the PNGs sit next to this `.tex` file and the filenames in `\includegraphics` match exactly.

11 Appendix: Norm Identities (for 2×2)

Let $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$. Then:

$$\|A\|_1 = \max\{|a| + |c|,\ |b| + |d|\}, \qquad \|A\|_\infty = \max\{|a| + |b|,\ |c| + |d|\},$$

$$\|A\|_F = \sqrt{a^2 + b^2 + c^2 + d^2}.$$

For vectors $x = (x_0, \ldots, x_3)$ and $x_r = (r_0, \ldots, r_3)$, with $d = x - x_r$:

$$\|d\|_1 = \sum_{k=0}^{3} |d_k|, \quad \|d\|_\infty = \max_k |d_k|, \quad \|d\|_2 = \sqrt{\sum_{k=0}^{3} d_k^2}.$$

12 Typical Workflow

1. Run the script to generate A, B and x_r; inspect printed norms.
2. Observe Figure 1 (vector slice) to understand the geometry of $\ell_1/\ell_\infty/\ell_2$.
3. Observe Figure 2 (matrix slice) to see how changing two entries moves A inside/outside the ball around A_r.
4. Adjust `radius/span/n` and re-run to test sensitivity.