Example: Maximizing the Minimum Eigenvalue of a Matrix Using SDP

Given a symmetric matrix X, we want to **maximize its smallest eigenvalue** while ensuring it remains positive semidefinite (PSD). This is a classic **SDP problem** because eigenvalues define the PSD constraint.

Problem Formulation

Let's define a symmetric **matrix variable** X **of size** 2×2:

$$X = \begin{bmatrix} x_{11} & x_{12} \\ x_{12} & x_{22} \end{bmatrix}$$

We seek to **maximize the smallest eigenvalue** of X, denoted as $\lambda_{\min}(X)$, subject to constraints.

SDP Formulation

1. Introduce a new scalar variable λ, representing the minimum eigenvalue.
2. Impose the **LMI (Linear Matrix Inequality) constraint**:
$$X - \lambda I \succeq 0$$
This ensures that all eigenvalues of X are at least λ.
3. Solve for **maximum** λ.

Thus, the SDP problem becomes:

$$\max_{\lambda, X} \quad \lambda$$

Subject to:
$$X - \lambda I \succeq 0$$ $$X \succeq 0$$
This ensures that the **minimum eigenvalue of** X **is maximized**, while X remains PSD.

Brief Explanation of the Code

This code **solves a semidefinite programming (SDP) problem** using the **CVXPY** library to **maximize the smallest eigenvalue** of a symmetric 2 x 2 matrix X.

Libraries Used:

- **cvxpy (cp)** → Defines and solves convex optimization problems.
- **numpy (np)** → Used for numerical operations (e.g., creating the identity matrix).

How It Works:

1. **Define Variables:**
   - X → A **symmetric** 2 \times 2 matrix.
   - `lambda_min` → The smallest eigenvalue to **maximize**.

2. **Define Constraints:**
   - `X - lambda_min * I >> 0` → Ensures \lambda_{\min} is the smallest eigenvalue.
   - `X >> 0` → Ensures X is **positive semidefinite (PSD)**.
   - `cp.trace(X) == 1` → **Prevents unbounded solutions**.
3. **Define Objective:**
   - **Maximize** `lambda_min`.
4. **Solve the Problem:**
   - Uses the **CVXOPT solver** (`problem.solve(solver=cp.CVXOPT)`).

CVXOPT → More reliable for SDP problems.

5. **Print Results:**
   - Displays the **solver status, optimal minimum eigenvalue, and optimal** X **matrix**.

Results:

Solving SDP problem...

Solver status: optimal

Optimal minimum eigenvalue: 0.4999999999999998

Optimal X matrix:

[[0.5 0.]

[0. 0.5]]