# Lecture 14: Positive Semidefinite Matrices & Semidefinite Programming


Rafael Oliveira

University of Waterloo
Cheriton School of Computer Science

rafael.oliveira.teaching@gmail.com


June 12, 2024

# Overview

- Positive Semidefinite Matrices
- Why Semidefinite Programming?
- Convex Algebraic Geometry
- Application: Control Theory
- Conclusion
- Acknowledgements

# Symmetric Matrices & Spectral Theorem

- A matrix $S \in \mathrm{Mat}(n, \mathbb{R})$ is symmetric if $S_{ij} = S_{ji}$ for all $i, j \in [n]$.

# Symmetric Matrices & Spectral Theorem

- A matrix $S \in \mathrm{Mat}(n, \mathbb{R})$ is symmetric if $S_{ij} = S_{ji}$ for all $i, j \in [n]$.
- $\lambda \in \mathbb{C}$ is an *eigenvalue* of $S$ if there exists $u \in \mathbb{C}^n$ such that $Su = \lambda u$. The vector $u$ is an *eigenvector* of $S$ corresponding to $\lambda$.

# Symmetric Matrices & Spectral Theorem

- A matrix $S \in \text{Mat}(n, \mathbb{R})$ is symmetric if $S_{ij} = S_{ji}$ for all $i, j \in [n]$.
- $\lambda \in \mathbb{C}$ is an *eigenvalue* of $S$ if there exists $u \in \mathbb{C}^n$ such that $Su = \lambda u$. The vector $u$ is an *eigenvector* of $S$ corresponding to $\lambda$.
- **Spectral theorem:** any symmetric matrix in $\text{Mat}(n, \mathbb{R})$ has $n$ real eigenvalues (counting with multiplicity), as well as an orthonormal basis (in $\mathbb{R}^n$) for the eigenvectors.

# Symmetric Matrices & Spectral Theorem

- A matrix $S \in \mathrm{Mat}(n, \mathbb{R})$ is symmetric if $S_{ij} = S_{ji}$ for all $i, j \in [n]$.
- $\lambda \in \mathbb{C}$ is an *eigenvalue* of $S$ if there exists $u \in \mathbb{C}^n$ such that $Su = \lambda u$. The vector $u$ is an *eigenvector* of $S$ corresponding to $\lambda$.
- **Spectral theorem:** any symmetric matrix in $\mathrm{Mat}(n, \mathbb{R})$ has $n$ real eigenvalues (counting with multiplicity), as well as an orthonormal basis (in $\mathbb{R}^n$) for the eigenvectors.
- In other words, we can write

$$S = \sum_{i=1}^{n} \lambda_i u_i u_i^T$$

where $\lambda_i \in \mathbb{R}$ and $u_i \in \mathbb{R}^n$ such that $\langle u_i, u_j \rangle = \delta_{ij}$.

# Characterizations of Positive Semidefinite Matrices

- If a symmetric matrix $S \in \mathrm{Mat}(n, \mathbb{R})$ only has non-negative eigenvalues, we say that $S$ is *positive semidefinite* (PSD), and we write $S \succeq 0$.

# Characterizations of Positive Semidefinite Matrices

- If a symmetric matrix $S \in \mathrm{Mat}(n, \mathbb{R})$ only has non-negative eigenvalues, we say that $S$ is *positive semidefinite* (PSD), and we write $S \succeq 0$.
- There are several equivalent characterizations of PSD matrices:
  1. all eigenvalues of $S$ are non-negative

# Characterizations of Positive Semidefinite Matrices

- If a symmetric matrix $S \in \mathrm{Mat}(n, \mathbb{R})$ only has non-negative eigenvalues, we say that $S$ is *positive semidefinite* (PSD), and we write $S \succeq 0$.
- There are several equivalent characterizations of PSD matrices:
  1. all eigenvalues of $S$ are non-negative
  2. $S = Y^T Y$ for some $Y \in \mathbb{R}^{d \times n}$, where $d \leq n$

# Characterizations of Positive Semidefinite Matrices

- If a symmetric matrix $S \in \mathrm{Mat}(n, \mathbb{R})$ only has non-negative eigenvalues, we say that $S$ is *positive semidefinite* (PSD), and we write $S \succeq 0$.
- There are several equivalent characterizations of PSD matrices:
  1. all eigenvalues of $S$ are non-negative
  2. $S = Y^T Y$ for some $Y \in \mathbb{R}^{d \times n}$, where $d \leq n$
  3. $x^T S x \geq 0$ for all $x \in \mathbb{R}^n$

# Characterizations of Positive Semidefinite Matrices

- If a symmetric matrix $S \in \mathrm{Mat}(n, \mathbb{R})$ only has non-negative eigenvalues, we say that $S$ is *positive semidefinite* (PSD), and we write $S \succeq 0$.
- There are several equivalent characterizations of PSD matrices:
  1. all eigenvalues of $S$ are non-negative
  2. $S = Y^T Y$ for some $Y \in \mathbb{R}^{d \times n}$, where $d \leq n$
  3. $x^T S x \geq 0$ for all $x \in \mathbb{R}^n$
  4. $S = LDL^T$, where $D$ is diagonal and non-negative, and $L$ is unit lower-triangular

# Characterizations of Positive Semidefinite Matrices

- If a symmetric matrix $S \in \mathrm{Mat}(n, \mathbb{R})$ only has non-negative eigenvalues, we say that $S$ is *positive semidefinite* (PSD), and we write $S \succeq 0$.
- There are several equivalent characterizations of PSD matrices:
  1. all eigenvalues of $S$ are non-negative
  2. $S = Y^T Y$ for some $Y \in \mathbb{R}^{d \times n}$, where $d \leq n$
  3. $x^T S x \geq 0$ for all $x \in \mathbb{R}^n$
  4. $S = LDL^T$, where $D$ is diagonal and non-negative, and $L$ is unit lower-triangular
  5. $S$ is in the convex hull of the set

$$\{uu^T \mid u \in \mathbb{R}^n\}$$

# Characterizations of Positive Semidefinite Matrices

- If a symmetric matrix $S \in \mathrm{Mat}(n, \mathbb{R})$ only has non-negative eigenvalues, we say that $S$ is *positive semidefinite* (PSD), and we write $S \succeq 0$.
- There are several equivalent characterizations of PSD matrices:
  1. all eigenvalues of $S$ are non-negative
  2. $S = Y^T Y$ for some $Y \in \mathbb{R}^{d \times n}$, where $d \leq n$
  3. $x^T S x \geq 0$ for all $x \in \mathbb{R}^n$
  4. $S = LDL^T$, where $D$ is diagonal and non-negative, and $L$ is unit lower-triangular
  5. $S$ is in the convex hull of the set
  $$\{uu^T \mid u \in \mathbb{R}^n\}$$
  6. $S = U^T D U$, where $D$ is diagonal and non-negative and $U \in \mathrm{Mat}(n, \mathbb{R})$ is orthonormal matrix (that is, $U^T U = I$).

# Characterizations of Positive Semidefinite Matrices

- If a symmetric matrix $S \in \mathrm{Mat}(n, \mathbb{R})$ only has non-negative eigenvalues, we say that $S$ is *positive semidefinite* (PSD), and we write $S \succeq 0$.
- There are several equivalent characterizations of PSD matrices:
  1. all eigenvalues of $S$ are non-negative
  2. $S = Y^T Y$ for some $Y \in \mathbb{R}^{d \times n}$, where $d \leq n$
  3. $x^T S x \geq 0$ for all $x \in \mathbb{R}^n$
  4. $S = LDL^T$, where $D$ is diagonal and non-negative, and $L$ is unit lower-triangular
  5. $S$ is in the convex hull of the set
     $$\{uu^T \mid u \in \mathbb{R}^n\}$$
  6. $S = U^T DU$, where $D$ is diagonal and non-negative and $U \in \mathrm{Mat}(n, \mathbb{R})$ is orthonormal matrix (that is, $U^T U = I$).
  7. Any principal minor of $S$ has non-negative determinant

# Characterizations of Positive Semidefinite Matrices

- If a symmetric matrix $S \in \mathrm{Mat}(n, \mathbb{R})$ only has non-negative eigenvalues, we say that $S$ is *positive semidefinite* (PSD), and we write $S \succeq 0$.
- There are several equivalent characterizations of PSD matrices:
  1. all eigenvalues of $S$ are non-negative
  2. $S = Y^T Y$ for some $Y \in \mathbb{R}^{d \times n}$, where $d \leq n$
  3. $x^T S x \geq 0$ for all $x \in \mathbb{R}^n$
  4. $S = LDL^T$, where $D$ is diagonal and non-negative, and $L$ is unit lower-triangular
  5. $S$ is in the convex hull of the set

     $$\{uu^T \mid u \in \mathbb{R}^n\}$$

  6. $S = U^T D U$, where $D$ is diagonal and non-negative and $U \in \mathrm{Mat}(n, \mathbb{R})$ is orthonormal matrix (that is, $U^T U = I$).
  7. Any principal minor of $S$ has non-negative determinant
- **Practice problem:** prove that these are all equivalent!

- Positive Semidefinite Matrices
- Why Semidefinite Programming?
- Convex Algebraic Geometry
- Application: Control Theory
- Conclusion
- Acknowledgements

# Mathematical Programming

**Mathematical Programming** deals with problems of the form

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & g_1(x) \geq 0 \\ & \quad \vdots \\ & g_m(x) \geq 0 \\ & x \in \mathbb{R}^n \end{aligned}$$

# Mathematical Programming

**Mathematical Programming** deals with problems of the form

$$
\begin{aligned}
\text{minimize} \quad & f(x) \\
\text{subject to} \quad & g_1(x) \geq 0 \\
& \quad \vdots \\
& g_m(x) \geq 0 \\
& x \in \mathbb{R}^n
\end{aligned}
$$

- Very general family of problems.

# Mathematical Programming

**Mathematical Programming** deals with problems of the form

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & g_1(x) \geq 0 \\ & \vdots \\ & g_m(x) \geq 0 \\ & x \in \mathbb{R}^n \end{aligned}$$

- Very general family of problems.
- Special case when all $f, g_1, \ldots, g_m$ are *linear*. *Linear Programming*

# Mathematical Programming

**Mathematical Programming** deals with problems of the form

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & g_1(x) \geq 0 \\ & \vdots \\ & g_m(x) \geq 0 \\ & x \in \mathbb{R}^n \end{aligned}$$

- Very general family of problems.
- Special case when all $f, g_1, \ldots, g_m$ are *linear*. *Linear Programming*
- More general case: *Semidefinite Programming*
  1. $A_1, \ldots, A_n, B \in \mathcal{S}^m$ are $m \times m$ symmetric matrices

# Mathematical Programming

**Mathematical Programming** deals with problems of the form

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & g_1(x) \geq 0 \\ & \vdots \\ & g_m(x) \geq 0 \\ & x \in \mathbb{R}^n \end{aligned}$$

- Very general family of problems.
- Special case when all $f, g_1, \ldots, g_m$ are *linear*. *Linear Programming*
- More general case: *Semidefinite Programming*
  1. $A_1, \ldots, A_n, B \in \mathcal{S}^m$ are $m \times m$ symmetric matrices
  2. Constraints:
  $$x_1 \cdot A_1 + \cdots + x_n \cdot A_n \succeq B$$
  3. Minimize linear function $c^T x$

# What is a Semidefinite Program?

- $\mathcal{S}^m := \mathcal{S}^m(\mathbb{R})$ space of all $m \times m$ symmetric matrices (real entries)

# What is a Semidefinite Program?

- $\mathcal{S}^m := \mathcal{S}^m(\mathbb{R})$ space of all $m \times m$ symmetric matrices (real entries)

**Semidefinite Programming** deals with problems of the form

$$
\begin{array}{rl}
\text{minimize} & c^T x \\
\text{subject to} & x_1 \cdot A_1 + \cdots + x_n \cdot A_n \succeq B \\
& x \in \mathbb{R}^n \\
& A_i, B \in \mathcal{S}^m(\mathbb{R}) \text{ (fixed matrices)}
\end{array}
$$

# What is a Semidefinite Program?

- $\mathcal{S}^m := \mathcal{S}^m(\mathbb{R})$ space of all $m \times m$ symmetric matrices (real entries)

**Semidefinite Programming** deals with problems of the form

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & x_1 \cdot A_1 + \cdots + x_n \cdot A_n \succeq B \\ & x \in \mathbb{R}^n \\ & A_i, B \in \mathcal{S}^m(\mathbb{R}) \text{ (fixed matrices)} \end{aligned}$$

Where we use $C \succeq D$ to denote that $C - D \succeq 0$ (i.e., $C - D$ is PSD).

# How does it generalize Linear Programming?

**Linear Programming**

$$
\begin{aligned}
\text{minimize} \quad & c^T x \\
\text{subject to} \quad & Ax \geq b \\
& x \in \mathbb{R}^n
\end{aligned}
$$

# How does it generalize Linear Programming?

**Linear Programming**

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & Ax \geq b \\ & x \in \mathbb{R}^n \end{aligned}$$

**Semidefinite Programming**

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & x_1 \cdot A_1 + \cdots + x_n \cdot A_n \succeq B \\ & x \in \mathbb{R}^n \end{aligned}$$

# How does it generalize Linear Programming?

**Linear Programming**

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \geq b \\ & x \in \mathbb{R}^n \end{array}$$

**Semidefinite Programming**

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x_1 \cdot A_1 + \cdots + x_n \cdot A_n \succeq B \\ & x \in \mathbb{R}^n \end{array}$$

Set $A_i$'s to be diagonal matrices, and $B = diag(b_1, \ldots, b_m)$

# Why should I care?

- Linear Programs appear everywhere in life: many problems of interest (resource allocation problems) can be modelled as linear program!

# Why should I care?

- Linear Programs appear everywhere in life: many problems of interest (resource allocation problems) can be modelled as linear program!
- Semidefinite Programming is no different!

# Why should I care?

- Linear Programs appear everywhere in life: many problems of interest (resource allocation problems) can be modelled as linear program!
- Semidefinite Programming is no different!
  - equilibrium analysis of dynamics and control (flight controls, robotics, etc.)
  - robust optimization
  - statistics and ML
  - continuous games
  - software verification
  - filter design
  - quantum computation and information
  - automated theorem proving
  - packing problems
  - many more

# Why should I care?

- Linear Programs appear everywhere in life: many problems of interest (resource allocation problems) can be modelled as linear program!
- Semidefinite Programming is no different!
  - equilibrium analysis of dynamics and control (flight controls, robotics, etc.)
  - robust optimization
  - statistics and ML
  - continuous games
  - software verification
  - filter design
  - quantum computation and information
  - automated theorem proving
  - packing problems
  - many more
- See more here

`https://windowsontheory.org/2016/08/27/proofs-beliefs-and-algorithms-through-the-lens-of-sum-of-squares/`

# Important Questions

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & x_1 \cdot A_1 + \cdots + x_n \cdot A_n \succeq B \\ & x \in \mathbb{R}^n \end{aligned}$$

# Important Questions

$$\begin{array}{rl} \text{minimize} & c^T x \\ \text{subject to} & x_1 \cdot A_1 + \cdots + x_n \cdot A_n \succeq B \\ & x \in \mathbb{R}^n \end{array}$$

1. When is a Semidefinite Program *feasible*?
   - Is there a solution to the constraints at all?

# Important Questions

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & x_1 \cdot A_1 + \cdots + x_n \cdot A_n \succeq B \\ & x \in \mathbb{R}^n \end{aligned}$$

1. When is a Semidefinite Program *feasible*?
   - Is there a solution to the constraints at all?
2. When is a Semidefinite Program *bounded*?
   - Is there a minimum? Is the minimum achievable? Or is the minimum $-\infty$?

# Important Questions

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & x_1 \cdot A_1 + \cdots + x_n \cdot A_n \succeq B \\ & x \in \mathbb{R}^n \end{aligned}$$

1. When is a Semidefinite Program *feasible*?
   - Is there a solution to the constraints at all?
2. When is a Semidefinite Program *bounded*?
   - Is there a minimum? Is the minimum achievable? Or is the minimum $-\infty$?
3. Can we characterize *optimality*?
   - How can we know that we found a minimum solution?
   - Do these solutions have nice description?
   - Do the solutions have *small bit complexity*?

# Important Questions

$$
\begin{aligned}
\text{minimize} \quad & c^T x \\
\text{subject to} \quad & x_1 \cdot A_1 + \cdots + x_n \cdot A_n \succeq B \\
& x \in \mathbb{R}^n
\end{aligned}
$$

1. When is a Semidefinite Program *feasible*?
   - Is there a solution to the constraints at all?
2. When is a Semidefinite Program *bounded*?
   - Is there a minimum? Is the minimum achievable? Or is the minimum $-\infty$?
3. Can we characterize *optimality*?
   - How can we know that we found a minimum solution?
   - Do these solutions have nice description?
   - Do the solutions have *small bit complexity*?
4. How do we design *efficient algorithms* that find *optimal solutions* to Semidefinite Programs?

- Positive Semidefinite Matrices
- Why Semidefinite Programming?
- Convex Algebraic Geometry
- Application: Control Theory
- Conclusion
- Acknowledgements

# Spectrahedra

To understand SDPs, we need to understand their *feasible regions*, which are called *spectrahedra* and described as *Linear Matrix Inequalities* (LMIs).

# Spectrahedra

To understand SDPs, we need to understand their *feasible regions*, which are called *spectrahedra* and described as *Linear Matrix Inequalities* (LMIs).

**Definition (Linear Matrix Inequalities)**

A linear matrix inequality is an inequality of the form:

$$A_0 + \sum_{i=1}^{n} A_i x_i \succeq 0,$$

where $A_0, \ldots, A_n$ are *symmetric matrices*.

# Spectrahedra

To understand SDPs, we need to understand their *feasible regions*, which are called *spectrahedra* and described as *Linear Matrix Inequalities* (LMIs).

## Definition (Linear Matrix Inequalities)

A linear matrix inequality is an inequality of the form:

$$A_0 + \sum_{i=1}^{n} A_i x_i \succeq 0,$$

where $A_0, \ldots, A_n$ are *symmetric matrices*.

## Definition (Spectrahedron)

A spectrahedron is a set defined by finitely many LMIs. In other words, it can be defined as:

$$S = \left\{ x \in \mathbb{R}^n \;\middle|\; \sum_{i=1}^{n} A_i x_i \succeq B, \quad A_i, B \in \mathcal{S}^m \right\}$$

# Spectrahedra

To understand SDPs, we need to understand their *feasible regions*, which are called *spectrahedra* and described as *Linear Matrix Inequalities* (LMIs).

## Definition (Spectrahedron)

A spectrahedron is a set defined by finitely many LMIs. In other words, it can be defined as:

$$S = \left\{ x \in \mathbb{R}^n \;\middle|\; \sum_{i=1}^{n} A_i x_i \succeq B, \quad A_i, B \in \mathcal{S}^m \right\}$$

# Spectrahedra

To understand SDPs, we need to understand their *feasible regions*, which are called *spectrahedra* and described as *Linear Matrix Inequalities* (LMIs).

**Definition (Spectrahedron)**

A spectrahedron is a set defined by finitely many LMIs. In other words, it can be defined as:

$$S = \left\{ x \in \mathbb{R}^n \ \middle|\ \sum_{i=1}^{n} A_i x_i \succeq B, \quad A_i, B \in \mathcal{S}^m \right\}$$

# Example of Spectrahedron

Polyhedron:

# Example of Spectrahedron

Circle:

# Example of Spectrahedron

Hyperbola:

# Example of Spectrahedron

Elliptic curve:

# Projected Spectrahedron

For both LPs and SDPs, it is enough to obtain a *linear projection of spectrahedron* (or polyhedron, if in LP).

# Projected Spectrahedron

For both LPs and SDPs, it is enough to obtain a *linear projection of spectrahedron* (or polyhedron, if in LP).

## Definition (Projected Spectrahedron)

A set $S \in \mathbb{R}^n$ is a *projected spectrahedron* if it has the form:

$$S = \left\{ x \in \mathbb{R}^n \;\middle|\; \exists y \in \mathbb{R}^t \text{ s.t. } \sum_{i=1}^{n} A_i x_i + \sum_{j=1}^{t} B_j y_j \succeq C, \quad A_i, B_j, C \in \mathcal{S}^m \right\}$$

# Projected Spectrahedron

For both LPs and SDPs, it is enough to obtain a *linear projection of spectrahedron* (or polyhedron, if in LP).

## Definition (Projected Spectrahedron)

A set $S \in \mathbb{R}^n$ is a *projected spectrahedron* if it has the form:

$$S = \left\{ x \in \mathbb{R}^n \;\middle|\; \exists y \in \mathbb{R}^t \text{ s.t. } \sum_{i=1}^{n} A_i x_i + \sum_{j=1}^{t} B_j y_j \succeq C, \quad A_i, B_j, C \in \mathcal{S}^m \right\}$$

# Example of Projected Spectrahedron

Projection of hyperbola:

# Example of Projected Spectrahedron

Projection quadratic cone intersected with halfspace:

# How do we test membership in the Spectrahedron?

- To be able to optimize, we must be able to test whether a given point $x \in \mathbb{R}^n$ is inside our spectrahedron

$$S = \left\{ x \in \mathbb{R}^n \mid \sum_{i=1}^{n} A_i x_i \succeq B, \quad A_i, B \in \mathcal{S}^m \right\}.$$

# How do we test membership in the Spectrahedron?

- To be able to optimize, we must be able to test whether a given point $x \in \mathbb{R}^n$ is inside our spectrahedron

$$S = \left\{ x \in \mathbb{R}^n \;\middle|\; \sum_{i=1}^{n} A_i x_i \succeq B, \quad A_i, B \in \mathcal{S}^m \right\}.$$

- Note that $x \in S$ is (by definition) equivalent to

$$Z = \sum_{i=1}^{n} A_i x_i - B \succeq 0$$

# How do we test membership in the Spectrahedron?

- To be able to optimize, we must be able to test whether a given point $x \in \mathbb{R}^n$ is inside our spectrahedron

$$S = \left\{ x \in \mathbb{R}^n \;\middle|\; \sum_{i=1}^{n} A_i x_i \succeq B, \quad A_i, B \in \mathcal{S}^m \right\}.$$

- Note that $x \in S$ is (by definition) equivalent to

$$Z = \sum_{i=1}^{n} A_i x_i - B \succeq 0$$

- So, how do we efficiently check if $Z \succeq 0$?

# How do we test membership in the Spectrahedron?

- To be able to optimize, we must be able to test whether a given point $x \in \mathbb{R}^n$ is inside our spectrahedron

$$S = \left\{ x \in \mathbb{R}^n \;\middle|\; \sum_{i=1}^{n} A_i x_i \succeq B, \quad A_i, B \in \mathcal{S}^m \right\}.$$

- Note that $x \in S$ is (by definition) equivalent to

$$Z = \sum_{i=1}^{n} A_i x_i - B \succeq 0$$

- So, how do we efficiently check if $Z \succeq 0$?
- Symmetric Gaussian Elimination!

# How do we test membership in the Spectrahedron?

- To be able to optimize, we must be able to test whether a given point $x \in \mathbb{R}^n$ is inside our spectrahedron

$$S = \left\{ x \in \mathbb{R}^n \ \middle| \ \sum_{i=1}^{n} A_i x_i \succeq B, \quad A_i, B \in \mathcal{S}^m \right\}.$$

- Note that $x \in S$ is (by definition) equivalent to

$$Z = \sum_{i=1}^{n} A_i x_i - B \succeq 0$$

- So, how do we efficiently check if $Z \succeq 0$?
- Symmetric Gaussian Elimination!
- We will use following characterizations of PSDness of symmetric $A \in \mathcal{S}^m$
  1. all eigenvalues of $A$ are *non-negative*
  2. $A = LDL^T$ for some $L$ lower triangular and unit diagonal, $D$ diagonal and non-negative
  3. $z^T A z \geq 0$ for any $z \in \mathbb{R}^m$
  4. Any principal minor of $A$ has non-negative determinant

# How do we test membership in the Spectrahedron?

- **Input:** symmetric matrix $A \in \mathcal{S}^m$
- **Output:** YES if $A \succeq 0$, NO otherwise (and output $z \in \mathbb{R}^m$ such that $z^T A z < 0$)

# How do we test membership in the Spectrahedron?

- **Input:** symmetric matrix $A \in \mathcal{S}^m$
- **Output:** YES if $A \succeq 0$, NO otherwise (and output $z \in \mathbb{R}^m$ such that $z^T A z < 0$)

- Our algorithm runs in time strongly polynomial.

- Positive Semidefinite Matrices
- Why Semidefinite Programming?
- Convex Algebraic Geometry
- Application: Control Theory
- Conclusion
- Acknowledgements

# Stability of Linear Systems

Setup:

- Linear difference equation

$$x(t+1) = Ax(t), \quad x(0) = x_0$$

- Discrete-time dynamical system.[1]

[1] When $A$ non-negative and $x_0$ non-negative we have Markov chains.

# Stability of Linear Systems

Setup:

- Linear difference equation

$$x(t+1) = Ax(t), \quad x(0) = x_0$$

- Discrete-time dynamical system.[1]
- Used to model time evolution of

[1] When $A$ non-negative and $x_0$ non-negative we have Markov chains.

# Stability of Linear Systems

Setup:

- Linear difference equation

$$x(t+1) = Ax(t), \quad x(0) = x_0$$

- Discrete-time dynamical system.[1]
- Used to model time evolution of
  - Temperatures of objects
  - Size of population
  - Voltage of electrical circuits
  - Concentration of chemical mixtures

[1] When $A$ non-negative and $x_0$ non-negative we have Markov chains.

# Stability of Linear Systems

Setup:

- Linear difference equation

$$x(t+1) = Ax(t), \quad x(0) = x_0$$

- Discrete-time dynamical system.[1]
- Used to model time evolution of
  - Temperatures of objects
  - Size of population
  - Voltage of electrical circuits
  - Concentration of chemical mixtures
- **Question:** when $t \to \infty$, under what conditions will $x(t)$ remain bounded? Or go to zero?

[1] When $A$ non-negative and $x_0$ non-negative we have Markov chains.

# Stability of Linear Systems

Setup:

- Linear difference equation

$$x(t+1) = Ax(t), \quad x(0) = x_0$$

- Discrete-time dynamical system.[1]
- Used to model time evolution of
  - Temperatures of objects
  - Size of population
  - Voltage of electrical circuits
  - Concentration of chemical mixtures
- **Question:** when $t \to \infty$, under what conditions will $x(t)$ remain bounded? Or go to zero?
- When system converges to zero, we say it is *stable*.

[1] When $A$ non-negative and $x_0$ non-negative we have Markov chains.

# Stability of Linear Systems

Setup:

- Linear difference equation

$$x(t+1) = Ax(t), \quad x(0) = x_0$$

- Discrete-time dynamical system.[1]
- Used to model time evolution of
  - Temperatures of objects
  - Size of population
  - Voltage of electrical circuits
  - Concentration of chemical mixtures
- **Question:** when $t \to \infty$, under what conditions will $x(t)$ remain bounded? Or go to zero?
- When system converges to zero, we say it is *stable*.
- System is stable iff $|\lambda_i(A)| < 1$

[1] When $A$ non-negative and $x_0$ non-negative we have Markov chains.

# Stability of Linear Systems

SDP viewpoint:

- Lyapunov functions (generalize *energy* in systems). Functions on $x(t)$ decrease monotonically on trajectories of the system.

# Stability of Linear Systems

SDP viewpoint:

- Lyapunov functions (generalize *energy* in systems). Functions on $x(t)$ decrease monotonically on trajectories of the system.
- For our discrete-time system, we have:

$$V(x(t)) = x(t)^T P x(t)$$

# Stability of Linear Systems

SDP viewpoint:

- Lyapunov functions (generalize *energy* in systems). Functions on $x(t)$ decrease monotonically on trajectories of the system.
- For our discrete-time system, we have:

$$V(x(t)) = x(t)^T P x(t)$$

- To make these monotonically decreasing, we need:

$$\begin{aligned} V(x(t+1)) \leq V(x(t)) &\Leftrightarrow x(t+1)^T P x(t+1) - x(t)^T P x(t) \leq 0 \\ &\Leftrightarrow x(t)^T A^T P A x(t) - x(t)^T P x(t) \leq 0 \\ &\Leftrightarrow A^T P A - P \preceq 0 \end{aligned}$$

# Stability of Linear Systems

SDP viewpoint:

- Lyapunov functions (generalize *energy* in systems). Functions on $x(t)$ decrease monotonically on trajectories of the system.
- For our discrete-time system, we have:

$$V(x(t)) = x(t)^T P x(t)$$

- To make these monotonically decreasing, we need:

$$\begin{aligned} V(x(t+1)) \leq V(x(t)) &\Leftrightarrow x(t+1)^T P x(t+1) - x(t)^T P x(t) \leq 0 \\ &\Leftrightarrow x(t)^T A^T P A x(t) - x(t)^T P x(t) \leq 0 \\ &\Leftrightarrow A^T P A - P \preceq 0 \end{aligned}$$

**Theorem**

*Given matrix $A \in \mathbb{R}^{m \times m}$, the following conditions are equivalent:*

1. *All eigenvalues of $A$ are inside unit circle, i.e. $|\lambda_i(A)| < 1$*
2. *There is $P \in \mathcal{S}^m$ such that*

$$P \succ 0, \quad A^T P A - P \prec 0$$

# Where is the control?

Setup:

- Linear difference equation, with *control input*

$$x(t+1) = Ax(t) + Bu(t), \quad x(0) = x_0$$

where $A \in \mathbb{R}^{m \times m}$, $B \in \mathbb{R}^{m \times k}$

# Where is the control?

Setup:

- Linear difference equation, with *control input*

$$x(t+1) = Ax(t) + Bu(t), \quad x(0) = x_0$$

  where $A \in \mathbb{R}^{m\times m}$, $B \in \mathbb{R}^{m\times k}$
- If we properly choose control input $u(t)$ we can make our system $x(t)$ behave in a way that we want (say, to stabilize an unstable system)

# Where is the control?

Setup:

- Linear difference equation, with *control input*

$$x(t+1) = Ax(t) + Bu(t), \quad x(0) = x_0$$

  where $A \in \mathbb{R}^{m\times m}$, $B \in \mathbb{R}^{m\times k}$
- If we properly choose control input $u(t)$ we can make our system $x(t)$ behave in a way that we want (say, to stabilize an unstable system)
- Want to do it by setting the control input to be $u(t) = Kx(t)$ for some fixed $K$ (so that we use the system as its own feedback)

# Where is the control?

Setup:

- Linear difference equation, with *control input*

$$x(t+1) = Ax(t) + Bu(t), \quad x(0) = x_0$$

  where $A \in \mathbb{R}^{m\times m}$, $B \in \mathbb{R}^{m\times k}$
- If we properly choose control input $u(t)$ we can make our system $x(t)$ behave in a way that we want (say, to stabilize an unstable system)
- Want to do it by setting the control input to be $u(t) = Kx(t)$ for some fixed $K$ (so that we use the system as its own feedback)
- Same thing as replacing $A \leftarrow A + BK$

# Where is the control?

Setup:

- Linear difference equation, with *control input*

$$x(t+1) = Ax(t) + Bu(t), \quad x(0) = x_0$$

  where $A \in \mathbb{R}^{m \times m}$, $B \in \mathbb{R}^{m \times k}$
- If we properly choose control input $u(t)$ we can make our system $x(t)$ behave in a way that we want (say, to stabilize an unstable system)
- Want to do it by setting the control input to be $u(t) = Kx(t)$ for some fixed $K$ (so that we use the system as its own feedback)
- Same thing as replacing $A \leftarrow A + BK$
- Now this is harder to solve via simple eigenvalue description. But still solved the same way via Lyapunov functions!

# Where is the control?

Setup:

- Linear difference equation, with *control input*

$$x(t+1) = Ax(t) + Bu(t), \quad x(0) = x_0$$

  where $A \in \mathbb{R}^{m \times m}$, $B \in \mathbb{R}^{m \times k}$
- If we properly choose control input $u(t)$ we can make our system $x(t)$ behave in a way that we want (say, to stabilize an unstable system)
- Want to do it by setting the control input to be $u(t) = Kx(t)$ for some fixed $K$ (so that we use the system as its own feedback)
- Same thing as replacing $A \leftarrow A + BK$
- Now this is harder to solve via simple eigenvalue description. But still solved the same way via Lyapunov functions!
- Want $P \succ 0$ such that

$$(A + BK)^T P(A + BK) - P \prec 0$$

# Where is the control?

Setup:

- Linear difference equation, with *control input*

$$x(t+1) = Ax(t) + Bu(t), \quad x(0) = x_0$$

  where $A \in \mathbb{R}^{m \times m}$, $B \in \mathbb{R}^{m \times k}$
- If we properly choose control input $u(t)$ we can make our system $x(t)$ behave in a way that we want (say, to stabilize an unstable system)
- Want to do it by setting the control input to be $u(t) = Kx(t)$ for some fixed $K$ (so that we use the system as its own feedback)
- Same thing as replacing $A \leftarrow A + BK$
- Now this is harder to solve via simple eigenvalue description. But still solved the same way via Lyapunov functions!
- Want $P \succ 0$ such that

$$(A + BK)^T P(A + BK) - P \prec 0$$

- Wait, this ain't no SDP! But we can make it into SDP with some matrix manipulations.

# Conclusion

- Mathematical programming - very general, and pervasive in Algorithmic life
- General mathematical programming very hard

[2]pun intended

# Conclusion

- Mathematical programming - very general, and pervasive in Algorithmic life
- General mathematical programming very hard
- Special cases have very striking applications!
  - Linear Programming (previous lectures)
  - Today: *Semidefinite Programming*

[2] pun intended

# Conclusion

- Mathematical programming - very general, and pervasive in Algorithmic life
- General mathematical programming very hard
- Special cases have very striking applications!
  - Linear Programming (previous lectures)
  - Today: *Semidefinite Programming*
- Semidefinite Programming and Duality - fundamental concepts, lots of applications!
  - Applications in Combinatorial Optimization (Max-Cut in *next lecture*!)
  - Applications in Control Theory
  - many more!

[2] pun intended

# Conclusion

- Mathematical programming - very general, and pervasive in Algorithmic life
- General mathematical programming very hard
- Special cases have very striking applications!
  - Linear Programming (previous lectures)
  - Today: *Semidefinite Programming*
- Semidefinite Programming and Duality - fundamental concepts, lots of applications!
  - Applications in Combinatorial Optimization (Max-Cut in *next lecture*!)
  - Applications in Control Theory
  - many more!
- Check out connections to Sum of Squares and a **bold**[2] attempt to have one algorithm to solve all problems! (i.e., one algorithm to rule them all)

`https://windowsontheory.org/2016/08/27/proofs-beliefs-and-algorithms-through-the-lens-of-sum-of-squares/`

[2] pun intended

# Acknowledgement

- Lecture based largely on:
    - [Blekherman, Parrilo, Thomas 2012, Chapter 2]

# References I


Blekherman, Grigoriy and Parrilo, Pablo and Thomas, Rekha (2012)
*Convex Algebraic Geometry*